Knowledge Graph Embedding Based on Adaptive Negative Sampling

Saige Qin, Guanjun Rao(✉), Chenzhong Bin, Liang Chang, Tianlong Gu, and Wen Xuan

Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Tochnology, Guilin 541004, China
rao-rgj@foxmail.com

Abstract. Knowledge graph embedding aims at embedding entities and relations in a knowledge graph into a continuous, dense, low-dimensional and real-valued vector space. Among various embedding models appeared in recent years, translation-based models such as TransE, TransH and TransR achieve state-of-the-art performance. However, in these models, negative triples used for training phase are generated by replacing each positive entity in positive triples with negative entities from the entity set with the same probability; as a result, a large number of invalid negative triples will be generated and used in the training process. In this paper, a method named adaptive negative sampling (ANS) is proposed to generate valid negative triples. In this method, it first divided all the entities into a number of groups which consist of similar entities by some clustering algorithms such as K-Means. Then, corresponding to each positive triple, the head entity was replaced by a negative entity from the cluster in which the head entity was located and the tail entity was replaced in a similar approach. As a result, it generated a set of high-quality negative triples which benefit for improving the effectiveness of embedding models. The ANS method was combined with the TransE model and the resulted model was named as TransE-ANS. Experimental results show that TransE-ANS achieves significant improvement in the link prediction task.

Keywords: Adaptive negative sampling · Knowledge graph embedding · Translation-based model

1 Introduction

Knowledge graph is a directed graph composing of different types of entities as nodes and relations between entities as edges. Essentially, knowledge graph is a semantic network that expresses the semantic relations between various types of entities. Its basic constituent unit is a triple (h, r, t), where h is a head entity, t is a tail entity, and r represents the relationship between head and tail. Recently, the knowledge graph has played a vital role in data mining, artificial intelligence and other fields, and it promotes the development of artificial intelligence applications, such as web search, and question answering.

With the advent of big data era, the scale of knowledge graph has grown rapidly, and various large-scale knowledge graphs (such as Freebase, WordNet, and NELL) have appeared. Although the scale of knowledge graph is very large, but it is still incomplete. So it is necessary to complete the current knowledge graphs. This is one of the current hot topics in knowledge graph. Recently, it is very popular to learn methods of learning vector representations of entities and relations in the knowledge graph, where embedding-based representation learning method show strong feasibility and robustness. These methods are to embed the entities and relations in knowledge graph into a continuous, dense, low-dimensional, and real-valued vector space.

Among various representational learning methods, translation-based presentation learning model achieves state-of-the-art performance. TransE is the most typical translation-based model which proposed by Bordes et al. in 2013. TransE treats relations as translations from head entity to tail entity in vector space. If a triple (h, r, t) holds, head entity vector **h**, tail entity vector **t** and relation vector **r** should satisfy $\mathbf{h} + \mathbf{r} = \mathbf{t}$. TransE is extremely simple and shows excellent performance in processing large-scale data, thus set off a research boom in translation-based presentation learning.

TransH model proposes that an entity has different representations in different relations, projects entity onto a hyperplane where the relation lies, and then performs translation operations on hyperplane. Lin et al. believe that entities and relations should be in different semantic spaces, and propose TransR model. TransR projects entities from the entity space into the relation space through a projection matrix, and then establishes translation operations in the relational space. Ji et al. proposed the TranSparse model considering imbalance and heterogeneity of entities and relations. TranSparse solves this problem by using a sparse matrix instead of a dense matrix in TransR. FT model considers that translation principle of $\mathbf{h} + \mathbf{r} = \mathbf{t}$ is too strict, thus establishing a more flexible translation principle $\mathbf{h} + \mathbf{r} = \alpha\mathbf{t}$, which improves the expression ability of the translation model. DT model considers that translation principle of FT model are still too complex, and further proposes dynamic translation principle $(\mathbf{h} + \alpha_h) + (\mathbf{r} + \alpha_r) = (\mathbf{t} + \alpha_t)$, which improves the performance of the translation model.

Although the translation-based representation learning model has been improved steadily, this aspect of research still faces a common challenge. In the representation learning model, knowledge graph used for training only has valid positive knowledge (positive triples), and there is no valid negative knowledge (negative triples), which proposes a great challenge for representation learning model training. In order to get a negative triple, the existing model usually deletes the head entity (or tail entity) in the positive triple and selects one entity randomly to replace head entity (or tail entity) in the entity set. Unfortunately, this method is not ideal, because of the large number of entity sets, the number of negative triples is even larger corresponding to each positive triple. In the negative triples, most of the negative triples are very different from the positive triples, making them extremely easy to distinguish. Thus, they are an invalid negative triple (e.g., negative triples (UnitedStates, President, NewYork)). If we just randomly replace an entity with equal-probability to generate a negative triple, that usually result in the generation is easy to distinguish negative in most cases. In other words, the large number of negative triples generated are not similar to the positive triples and such negative triples do not help model training.

In this paper, we believe that generating an effective negative triple is very important for training the model, and the current method of random substitution with equal-probability is not reasonable. To that end, we propose an adaptive negative sampling method to generate a valid negative triple. We integrate adaptive negative sampling method into TransE, named TransE-ANS, and perform extensive experiments on four common datasets (i.e., WN18, WN11, FB15k and FB13), both of them obtain superior state-of-the-art performance.

The rest of this paper is organized as follows. We introduce related work in Sect. 2. We detail a limitation of equal-probability sampling and a method of adaptive negative sampling in Sect. 3. We evaluate our method through experiments in Sect. 4. Conclusions are summarized in Sect. 5.

2 Related Work

We divided the previous studies into two categories: one is a translation-based model, and the other is an embedded model.

2.1 Translation-Based Models

TransE [1] treats r as the translation operation from h to t in triple (h, r, t). TransE believes that each triple in knowledge graph should satisfy $\mathbf{h} + \mathbf{r} = \mathbf{t}$. In other words, when (h, r, t) is a positive triple, ($\mathbf{h} + \mathbf{r}$) should be close to ($\mathbf{t}$); otherwise ($\mathbf{h} + \mathbf{r}$) will go away from ($\mathbf{t}$). Therefore, score function for embedding of a triple is $f_r(h,t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. TransE is a simple and effective model. Especially, when dealing with 1-to-1 simple relations, the performance of the model is particularly outstanding, but there are some defects in dealing with complex relations of 1-to-N, N-to-1 and N-to-N. For example, there is a triple set (h_1, r, t), (h_2, r, t), … (h_n, r, t). Embedding by TransE, we might get $h_1 = h_2 = \ldots = h_n$.

Wang et al. [2] believe that each entity should have different representations in different relations. TransH projects head entity vector and tail entity vector to a relation hyperplane by normal vector ω_r, and correspondingly obtains the projected vectors $\mathbf{h}_\perp$ and $\mathbf{t}_\perp$. Its score function is $f_r(h,t) = \|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2$.

TransR/CtransR [3] proposes to embed entities and relations into different semantic spaces respectively and constructs a projection matrix M_r for each relation r. Then, it uses M_r to project the entity vector into the relation space. Its score function is $f_r(h,t) = \|\mathbf{h}\mathrm{M}_r + \mathbf{r} - \mathbf{t}\mathrm{M}_r\|_2^2$.

TranSparse [4] mainly considers the heterogeneity and imbalance of entities and relations in knowledge graph to prevent overfitting of simple relations and underfitting of complex relations. TranSparse constructs sparse matrices $\mathrm{M}_r^h(\theta_r^h)$ and $\mathrm{M}_r^t(\theta_r^t)$ for head entities and tail entities adaption based on the complexity of each relation. Its score function is $f_r(h,t) = \left\|\mathrm{M}_r^h(\theta_r^h)\mathbf{h} + \mathbf{r} - \mathrm{M}_r^t(\theta_r^t)\mathbf{t}\right\|_2^2$.

FT [5] thought that the translation principles of the existing models $\mathbf{h} + \mathbf{r} = \mathbf{t}$ are too strict to express the complex diversity of entities and relations. Therefore, a novel and more flexible translation principle $\mathbf{h} + \mathbf{r} = \alpha\mathbf{t}$ is proposed. The FT can flexibly model complex and various entities and relations, and easily extend to a series of translation models such as TransE. FT uses the following score function $f_r(h,t) = (\mathbf{h}+\mathbf{r})^{\mathrm{T}}\mathrm{t} + \mathbf{h}^{\mathrm{T}}(\mathbf{t}+\mathbf{r})$.

Chang et al. [6] believed that FT model was still too strict in changing entities in a certain direction, and put forward a representation learning model based on the principle of dynamic translation. DT defined a new kind of translation rules $(\mathbf{h}+\alpha_h)+(\mathbf{r}+\alpha_r) = (\mathbf{t}+\alpha_t)$, relax the constraints of entities and relations. Like the FT model, the DT model is also easy to extend to a series of translation models such as TransE.

2.2 Other Models

SE [7] constructs two relation-specific matrices M_{rh} and M_{rt} respectively for head entities and tail entities. Its score function is $f_r(h,t) = \|\mathrm{M}_{rh}\mathbf{h} - \mathrm{M}_{rt}\mathbf{t}\|_1$.

LFM [8, 9] encodes each entity and sets a matrix M_r for each relation. It defines the following bilinear scoring function for each triple $f_r(h,t) = \mathbf{h}^{\mathrm{T}}\mathrm{M}_r\mathbf{t}$.

SME [10, 11] constructs four different projection matrices to capture semantic associations between entities and relations, and for each triples defines a linear score function $f_r(h,t) = (\mathrm{M}_h\mathbf{h} + \mathrm{M}_{r1}\mathbf{r} + \mathrm{b}_1)^{\mathrm{T}}(\mathrm{M}_t\mathbf{t} + \mathrm{M}_{r2}\mathbf{r} + \mathrm{b}_2)$, and a bilinear score function $f_r(h,t) = (\mathrm{M}_h\mathbf{h} \otimes \mathrm{M}_{r1}\mathbf{r} + \mathrm{b}_1)^{\mathrm{T}}(\mathrm{M}_t\mathbf{t} \otimes \mathrm{M}_{r2}\mathbf{r} + \mathrm{b}_2)$.

NTN [12] uses bilinear tensors to replace the linear transformation layer of traditional neural networks, and connects head entities and tail entities vectors in different dimensions.

3 Our Method

In this section, we first demonstrate the impact of generating a negative triple by using the same probability to replace h (or t) in a positive triple on knowledge graph embedding models. Then we propose an adaptive negative sampling principle and combine it with the typical model TransE.

3.1 The Limitations of Simple Random Sampling

Entity equal-probability sampling is used for generating negative triples in most of existing knowledge graph embedding models, that is, an entity was extracted from entity set with the same probability to replace h (or t) in positive triples. We refer to the extracted entity as a negative entity and the replaced entity as a positive entity. However, this method may suffer from a problem in generating negative triples: Negative entities are not similar to positive entities, which results in an invalid negative triple. Next, we will describe this issue in detail.

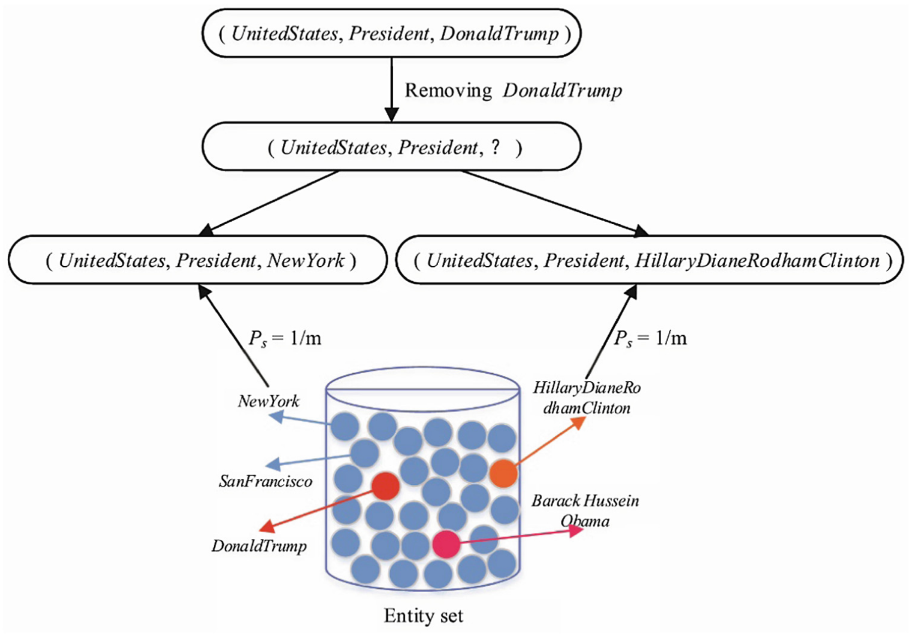
(UnitedStates, President, DonaldTrump)
Removing DonaldTrump
(UnitedStates, President, ?)
(UnitedStates, President, NewYork)
(UnitedStates, President, HillaryDianeRodhamClinton)
$P_s = 1/m$
$P_s = 1/m$
NewYork
SanFrancisco
DonaldTrump
HillaryDianeRodhamClinton
Barack Hussein Obama
Entity set

Fig. 1. The Simple Random Sampling. The circle represent entities, and the closer colors of two circles are, the more similar they are; P_s is sampling probability; m is the number of entities.

A certain similarity exists between each entity. Given a positive entity, it has multiple negative entities, and the similarity between positive entity and each negative entity is different. A negative entity extracted by simple random sampling may have a very low similarity with the corresponding positive entity, so an invalid negative triple may be generated. Invalid negative triples are little help to the effective embedding of learning knowledge graph. In the following paragraph, we use a concrete example to illustrate the importance of valid negative triple in knowledge graph embedding learning.

As shown in Fig. 1, supposing there is a positive triple (UnitedStates, President, Trump) existing in the knowledge graph. According to simple random sampling, a negative triple is generated by replacing the tail. Specifically, we get an incomplete triple (UnitedStates, President, ?) by removing Trump. Then, we extract a tail entity from entity set with simple random sampling. Assuming that a City type entity NewYork is extracted, we will get a ridiculous negative triple (UnitedStates, President, NewYork), which is an invalid negative triple, as the relation President requires a tail to be a Person type. Therefore, the generated triple is an invalid tail for the relation President by a simple type constraint judgment, which means the negative triple is invalid. In contrast, if a Person type entity Hillary is extracted, we will get a very valid negative triple (UnitedStates, President, Hillary). For anyone who does not have a detailed understanding of the United States president history, he cannot judge whether

Hillary is President of the United States because she is very similar to Trump in many ways. Such a valid negative entity can generate a valid negative triple, which has a positive effect on knowledge graph embedding.

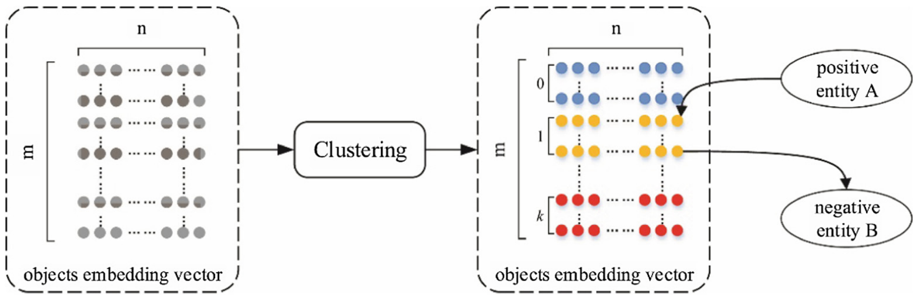

Fig. 2. The Adaptive Negative Sampling framework. m and n represent the number of entity vectors and the dimensions of the vector, respectively. Entity vectors are clustered to obtain k clusters, each cluster containing several entity vectors. When the positive entity A belongs to cluster 1, the negative entity B will come from cluster 1.

In the training of knowledge graph embedding model, we hope to obtain a negative entity that is similar to positive entity. However, this issue has seriously affected knowledge graph embedding. Therefore, we propose Adaptive Negative Sampling to solve this problem.

3.2 Adaptive Negative Sampling

A valid negative triple can help the knowledge graph embedding, so we need to get a valid negative triple. The key of obtaining a valid negative triple is extracting a negative entity similar to a positive entity. In vector space, we can determine whether they are similar by calculating the distance between two entity vectors. If the distance between two entity vectors is smaller, the more similar they are and vice versa. This motivates us the idea of clustering entities and then looking for similar entities in a cluster. Therefore, we decided to use simple and effective K-Means algorithm [13–15] to cluster entities, so that positive and negative entities come from the same cluster. We refer to the above sampling process of negative entities as Adaptive Negative Sampling (ANS). The Adaptive Negative Sampling framework is shown in Fig. 2. The followed is the K-Means clustering of entities in knowledge graph.

Given a knowledge graph $G = (E, R, S)$, where $E = \{e_1, e_2, \ldots, e_N\}$ denotes a set of entities consisting N entities in knowledge graph, $R = \{r_1, r_2, \ldots, r_M\}$ indicates a set of relations consisting M relations in knowledge graph, and $S \subseteq E \times R \times E$ represents a set of triples in knowledge graph. We aim to divide N entities into K clusters by K-Means in which each entity belongs to a cluster with the nearest mean, i.e., the sum of Euclidian distances from each entity to its cluster center is the smallest. Our target formula is as follows:

$$\arg min \sum\nolimits_{i=1}^{K} \sum\nolimits_{e \in \mathrm{C}_i} \|\mathbf{e} - \mathrm{c}_i\|_{l1} \tag{1}$$

where K denotes the number of clusters, **e** is an entity vector, c_i represents i^{th} cluster center vector, C_i is a set of entity e in i^{th} cluster. Entity set E is divided into K clusters $E_1, E_2, \ldots, E_K$ by K-Means clustering. There is a high similarity between entities in the same cluster. Given a positive entity $e \in E_k (k \in 1, 2, \ldots, K)$, a negative entity e' will be selected from the cluster E_k. Then, the obtained negative entity e' and the positive entity e will have a high similarity. Therefore, we can get a valid negative triple to help knowledge graph embedding.

Algorithm 1 Learning TransE-ANS

```
Require: Training sets positive triples $S = \{(h, r, t)\}$ and negative triples $S' = \{(h', r, t) | h' \in E_h\} \cup \{(h, r, t') | t' \in E_t\}$, entity set $E$, relation set $R$, margin $\gamma$, embedding dimension $n$, learning rate $\alpha$, K-Means cluster number $k$, entity cluster $E_i$ (i = 1, 2, …, k)
Ensure: Entity and relation embeddings.
1:  initialize:
2: r ←uniform $(-\frac{6}{\sqrt{n}}, \frac{6}{\sqrt{n}})$ for each $r \in R$
3: r ←$r/\|r\|$ for each relation $r \in R$
4: e ←uniform $(-\frac{6}{\sqrt{n}}, \frac{6}{\sqrt{n}})$ for each $e \in E$
5:  e ←$e/\|e\|$ for each relation $e \in E$
6:  loop:
7:  $S_{batch} \leftarrow$ sample $(S, b)$ // sample a minibatch of size $b$
8:  $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets
9:  for $(h, r, t) \in S_{batch}$ do
10:     $(h', r, t') \leftarrow$ sample$(S'_{(h, r, t)})$ //sample a corrupted triplet
11:     $T_{batch} \leftarrow T_{batch} \cup \{(h, r, t), (h', r, t')\}$
12:  end for
13:  Update embedding w.r.t.
     $\sum_{((h,r,t),(h',r,t')) \in T_{batch}} \nabla[f_r(h,t) + \gamma - f_r(h',t')]_+$
14:   if epoch % 50 == 0 then
15:       Update $E_i$, // K-Means clustering
16:  end if
17:  end loop
```

3.3 TransE-ANS

In this section, we discuss the TransE-ANS model constructed by combining ANS with TransE model. TransE-ANS embeds entities and relations into the same vector space using the translation principles $\mathbf{h} + \mathbf{r} = \mathbf{t}$ of TransE. Therefore, scoring function of TransE-ANS is as follows:

$$f_r(h,t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{l1} \tag{2}$$

In TransE-ANS, we use a marginal loss function as our training target. The marginal loss function is as follows:

$$L = \sum\nolimits_{(h,r,t)\in S} \sum\nolimits_{(h',r,t')\in S'} [f_r(h,t) + \gamma - f_r(h',t')]_+ \tag{3}$$

where S is a set of positive triples, $S' = \{(h',r,t)|h' \in E_h\} \cup (h,r,t')|t' \in E_t\}$, E_h is the cluster where h is located, E_t is the cluster where t is located.) is a set of negative triples, $[f_r(h,t) + \gamma - f_r(h',t')]_+ = \max(0, [f_r(h,t) + \gamma - f_r(h',t')])$, γ is a margin. We use the stochastic gradient descent (SGD) [16] to minimize the marginal loss function.

Algorithm 1 shows the complete training process of our model. For every 50 training epochs, we cluster the entity vectors we get. In 15th line of the algorithm, assuming that we use the K-Means clustering algorithm, but obviously which can substitute it with any clustering algorithm.

3.4 Comparison of Complexity

The complexity of several knowledge graph embedding models is shown in Table 1. In general, the number of cluster centers is much smaller than entities, so the parameter complexity of TransE-ANS is same as TransE. In terms of time complexity, comparing with TransE model, the TransE-ANS model only increases the time of each K-Means clustering. Therefore, our approach has high efficiency.

Table 1. The complexity of several knowledge graphs embedded in the model.

Model	Parameters	Operations (Time)
SE	$O(N_e m + 2N_r n^2)(m = n)$	$O(2m^2 N_t)$
SME(linear)	$O(N_e m + N_r n + 4mk + 4k)(m = n)$	$O(4mkN_t)$
SME(bilinear)	$O(N_e m + N_r n + 4mks + 4k)(m = n)$	$O(4mksN_t)$
LFM	$O(N_e m + N_r n^2)(m = n)$	$O((m^2 + m)N_t)$
SLM	$O(N_e m + N_r(2k + 2nk))(m = n)$	$O((2mk + k)N_t)$
TransE	$O(N_e m + N_r n)(m = n)$	$O(N_t)$
TransH	$O(N_e m + 2N_r n)(m = n)$	$O(2mN_t)$
TransR	$O(N_e m + N_r(m + 1)n)$	$O(2mnN_t)$
CTransR	$O(N_e m + N_r(m + d)n)$	$O(2mnN_t)$
TransD	$O(2N_e m + 2N_r n)(m = n)$	$O(2nN_t)$
GTrans-SW	$O(N_e m + 3N_r n)(m = n)$	$O(nN_t)$
TransE-ANS	$O((N_e + K)m + N_r n)(m = n)$	$O(N_t + KiN_e n)$

4 Experiments and Analysis

Table 1 shows the complexity of various embedded models. In Table 1, N_e is the number of entities and N_r is the number of relations. N_t is the number of triples in the knowledge graph. m and n represent the dimension of the entity embedding space and relations embedding space, respectively. k represents the number of hidden nodes of the neural network, s denotes the number of slices of the tensor, and K denotes the number of K-Means cluster centers, i denotes the number of cluster iterations. We use link prediction and triple classification tasks to evaluate our approach. For reducing the implementation time of model, our code uses multiple threads in Linux system to achieve rapid training and testing.

4.1 Data Setting

The datasets that we use are from two widely used knowledge graphs: WordNet and Freebase. WordNet is a large knowledge graph of English vocabulary. Freebase is a large-scale knowledge graph of human knowledge that stores general facts of the world. In this experiments, we use four subsets from WordNet and Freebase, i.e., WN18 and FB15K. Table 2 shows these two datasets statistics.

Table 2. Dataset information.

Dataset	#Ent	#Rel	#Train	#Vaild	#Test
WN18	40943	18	141442	5000	5000
FB15K	14951	1345	483142	50000	59071

Table 3. Optimal parameter setting in link prediction.

Dataset	Metric	Epoch	λ	γ	n	B	K	i	D.S
WN18	Mean Rank	2000	0.001	5.5	50	100	14	20	l_1
	Hit@10	2000	0.001	3	50	100	14	20	l_1
FB15K	Mean Rank	2000	0.001	4	200	200	64	20	l_1
	Hit@10	2000	0.001	2	200	200	64	20	l_1

4.2 Link Prediction

Link prediction aims to predict a missing entity h (or t) in a triple (h, r, t) [1, 8, 12]. We call entity h (or t) missing in a test triple (h, r, t) as the correct entity, and all the entity other than the correct entity are considered as the candidate entity. Firstly, we use each candidate entity to replace h (or t) of a test triple (h, r, t) to get the candidate triple. Then, the score of the test triple and each candidate triple are calculated. Finally, the correct entity and the candidate entity are sorted in ascending order according to the score. We use the two indicators [1] as our evaluation metrics: (1) the average rank of the correct entity (Mean Rank); and (2) the proportion of the correct entity ranked in top 10 (Hits@10).

It is worth nothing that the candidate triples may exist in knowledge graph, and these candidate triples should be considered as the correct triples. Its scores are likely to be lower than the correct triples. Thus, we should filter out these candidate triples that have already appeared in train, valid and test sets. We set the filter during the test to filter out these candidate triples, and we use evaluation setting by "Filt", otherwise we use evaluation setting by "Raw".

In this task, in order to get the influence of each parameter on our model, we try various parameter settings. We select the training period epoch among {1000, 1500, 2000}, the learning rate λ among {0.01, 0.001, 0.0001}, the margin value γ among {1, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6}, the dimension n among {25, 50, 100, 200}, the nimi-batch size B among {100, 200, 500, 1000}, the number of cluster center K among {14, 32, 64}, the number of clustering iteration i among {10, 20, 50}, and the dissimilarity D.S is l1-norm. We obtained two different optimal parameter settings for Mean Rank and Hits@10 on the two datasets, as shown in Table 3. We obtained the experimental results of Mean Rank and Hits@10 by these two settings.

Table 4. Link prediction results on WN18.

Metric	Mean Rank		Hits@10	
	Raw	Filt	Raw	Filt
SE	1011	985	68.5	80.5
SME(linear/bilinear)	542/526	533/509	65.1/54.7	74.1/61.3
LFM	469	456	71.4	81.6
TransE	263	251	75.4	89.2
TransH(unif/bern)	318/401	303/388	75.4/73.0	86.7/82.3
TransR(unif/bern)	232/238	219/225	78.3/79.8	91.7/92.0
CTransR(unif/bern)	243/231	230/218	78.9/79.4	92.3/92.3
TransD(unif/bern)	242/224	229/212	79.2/79.6	92.5/92.2
TransSparse(unif/bern)	233/223	221/211	79.6/80.1	93.4/93.2
TransSparse-DT(unif/bern)	248/234	232/221	80.0/**81.4**	93.6/94.3
TransE-ANS(unif/bern)	**220/207**	**208/195**	**80.2**/80.6	**94.0/94.6**

The experimental results of WN18 and FB15K link prediction tasks are shown in Tables 4 and 5, respectively. We compare TransE-SNS with a few state-of-the-art methods in the link prediction task on WN18 and FB15K, including SE, SME, LFM, TransE, TransH, TransR/CTransR, TransD, TranSparse and TranSparse-DT. We use the results reported in their papers or [3] directly since the data set is the same.

Table 5. Link prediction results on FB15K.

Metric	Mean Rank		Hits@10	
	Raw	Filt	Raw	Filt
SE	273	162	28.8	39.8
SME(linear/bilinear)	274/284	154/158	30.7/31.3	40.8/41.3
LFM	283	164	26.0	33.1
TransE	243	125	34.9	47.1
TransH(unif/bern)	211/212	84/87	42.5/45.7	58.5/64.4
TransR(unif/bern)	226/198	78/77	43.8/48.2	65.5/68.7
CTransR(unif/bern)	233/199	82/**75**	44.0/48.4	66.3/70.2
TransD(unif/bern)	211/194	67/91	49.4/53.4	74.2/77.3
TransSparse(unif/bern)	216/190	66/82	50.3/53.7	78.4/79.9
TransSparse-DT(unif/bern)	208/**188**	58/79	**51.2/53.9**	78.4/80.2
TransE-ANS(unif/bern)	**198**/210	**56**/95	48.9/52.5	**80.1/83.0**

As shown in Table 4, TransE-ANS performance has been greatly improved compared to TransE. Although, TransE-ANS slightly behind TranSparse-DT in Hits@10 (raw, bern), TransE-ANS still achieved state-of-the-art performance in most cases. As shown in Table 5, TransE-ANS performance has still been greatly improved in FB15K compared with TransE. Regrettably, our method failed to achieve state-of-the-art performance in both Mean Rank(bern) and Hits@10(raw). We believe that there are two reasons for this result: One reason is that the data of FB15K is sparse, and there are fewer entities with Multiple identical relationships (i.e. each entity has fewer similar entities), which results in a certain number of entities with lower similarity in each cluster after clustering. Another reason is that cluster centers K is difficult to determine, and the choice of our K value has some limitations. Therefore, K-Means clustering does not classify entities well.

Tables 6 and 7 are link prediction results on FB15K by relation category. The relation categories include 1-to-1, 1-to-N, N-to-1, and N-to-N, which account for 24%, 23%, 29% and 24% of the total number of relations, respectively. Table 6 shows results of predicting head entities in different relation categories. Table 7 shows results of predicting tail entities in different relation categories.

As can be seen from Table 6, Among different categories of relations, our method always finds the best result in a certain situation. Particularly, in the N-to-N relationship, our approach achieves the most state-of-the-art performance. As shown in Table 7, we can get similar conclusions as in Table 6, and achieve state-of-the-art performance in more cases in Table 7. Overall, the performance of our approach is far superior to the baseline model TransE, and in most cases achieves the most advanced performance over all current models.

Table 6. Prediction results of head on FB15K by relation category (%).

Tasks	Predicting head (Hit@10)			
Relation category	1-to-1	1-to-N	N-to-1	N-to-N
SE	35.6	62.6	17.2	37.5
SME	35.1/30.9	53.7/69.6	19.0/19.9	40.8/38.6
TransE	43.7	65.7	18.2	47.2
TransH(unif/bern)	66.7/66.8	81.7/87.6	30.2/28.7	57.4/64.5
TransR(unif/bern)	76.9/78.8	77.9/89.2	38.1/34.1	66.9/69.2
CTransR(unif/bern)	78.6/81.5	77.8/89.0	36.4/34.7	68.0/71.2
TransD(unif/bern)	80.7/86.1	85.8/95.5	47.1/39.8	75.6/78.5
TransSparse(unif/bern)	83.2/87.1	85.2/**95.8**	51.8/44.4	80.3/81.2
TransSparse-DT(unif/bern)	83.0/**87.4**	85.7/**95.8**	**51.9**/47.7	80.5/81.6
TransE-ANS(unif/bern)	**83.4**/84.1	**88.8/95.8**	45.6/**48.4**	**83.2/85.3**

Table 7. Prediction results of tail on FB15K by relation category (%).

Tasks	Predicting tail (Hit@10)			
Relation category	1-to-1	1-to-N	N-to-1	N-to-N
SE	34.9	14.6	68.3	41.3
SME	32.7/28.2	14.9/13.1	61.6/76.0	40.8/38.6
TransE	43.7	19.7	66.7	50.0
TransH(unif/bern)	63.7/65.5	30.1/39.8	83.2/83.3	60.8/67.2
TransR(unif/bern)	76.2/79.2	38.4/37.4	76.2/90.4	69.1/72.1
CTransR(unif/bern)	77.4/80.8	37.8/38.6	78.0/90.1	70.3/73.8
TransD(unif/bern)	80.0/85.4	54.5/50.6	80.7/94.4	77.9/81.2
TransSparse(unif/bern)	82.6/87.5	60.0/57.0	**85.5**/94.5	82.5/83.7
TransSparse-DT(unif/bern)	82.8/86.7	59.9/56.3	**85.5/94.8**	82.9/84.0
TransE-ANS(unif/bern)	**87.4/88.5**	**60.8/60.5**	83.3/94.5	**83.3/85.7**

5 Conclusion

In this paper, we introduce a method named Adaptive Negative Sampling that generates valid negative triples to help knowledge graph embedding. It considers the limitations of simple random sampling. We construct TransE-ANS model by combining ANS with TransE model. Experimental results show that TransE-ANS performs better than other models in link prediction task, since it can generate a set of high-quality negative triples which are benefit for improving the effectiveness of embedding models. We will further explore the combination of different clustering algorithms and knowledge graph embedding models in the future.

Acknowledgements. This work was partially supported by the National Natural Science Foundation of China (Nos. U1501252, 61572146 and U1711263), the Natural Science Foundation of Guangxi Province (No. 2016GXNSFDA380006), the Guangxi Innovation-Driven Development Project (No. AA17202024) and the Guangxi Universities Young and Middle-aged Teacher Basic Ability Enhancement Project (No. 2018KY0203).

References

1. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedings of NIPS, Lake Tahoe, NV, USA, pp. 2787–2795 (2013)
2. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of AAAI, Quebéc city, QC, Canada, pp. 1112–1119 (2014)
3. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of AAAI, Austin, TX, USA, pp. 2181–2187 (2015)
4. Ji, G., Liu, K., He, S., Zhao, J.: Knowledge graph completion with adaptive sparse transfer matrix. In: Proceedings of AAAI, Phienix, AZ, USA, pp. 985–991 (2016)
5. Feng, J., Huang, M., Wang, M., Zhou, M., Hao, Y., Zhu, X.: Knowledge graph embedding by flexible translation. In: Proceedings of KR, Cape Town, South Africa, pp. 557–560 (2016)
6. Chang, L., Zhu, M., Gu, T., Bin, C., Qian, J., Zhang, J.: Knowledge graph embedding by dynamic translation. IEEE Access **5**, 20898–20907 (2017)
7. Bordes, A., Weston, J., Collobert, R., Bengio, Y.: Learning structured embeddings of knowledge bases. In: Proceedings of AAAI, San Francisco, CA, USA, pp. 301–306 (2011)
8. Jenatton, R., Roux, N.L., Bordes, A., Obozinski, G.: A latent factor model for highly multi-relational data. In: Proceedings of NIPS, Lake Tahoe, NV, USA, pp. 3167–3175 (2012)
9. Sutskever, I., Salakhutdinov, R., Tenenbaum, J.B.: Modelling relational data using Bayesian clustered tensor factorization. In: Proceedings of NIPS, Vancouver, BC, Canada, pp. 1821–1828 (2009)
10. Bordes, A., Glorot, X., Weston, J., Bengio, Y.: A semantic matching energy function for learning with multi-relational data. Mach. Learn. **94**(2), 233–259 (2014)
11. Bordes, A., Glorot, X., Weston, J., Bengio, Y.: Joint learning of words and meaning representations for open-text semantic parsing. In: Proceedings of AISTATS, La Palma, Canary Islands, pp. 127–135 (2012)
12. Socher, R., Chen, D., Manning, C.D., Ng, A.Y.: Reasoning with neural tensor networks for knowledge base completion. In: Proceedings of NIPS, Lake Tahoe, NV, USA, pp. 926–934 (2013)
13. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: a K-Means clustering algorithm. J. R. Stat. Soc. **28**(1), 100–108 (1979)
14. Hamerly, G., Elkan, C.: Algorithm AS 136: a K-Means clustering algorithm. In: Proceedings of CIKM, McLean, VA, USA, pp. 600–607 (2002)
15. Celebi, M.E., Kingravi, H.A., Vela, P.A.: A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Syst. Appl. **40**(1), 200–210 (2013)
16. Duchi, J.C., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. JMLR **12**, 2121–2159 (2011)