Knowledge Graph Embedding with Diversity of Structures

Wen Zhang
≪ supervised by Huajun Chen ≫
College of Computer Science and Technology
Zhejiang University, Hangzhou, China
wenzhang2015@zju.edu.cn

ABSTRACT

In recent years, different web knowledge graphs, both free and commercial, have been created. Knowledge graphs use relations between entities to describe facts in the world. We engage in embedding a large scale knowledge graph into a continuous vector space. TransE, TransH, TransR and TransD are promising methods proposed in recent years and achieved state-of-the-art predictive performance. In this paper, we discuss that graph structures should be considered in embedding and propose to embed substructures called "one-relation-circle" (ORC) to further improve the performance of the above methods as they are unable to encode ORC substructures. Some complex models are capable of handling ORC structures but sacrifice efficiency in the process. To make a good trade-off between the model capacity and efficiency, we propose a method to decompose ORC substructures by using two vectors to represent the entity as a head or tail entity with the same relation. In this way, we can encode the ORC structure properly when apply it to TransH, TransR and TransD with almost the same model complexity of themselves. We conduct experiments on link prediction with benchmark dataset WordNet. Our experiments show that applying our method improves the results compared with the corresponding original results of TransH, TransR and TransD.

Keywords

knowledge graph embedding, substructure diversity, link prediction,knowledge graph completion

1. PROBLEM

Knowledge Graphs (KGs) have become a crucial resource for many tasks in machine learning, data mining and artificial intelligence applications. A knowledge graph is a multi-relational graph composed of entities as nodes and relations as edges with different types. An instance of one edge is a triple (e_1, r, e_2) which describes the relation r between the first head entity e_1 and the second tail entity e_2. In the past a few decades, there have been great achievements in building large scale knowledge graphs, most notably WordNet[1] [15], Freebase [11], YAGO [8][2]and NELL [4][3]. But existing knowledge graphs are far from complete. It is necessary to develop *knowledge graph completion* (KGC) methods to find the missing triples in order to improve the general quality of KGs.

In recent years, a series of approaches have been proposed to embed a knowledge graph into a continuous vector space while preserving the properties of the knowledge graph. For example, each entity is represented as a vector in the space and each relation is modeled as a translation from the first head entity to the second tail entity. We call these embedding methods as *translate-based methods*. The representations of entities and relations are obtained by minimizing a margin-based global loss function involving all triples in knowledge graphs. As a result, the single representation of entities and relations will encode the global information and can be used in other applications, including knowledge graph completion.

Translate-based methods are based on the assumption: $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2$ for triple (e_1, r, e_2) in which $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{r}$ are vector representations for entity e_1, e_2 and relation r. We find there are flaws under such assumption when dealing with some special substructures of knowledge graphs such as "one-relation-circle" (ORC) structures. Some ORC structures respectively correspond to special kinds of relations. For example, C1, C2, C3 in Section 3 correspond to three object properties as defined in OWL 2 (Web Ontology Language)[4]: reflexive object properties, symmetric object properties and transitive object properties. To catch the semantic information of these relations, the models must be able to encode ORC structures. The ORC structure is defined as follows:

Definition 1. *(one-relation-circle) (ORC) Given a subgraph $\mathcal{G}' = \{\boldsymbol{E}', \boldsymbol{R}'\}$ of knowledge graph $\mathcal{G} = \{\boldsymbol{E}, \boldsymbol{R}\}$, m is the number of relation(edge) types in $\mathcal{G}'$. $d(e)$ is the degree(in degree + out degree) of entity e. We call $\mathcal{G}'$ is an "one-relation-circle structure" if and only if $m = 1$, $\forall e \in \mathcal{G}', d(e) = 2$.*

However, to encode the ORC structures, the corresponding relation representations must approach to zero vector under

WWW 2017 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3053380

[1]https://wordnet.princeton.edu/
[2]http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/
[3]http://rtw.ml.cmu.edu/rtw/
[4]https://www.w3.org/TR/owl2-syntax/

the assumption of translate-based methods: $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2$ for triple (e_1, r, e_2). A few previous models such as Distant [5], Single Layer [22] and NTN [22] are capable of preserving such ORC structures but the model complexity and running time also increased.

In this paper, our main contributions are: (i) we propose a method applied to translate-based models to decompose ORC structures while reserving the model complexity and efficiency. (ii) We illustrate the importance for embedding methods to handle the ORC structures in knowledge graphs.

In Section 2, we introduce related works. In Section 3, we analyze the problem of translate-based models to deal with ORC structures and proposed our method. In section 4, we introduce how to apply our method to TransH, TransR and TransD. In section 5, we conduct experiment on link prediction and analyze the results. In section 6, we draw conclusions and discuss future works.

2. STATE OF THE ART

2.1 Translate-based models

For a triple (e_1, r, e_2), translate-based models regard relation r as a translation from the first head entity to the second tail entity and they all under the assumption that $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2$. The bold letters $\mathbf{e}_1, \mathbf{e}_2, \mathbf{r}$ are representations of e_1, e_2 and r in vector spaces. One of the most notable model is TransE [6] which represents both entities and relations as vectors. As pointed out in TransH [29], TransE has difficulty in dealing with more complex relations such as 1-N, N-1 and N-N. To address this problem, TransH projects entities to a relation specific hyperplane before computing the distance between entities because the projecting operation allows different entities to have the same vector presentation on one relation hyperplane. Different from TransE and TransH to represent all elements in the same space, TransR [26] represents entities in entity space and relations in relation space which makes the model more flexible. Considering the diversity of entities and relations simultaneously, TransD [9] constructs a dynamic mapping matrix for each entity-relation pair. These models minimize a margin-based pairwise ranking loss function over the training data. TransA [25] introduces an adaptive local margin approach that determines margin by a closed set of entity candidates.

2.2 Models encoding more information besides facts

As most models only use the information of facts(triples) in knowledge graphs, there are many models considering to utilize more semantic information such as path, type constrains of entities and entity descriptions. PTransE [27] uses both facts and multiple-step relation paths between entities and PRA [16] is a model mainly considering the information of paths. KR-EAR [28] treats attributes of entities and relations between entities as different properties. REACAL [13], [7] and TKRL [19] also utilize the type information of entities. DKRL [24] and SSP [10] combine entity descriptions to represent learning of knowledge graphs. All these models argue that there are a lot of semantic information in knowledge graphs besides facts.

2.3 Other models

There are many other approaches. HolE [12] learns more expressive combination operators in order to predict the existence of triples in the KGs. KG2E [21] and TransG [23] are probability-based models. RESCAL [13] adopts the idea of tensor factorization. TATEC [3] combines two and three-ways together. SE [2], SME [1], NTN [22] and ProjE [20] are neural-based models which also attract much attention.

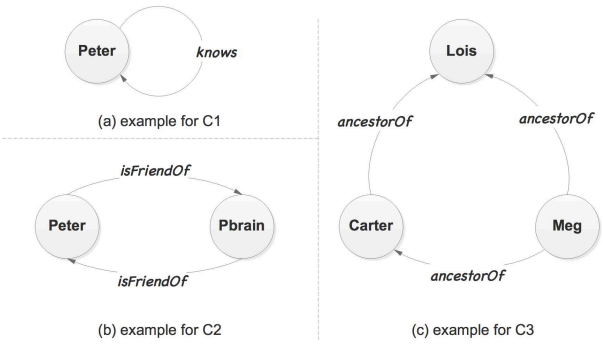

Figure 1: examples for three basic ORC structures

As translate-based methods achieve a good trade-off between predictive accuracy and computational efficiency, we mainly pay attention to translate-based methods like TransE and its extension models.

3. PROPOSED APPROACH

Firstly, we describe some common notations in this section. e_1 denotes the first head entity, e_2 is the second tail entity, r denotes the relation. The bold letters $\mathbf{e}_1, \mathbf{e}_2, \mathbf{r}$ denote the corresponding representations of e_1, e_2, r. $\triangle$ is a set of true triples in knowledge graphs and $(e_1, r, e_2) \in \triangle$ means that triple (e_1, r, e_2) is true.

In this section we introduce some basic structures of ORC and analyze the difficulty for translate-based models to encode these structures. Then we illustrate our method which could be applied to translate-based models and makes them encode ORC structures successfully.

We divide ORC structures into different types according to the number of entities(nodes). There are three main types of ORC structures: C1, C2 and C3.

3.1 C1:ORC structure with one entity

DEFINITION 2. *$C1$ is an ORC structure containing one entity and $\forall(e_1, r, e_2) \in C1, e_1 = e_2$.*

For example, we have the following triple: (Peter, *knows*, Peter) $\in \triangle$. It means the node Peter connects to itself by an edge labeled with *knows* as shown in Fig.1(a) and this triple constructs a C1.

Under the assumption of translate-based methods, for C1: $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2, \mathbf{e}_1 = \mathbf{e}_2$ which means $\mathbf{r} \approx \mathbf{0}$.

3.2 C2: ORC structure with two entities

DEFINITION 3. *$C2$ is an ORC structure containing two entities e_1, e_2. And $(e_1, r, e_2) \in C2$, $(e_2, r, e_1) \in C2$.*

For example, if x is a friend of y then y is likely to be a friend of x, so there may be exist the following triples: (Peter, $isFriendOf$, Pbrain)$\in \triangle$, (Pbrain, $isFriendOf$, Peter) $\in \triangle$. These two triples construct a C2 as shown in Fig.1(b).

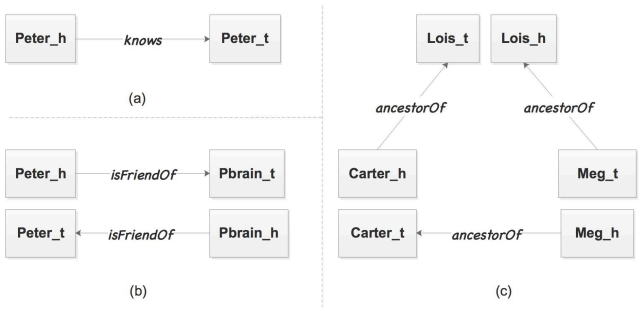

Figure 2: Exapmles to decompose the ORC

Under the assumption of translate-based methods, for C2: $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2, \mathbf{e}_2 + \mathbf{r} \approx \mathbf{e}_1$ which means $\mathbf{r} \approx \mathbf{0}$.

3.3 C3: ORC structure with three entities

DEFINITION 4. *C3 is an ORC structure containing three entities* e_1, e_2, e_3. *And* $(e_1, r, e_2) \in C3, (e_2, r, e_3) \in C3$ *and* $(e_1, r, e_3) \in C3$ *or* $(e_3, r, e_1) \in C3$.

For example, there may exist both the following triples: (Lois, $ancestorOf$, Meg) $\in \triangle$, (Carter, $ancestorOf$, Lois) $\in \triangle$, (Meg, $ancestorOf$, Carter) $\in \triangle$ as shown in Fig.1(c). These three triples construct a C3.

C3 has two possible structures. Under the assumption of translate-based methods, for C3: $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2, \mathbf{e}_2 + \mathbf{r} \approx \mathbf{e}_3, \mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_3$ or $\mathbf{e}_3 + \mathbf{r} \approx \mathbf{e}_1$ which means $\mathbf{r} \approx \mathbf{0}$.

There are also many other ORC structures besides the basic form C1, C2, C3 and we will not list them one-by-one. As the analysis before, translate-based methods are unable to encode such ORC structures in knowledge graphs unless make the corresponding relations approach to zero vector. TransE [6] represents every entity as a vector. TransH [29] projects entities on a relation specific hyperplane before computing the distance and TransR [26] project entities from entity space to relation space by a projection matrix while TransD [9] use a dynamic projection matrix. In these models, the representation of an entity as a head entity or tail entity under one relation is the same and this is the main reason why they can't encode ORC structures. To address this problem, we propose to separate the representation of every entity as a head or tail entity to decompose the circles. Fig.2 shows the structures with no circles corresponding to the examples in Fig.1 by representing entities differently. Next we will illustrate how to apply our method to TransH, TransR and TransD.

4. METHODOLOGY

4.1 Our method

In order to decompose the *ORC* circles and adapt to the diversity of structures, we propose to make every entity e has one vector representation $\mathbf{e}$ for itself and two different vector representations, *head representation* $\mathbf{e}_h$ and *tail representation* $\mathbf{e}_t$, with relation r . And $\mathbf{e}_h$ and $\mathbf{e}_t$ can be defined as computation results of other vectors or matrices according to the original definition in different translate-based models. The basic idea is illustrated in Fig. 3.

More specifically, for triple $(e_1, r, e_2) \in \triangle$, we make the head representation of e_1 and the tail representation of e_2 satisfy: $\mathbf{e}_{1h} + \mathbf{r} \approx \mathbf{e}_{2t}$. $\forall (e_1, r, e_2) \in \triangle$, the score function is defined as follows:

$$f_r(e_1, e_2) = ||\mathbf{e}_{1h} + \mathbf{r} - \mathbf{e}_{2t}||_{L_1/L_2}$$

$\mathbf{e} \in \mathbb{R}^d, \mathbf{r} \in \mathbb{R}^d$, d is the dimension of entity vectors and relation vectors. L_1 is L1 regularization and L_2 is L2 regularization. For a true triple, we make the score of score function small and for a false triple, make the score large. This definition can be used for all TransH, TransR and TransD.

4.1.1 *applied to TransH*

As TransH does, we project both the head entity and tail entity to the relation hyperplane before compute the distance between them. We separate the relation hyperplane into *head relation hyperplane* represented by $\mathbf{w}_{rh}$ and *tail relation hyperplane* represented by $\mathbf{w}_{rt}$. And $\mathbf{e}_{1h}$, $\mathbf{e}_{2t}$ are defined as:

$$\mathbf{e}_{1h} = \mathbf{e}_1 - \mathbf{w}_{rh}^\top \mathbf{e}_1 \cdot \mathbf{w}_{rh}, \quad \mathbf{e}_{2t} = \mathbf{e}_2 - \mathbf{w}_{rt}^\top \mathbf{e}_2 \cdot \mathbf{w}_{rt}$$

$\mathbf{w} \in \mathbb{R}^d$. During training process, the following constraints are considered: $\forall (e_1, r, e_2)$, $||\mathbf{e}_1|| \leq 1$, $||\mathbf{e}_2|| \leq 1$, $||\mathbf{r}|| \leq 1$, $||\mathbf{e}_1|| \leq 1$, $||\mathbf{w}_{rh}^\top \mathbf{e}_1 \cdot \mathbf{w}_{rh}|| \leq 1$, $||\mathbf{w}_{rt}^\top \mathbf{e}_2 \cdot \mathbf{w}_{rt}|| \leq 1$.

4.1.2 *applied to TransR*

We define two mapping matrix for every relation as $\mathbf{M}_{rh}$ which projects head entity to *head relation space* and $\mathbf{M}_{rt}$ which projects tail entity to *tail relation space*. And $\mathbf{e}_{1h}$ and $\mathbf{e}_{2t}$ are defined as:

$$\mathbf{e}_{1h} = \mathbf{M}_{rh}\mathbf{e}_1, \quad \mathbf{e}_{2t} = \mathbf{M}_{rt}\mathbf{e}_2$$

$\mathbf{M} \in \mathbb{R}^{d \times d}$. In practice, we also enforce constrains on the norm of the embedding $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{r}$ and the mapping matrices: $\forall (e_1, r, e_2)$, we make $||\mathbf{e}_1|| \leq 1$, $||\mathbf{e}_2|| \leq 1$, $||\mathbf{r}|| \leq 1$, $||\mathbf{M}_{rh}\mathbf{e}_1|| \leq 1$, $||\mathbf{M}_{rt}\mathbf{e}_2|| \leq 1$.

4.1.3 *applied to TransD*

We define two project vectors for every relation r: (i) $\mathbf{r}_h$ to compute the mapping matrix for head entities and (ii) $\mathbf{r}_t$ to compute the mapping matrix for tail entities. Every entity e corresponds to two vectors: $\mathbf{e}$ represents itself meaning and $\mathbf{e}_p$ is used compute the mapping matrix. $\mathbf{e}_{1h}$ and $\mathbf{e}_{2t}$ are defined as:

$$\mathbf{e}_{1h} = \mathbf{M}_{rh}\mathbf{e}_1, \quad \mathbf{e}_{2t} = \mathbf{M}_{rt}\mathbf{e}_2$$

in which:

$$\mathbf{M}_{rh} = \mathbf{r}_h \mathbf{e}_{1p}^\top + \mathbf{I}^{d \times d}, \quad \mathbf{M}_{rt} = \mathbf{r}_t \mathbf{e}_{2p}^\top + \mathbf{I}^{d \times d}$$

$\mathbf{r}_h \in \mathbb{R}^d, \mathbf{r}_t \in \mathbb{R}^d, \mathbf{e}_p \in \mathbb{R}^d$. In experiments, we enforce constrains: $\forall (e_1, r, e_2)$,$||\mathbf{e}_1|| \leq 1$, $||\mathbf{e}_2|| \leq 1$, $||\mathbf{r}|| \leq 1$, $||(\mathbf{r}_h \mathbf{e}_{1p}^\top + \mathbf{I}^{d \times d})\mathbf{e}_1|| \leq 1$, $||(\mathbf{r}_t \mathbf{e}_{2p}^\top + \mathbf{I}^{d \times d})\mathbf{e}_2|| \leq 1$.

4.2 Training

To encourage discrimination between golden triples and incorrect triples, we use the margin-based ranking loss as used in existing translate-based methods:

$$\mathcal{L} = \sum_{(e_1, r, e_2) \in \triangle} \sum_{(e_1', r', e_2') \in \triangle'} [f_r(e_1, e_2) + \gamma - f_{r'}(e_1', e_2')]_+$$

where $[x]_+ = \max(0, x)$, $\triangle$ is the set of correct triples, $\triangle'$ is the set of corrupted triples constructed by replacing

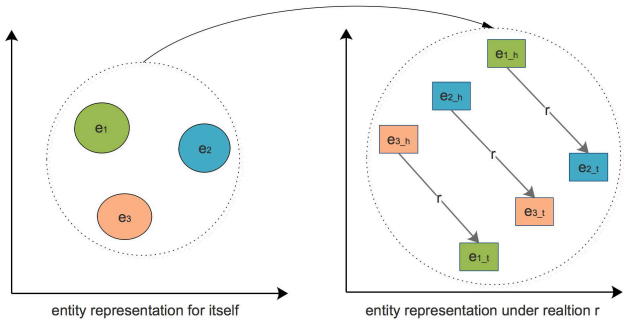

Figure 3: Basic idea of our method

the head or tail entity in correct triples. When corrupting the triple, we also follow [29] and assign different probabilities for head/tail entity replacement to make sure the false-negative instances more likely to be wrong. So we make those 1-to-N relations more chance to replace head entity and N-to-1 relations more chance to replace tail entity. In experiments, we denote the traditional sampling method replacing head/tail entity with same probability as "unif" and the new method in [29] as "bern".

The learning process is carried out using stochastic gradient descent (SGD) [18]. We initialize the entity vectors and relation vectors with the result of TransE when applying our method to TransR and TransD to avoid overfitting.

5. RESULTS

5.1 Dataset and experiment setting

WordNet [14] is a large lexical database of English. Freebase is a large knowledge graph containing facts in the world. We show some details of WN18 and FB15k both released in [6] with "one-relation-circle" matters in Table 1. WN18 is a subdataset of WordNet which containing 18 relations and 151442 triples. FB15k is a subdataset of Freebase which containing 1345 relations and 592213 triples. We list the numbers and percent of the triples consisting at least one of C1, C2 and C3 in the datasets. As we can see, there are about 21% such triples in WN18 and 19% in FB15k. We also notice that there are more C2 structures in WN18 and more C3 structures in FB15k which means the inner graph structures of these two datasets are different. Table 1 supports that ORC structures are common in KGs and it is necessary to enable models to encode these structures. We test our method for link prediction with the benchmark dataset WN18. The details of WN18 are in Table 2.

Table 1: Triple numbers and percents of ORC structures in WN18 and FB15k

	C1	C2	C3
WN18	9(0.00%)	30048(19.84%)	2237(1.48%)
FB15k	2250(0.38%)	52376(8.84%)	59620(10.07%)

5.2 Link prediction

Link Prediction aims to predict the missing entities e_1 or e_2 for a triple (e_1, r, e_2). Namely to predict e_1 when given $(?, r, e_2)$ and predict e_2 when given $(e_1, r, ?)$. Similar to the settings in [6], the task returns a list of candidate entities from the knowledge graph rather than one best answer. In this task, we remove the head or tail entity and replace it with all the entities in the dataset for each triple in test set. Then we compute the score of those corrupted triples and rank them by descending order. The rank of the current test triple is stored finally. Following the previous works we also adopt the evaluation measure mean rank (i.e., mean rank of the test triples) and hit@10 (the proportion of test triples ranked in top-10). It is clear that a good predictor should has lower mean rank and higher hit@10.

Because triples in knowledge graph also will exist in corrupted ones, so we should remove the corrupted triples included in train, valid and test sets before ranking. Follow the existing methods we call this evaluation setting as "Filter" and the original setting as "Raw".

We mark the model using our method with "(dORC)" with the meaning *"decompose the ORC structures"*. We select margin γ among $\{0.5, 0.75, 1, 2, 3, 4, 5\}$, the dimension of entities and relations d among $\{50, 100\}$, the learning rate r among $\{0.001, 0.005, 0.01\}$ and the mini-batch size B among $\{100, 200, 1000, 1400\}$ .The best configurations are: $d = 50, \gamma = 3, r = 0.005, B = 200$ for TransH(dORC)(unif); $d = 50, \gamma = 3, r = 0.005, B = 200$ for TransH(dORC)(bern); $d = 50, \gamma = 5, r = 0.001, B = 1400$ for TransR(dORC)(unif); $d = 50, \gamma = 5, r = 0.001, B = 1400$ for TransR(dORC)(bern); $d = 50, \gamma = 0.75, r = 0.005, B = 200$ for TransD(dORC)(bern). We take L_1 in experiment. The results are shown in Table 3.

Table 2: Details of WN18 in expriment

Dataset	#Rel	#Ent	#Train	#Valid	#Test
WN18	18	40943	141442	5000	5000

Table 3: Results on WN18 for Link Prediction

Method	Mean Rank		Hit@10(%)	
	Raw	Filter	Raw	Filter
Unstructured [1]	315	304	35.3	38.2
RESCAL [13]	1180	1163	37.2	52.8
SE [2]	1011	985	68.5	80.5
SME(linear) [1]	545	533	65.1	74.1
SME(Bilinear) [1]	526	509	54.7	61.3
LFM [17]	469	456	71.4	81.6
TransE [6]	263	251	75.4	89.2
TransH(unif) [29]	318	303	75.4	86.7
TransH(bern) [29]	401	388	73.0	82.3
TransR(bern) [26]	238	225	79.8	92.0
TransR(unif) [26]	232	219	78.3	91.7
CTransR(unif) [26]	243	230	78.9	92.3
CTransR(bern) [26]	231	218	79.4	92.3
TransD(unif) [9]	242	229	79.2	92.5
TransD(bern) [9]	224	212	79.6	92.2
TransH(dORC)(unif)	298	286	79.4	**93.3**
TransH(dORC)(bern)	278	271	80.2	93.0
TransR(dORC)(unif)	224	212	79.3	92.1
TransR(dORC)(bern)	231	219	**80.9**	92.5
TransD(dORC)(bern)	**205**	**192**	79.7	92.4

5.3 Analysis of results

In Table 3, the bold numbers are the best results of all methods and the numbers with underlines are the better results between the original models and corresponding models using our method. For example, mean rank result of TransD(dORC)(bern) is 205 with "Raw" setting, the number is underlined because it is better than TransD(unif) result 224 with "Raw" setting. As we can see, both mean rank and hit@10 results are improved compared to original models in five experiments we do.

6. CONCLUSIONS AND FUTURE WORK

Allowing for potentially interrelating arbitrary entities with each other is an important property of knowledge graphs. This requires embedding models flexible enough to fit the complicated structures caused by the connection between a large amount of entities and relations. In this paper, we focus on a special kind of substructures "one-relation-circle" and propose to separate the representation of entities as head entity and as tail entity to decompose the circles. There are also many more complicated substructures in knowledge graphs we haven't considered. It will be part of our future works to propose models more flexible to encode the diverse structures of knowledge graphs.

The purpose of constructing knowledge graphs is to make machine understand the world more properly. Facts in KGs are a part of human knowledge and other knowledge like rules and commonsense are also important and interesting. In the future, we will pay attention to more kinds of knowledge and try to propose embedding methods for them.

7. ACKNOWLEDGEMENT

This work is funded by NSFC 61473260, NSFC 61673338, national key S&T Special projects 2015ZX03003012-004 and Alibaba-ZJU joint project on "e-Business Knowledge Graph".

8. REFERENCES

[1] J. W. A. Bordes, X. Glorot and Y. Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 2013.

[2] R. C. A. Bordes, J.Weston and Y. Bengio. Learning structured embeddings of knowledge bases. *AAAI*, 2011.

[3] N. U. Alberto GarcÂt'Äśa-DurÂt'an, Antoine Bordes and Y. Grandvalet. Combining two and three-way embedding models for link prediction in knowledge bases. *Journal of Artificial Intelligence Research*, 2016.

[4] B. K. B. S. E. R. H. J. Andrew Carlson, Justin Betteridge and T. M. Mitchell1. Toward an architecture for never-ending language learning. *AAAI*, 2010.

[5] R. C. Antoine Bordes, JasonWeston and Y. Bengio. Learning structured embeddings of knowledge bases. *AAAI*, 2011.

[6] U. N. G.-D. A. W. J. Bordes, Antoine and O. Yakhnenko. Translating embeddings for modeling multi-relational data. *NIPS*, 2013b.

[7] S. B. Denis KrompaÃ§ and V. Tresp. Type-constrained representation learning in knowledge graphs. *ISWC*, 2015.

[8] G. K. F. M. Suchanek and G. Weikum. Yago: a core of semantic knowledge. *WWW*, 2007.

[9] L. X. K. L. Guoliang Ji, Shizhu He and J. Zhao. Knowledge graph embedding via dynamic mapping matrix. *ACL*, 2015.

[10] M. H. Han Xiao and X. Zhu. Ssp: Semantic space projection for knowledge graph embedding with text descriptions. *AAAI*, 2017.

[11] P. P. T. S. K. Bollacker, C. Evans and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. *SIGMOD*, 2008.

[12] L. R. M. Nickel and P. Tomaso. Holographic embeddings of knowledge graphs. *AAAI*, 2016.

[13] V. T. M. Nickel and H.-P. Kriegel. A three-way model for collective learning on multi- relational data. *ICML*, 2011.

[14] B. R. F. C. G. D. M. K. Miller, G.A. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 1990.

[15] G. A. Miller. Wordnet: A lexical database for english. *Communications of the ACM Vol. 38*, 1995.

[16] F. P. Ni Lao, Amarnag Subramanya and W. Cohen. Reading the web with learned syntactic-semantic inference rules. *EMNLP*, 2012.

[17] A. B. G. O. R. Jenatton, N. Le Roux. A latent factor model for highly multi-relational data. *AISTATS*, 2010.

[18] H. Robbins and Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 1951.

[19] Z. L. Ruobing Xie and M. Sun. Representation learning of knowledge graphs with hierarchical types. *IJCAI*, 2016.

[20] B. Shi and T. Weninger. Proje:embedding projection for knowledge graph completion. *AAAI*, 2017.

[21] G. J. Shizhu He, Kang Liu and J. Zhao. Learning to represent knowledge graphs with gaussian embedding. *CIKM*, 2015.

[22] C. D. M. C. D. Socher, Richard and A. Y. Ng. Reasoning with neural tensor networks for knowledge base completion. *NIPS*, 2013.

[23] M. Xiao, H.; Huang and X. Zhu. Transg : A generative model for knowledge graph embedding. *ACL*, 2016.

[24] Z. J. J. L. H. Xie, R.; Liu and M. Sun. Representation learning of knowledge graphs with entity descriptions. *AAAI*, 2016.

[25] H. L. X. J. Y. Jia, Y. Wang and X. Cheng. Locally adaptive translation for knowledge graph embedding. *AAAI*, 2016.

[26] M. S. Y. L. Y. Lin, Z. Liu and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. *AAAI*, 2015.

[27] Z. L. Y. Lin and M. Sun. Modeling relation paths for representation learning of knowledge bases. *EMNLP*, 2015.

[28] M. S. Yankai Lin, Zhiyuan Liu. Knowledge representation learning with entities, attributes and relations. *IJCAI*, 2016.

[29] J. F. Z. Wang, J. Zhang and Z. Chen. Knowledge graph embedding by translating on hyperplanes. *AAAI*, 2014.