World Wide Web (2023) 26:2887–2907
https://doi.org/10.1007/s11280-023-01167-x

Correlation embedding learning with dynamic semantic enhanced sampling for knowledge graph completion

Haojie Nie[1] · Xiangguo Zhao[2] · Xin Bi[3] · Yuliang Ma[4] · George Y. Yuan[5]

Received: 19 October 2022 / Revised: 21 February 2023 / Accepted: 12 March 2023 /
Published online: 19 May 2023

Abstract
Knowledge graph completion aims to solve the problem of incompleteness and sparsity in knowledge graphs. However, the negative sampling strategy in current completion methods samples entities with equal probability and cannot guarantee the quality of negative samples, causing gradient disappearance during the training process. In addition, existing embedding learning methods ignore the diversity and complexity of the entities and relations in a knowledge graph. Therefore, we design a dynamic semantic sampling and correlation embedding completion framework that includes a negative sampling algorithm based on dynamic semantic similarity and a correlation embedding model. The negative-sampling algorithm can gradually explore high-quality negative samples to participate in model training. The embedding model can enrich embeddings by learning the sequential and correlated information of entities and relations in the knowledge graph. Finally, we conducted experiments on two public datasets, and the experimental results proved the performance of our method.

This article belongs to the Topical Collection: *Special Issue on Knowledge-Graph-Enabled Methods and Applications for the Future Web*
Guest Editors: Xin Wang, Jeff Pan, Qingpeng Zhang, and Yuan-Fang Li.

Haojie Nie contributed equally this work.

✉ Xiangguo Zhao
zhaoxiangguo@mail.neu.edu.cn

Haojie Nie
qazxse2010@163.com

1 School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China

2 College of Software, Northeastern University, Shenyang 110819, China

3 Key Laboratory of Ministry of Education on Safe Mining of Deep Metal Mines, Northeastern University, Shenyang 110819, China

4 College of Information Science and Technology, Northeastern University, Shenyang 110819, China

5 Thinvent Digital Technology Co.,Ltd., No. 681 Torch Avenue, High-Tech Development Zone, Nanchang 410000, China

Keywords Knowledge graph · Knowledge graph completion · Dynamic negative sampling · Sequence-correlated embedding learning

1 Introduction

In order to efficiently organize and represent the large amount of knowledge information contained in the massive information resources, researchers have proposed the concept of knowledge graphs and applied them to the field of semantic search. Essentially, knowledge graphs provide semantically structured graphical representations of knowledge facts, which are expressed in the form of a triple (h, r, t), where h and t represent head and tail entities, respectively, while r represents relations between entities.

Due to the great representation of knowledge, knowledge graph has attracted widespread attention in the academia and industry, and many of them have applied KG in recommender systems [1, 2], conversational agents [3], and knowledge reasoning [4–6]. Although commonly used large knowledge graphs such as Freebase [7], DBpedia [8], and YAGO [9] contain millions of entities and relations, knowledge graphs still suffer from an incomplete issue. For example, 71% of the person entities in Freebase do not have the attribute of birthplace [10]. A large number of knowledge facts is hidden or missing during the graph construction [11–13], and the incomplete structure and content of knowledge graphs directly affecting downstream intelligent applications. Thus, knowledge graph completion has become a key research topic.

The knowledge graph completion task is to predict the missing part with given the two elements of the triple, such as $(?, r, t)$, $(h, ?, t)$, or $(h, r, ?)$, where the question represents the missing entity or relation. Many scholars have predicted a triple by learning the embedding of entities and relations to complete the missing element. The embedding are learned by scoring the negative sample triples $(\bar{h}, r, \bar{t})$ and the positive triples (h, r, t). However, there are no natural negative samples in existing knowledge graphs. Therefore, it is critical to design an effective negative sampling strategy for embedding learning methods. It is more critical to learn the embedding when negative samples are available.

These methods use random negative sampling [14] to obtain negative training samples. The method of randomly extracting entities from the entity set for constructing negative examples cannot guarantee the quality of the negative samples because the knowledge graph is often incomplete. Low-quality negative samples easily cause the problem of gradient disappearance in the model learning process. For embedding learning, these methods use entity contextual information to enrich entity representation [15] or use hypergraphs to model higher-order relation features. However, these methods ignored the sequential correlation information of the triple and the complex influence of different relations on entities, which will reduce the representation ability of the model.

In this paper, we addressed the two above problems. (1)For the low-quality negative samples problem, we first set the probability of different candidate sets according to different types of relations, so as to reduce the generation of low-quality negative sampling triples. Then we explore negative triples with high semantic similarity to positive triples, and dynamically updates the candidate sets to retain high-quality negative samples for model training. (2)As for the problem of incomplete use of information inside the triples, we consider the sequential information inside the triple and the partial correlation between elements inside the triple. Meanwhile, we adopt CNN to mine the convolutional correlation information between different relations on entities to enhance the embedding representation of knowledge facts.

We design a completion framework that includes a negative sampling algorithm based on dynamic semantic similarity and a correlation embedding model. The negative sampling algorithm mainly includes: 1)*Determine the replacing entity* is used to reduce the generation of low-quality negative sampling triples by setting the different probabilities of replacing the head and tail entities; 2)*Dynamic constructing the negative candidate set* is designed to guarantee the quality of negative samples by constructing from the selected candidate set and comparing the semantic similarity. The embedding model consists of two parts: 1)*Sequential correlation capture module*. This module is to extract sequential correlation information inside the triple, and interactions between the entities and relations; 2)*Convolutional correlation module*: This module adopts convolutional neural networks to capture the complex correlational features and mine the impact of different relations on entities.

To summarize, the main contributions of this paper are as follows.

(1) We design a general negative sampling algorithm based on dynamic semantic similarity, which can capture high-quality negative sample triples for embedding model training.
(2) We design the correlation embedding model, which can enhance the embeddings of knowledge facts by considering the sequential correlated information inside the triples and the correlated features between different entities and relations.
(3) We conducted comparison experiments on two public datasets, WN18RR and FB15K-237. The experimental results verify the performance of the semantic negative sampling strategy and the embedding model proposed in this paper.

The remainder of this paper is organized as follows. Section 2 introduces the related work. Section 3 introduces the problem definition. Section 4 introduces our negative sampling algorithm and embedding model. Experimental results and ablation studies are presented in Section 5. Finally, Section 6 concludes this paper.

2 Related work

The following section introduces the current status of research on negative sampling and completion techniques.

2.1 Negative sampling methods

2.1.1 Static negative sampling

Static negative sampling algorithms ignore the changes in the distribution of samples during the training process and sample directly in the set of entities. The two main static negative sampling algorithms are random negative sampling and Bernoulli negative sampling.

Random negative sampling is one of the most widely used negative sampling methods in knowledge graph embedding learning, such as TransE [14], TransR [16] and TransD [17]. It randomly samples an entity from the set of entities in the knowledge graph to replace the head entity or tail entity in the positive triple (h, r, t) and obtain the negative triple $(\bar{h}, r, t)$ or $(h, r, \bar{t})$.

The random negative sampling approach is simple and efficient. However, randomly drawing entities from the entity set to construct negative examples is highly prone to generating low-quality negative samples because knowledge graphs are often incomplete.

Bernoulli negative sampling is first proposed by TransH [18] and sets different probabilities according to the mapping properties of the relation. The probability of constructing the head

entity negative triple is higher when the mapping property of the relation is "one to-many." In contrast, the algorithm constructs the tail entity negative triple with a higher probability when the relation is "many-to-one."

The interference of low-quality negative triples is avoided to some extent by setting different probabilities of constructing negative triples, however, the Bernoulli negative sampling algorithm cannot guarantee the embedding quality of negative samples dynamically improved with model training.

2.1.2 Dynamic negative sampling

The algorithm with static negative sampling did not observe changes in the sample score distribution of the embedded model during training. Subject to Generative Adversarial Networks (GAN) [19] driven by their success in modeling dynamic distributions, methods such as IGAN [20] and KBGAN [21] have applied GAN to generate negative sampling triples.

Given a positive triple (h, r, t), IGAN models the distribution of all entities by $\bar{h}, \bar{t} \backsim p(e \mid (h, r, t))$, then generates a corresponding negative sample triple $(\bar{h}, r, \bar{t})$. With joint training, IGAN can perform dynamic sampling of negative sample triples.

Rather than modeling the distribution of the entire entity set, KBGAN learns the distribution of negative samples scores in a subset, and sample relatively high-quality triples.

Based on GAN, NSCaching [22] uses a caching mechanism for sampling the higher scoring negative sampling triples, without using generators and discriminators. However, the negative sampling triples have participated in the training process while the model scores them in the cache, and the model has essentially learned the low-quality negative samples.

Although GAN provides a way to model dynamic negative sample distributions, it is very unstable and difficult to train [23], and its performance varies across models. In addition, both IGAN and KBGAN must be pre-trained, which increases the training cost. Therefore, we design a general negative sampling algorithm that can be applied to most KG embedding models for capturing high-quality negative sampling triples.

2.2 Completion methods

Embedding learning-based [24–26] models significantly improve the accuracy of completion by efficiently computing the semantic relations between entities and relations in a low-dimensional space.

The translation model is the most classical approach to knowledge embedding learning. Representative translation models include TransE [14], TransH [18], TransR [16] and TransD [17]. The main idea behind translation models is to consider the process of finding effective triples as the translation operation of entities through relations, and then learn the embedding representation of knowledge by minimizing the marginal-based loss functions of the above models.

Semantic matching models DistMult [27], CompIEx [28] and HoIE [29] use semantic similarity-based scoring functions to mine potential semantic associations between entities and relations, and obtain the likelihood of new facts to hold by embeddings of entities and relations, thus predicting new knowledge and complementing the knowledge graph.

Network representation learning approaches complete the knowledge graph by merging the information extracted from the network topology with the content information of the nodes and edges. The classical network representation learning models include DeepWalk [30], LINE [31], SDNE [32].

Recently, neural network-based completion models have been rapidly developing. ConvE [33] proposed a two-dimensional convolutional neural network to learn the embeddings of knowledge. The ConvKB [34] model improves on the ConvE model by using one-dimensional convolution to extract global features based on relations, and achieves improvements in the accuracy of link prediction. SACN [35] is an end-to-end structure-aware convolutional network. The SACN model utilizes a weighted graph convolutional network as an encoder that aggregates the node structure, node attributes, and edge types of the knowledge graph, and improves the ConvE model with learnable weights applied to the information of locally aggregated neighboring structures. CapsE [36] handles the embeddings of triples using a capsule network. KBGAT [37] incorporates an attention mechanism to capture the influence of neighboring nodes on the entity. DSKG [38] uses recurrent neural networks to capture the sequence information of the triples.

The above representation-based learning completion methods still suffer from the problem of over-simplifying the embeddings of knowledge facts and do not completely consider the diversity and complexity of entities and relations in the knowledge graph, which reduces the accuracy of the model. Therefore, we proposed a correlation embedding model to solve these problems.

3 Problem definition and analysis

3.1 Symbol definition

We first introduce the basic symbols that used in this paper, as shown in Table 1.

3.2 Sample quality definition

Negative sample triples: Suppose there exists a knowledge graph KG. The facts in KG are stored in the set of triples $S = (h, r, t)$, each triple (h, r, t) consists of head entity $h \in E$,

Table 1 Summary of notations

Symbol	Meaning description
E	the set of entities in knowledge graph
R	the set of relations in knowledge graph
h	a head entity, $h \in E$
r	a relation, $r \in R$
t	a tail entity, $t \in E$
(h, r, t)	a fact triple in knowledge graph
$S = \{(h, r, t)\}$	the set of fact triples
$\bar{S} = \{(\bar{h}, r, \bar{t})\}$	the set of negative triples
w_{rh}	the attention of the relation to the head entity
$w_{rh}^{'}$	the normalized attention
$\mathbf{M}_{hr}$	the embeddings matrices of head entity and relation
$\mathbf{M}_{rt}$	the embeddings matrices of relation and tail entity

relation $r \in R$, and tail entity $t \in E$, where E is the set of entities and R is the set of relations. Negative sample triples $(\bar{h}, r, t)$ and $(h, r, \bar{t})$ are derived by replacing the entities of the corresponding positive triples with entities from the knowledge graph.

To better understand the high- and low-quality negative triples, we define sample quality as follows. The negative samples that are closer to the positive triples in terms of semantics are considered to be of high-quality for the embedding model. Meanwhile, the negative samples that are more different from the positive triples in terms of semantics are considered to be of low-quality for the embedding model.

As shown in Fig. 1, the construction of negative triples by random negative sampling is effective in the initial stage of model training. However, as training continues, most of the negative triples are located outside the margin (i.e., inside the blue circle in Fig. 1), and there is a high probability that the negative sample entities inside the blue circle will be selected by random sampling such that the low-quality negative samples are not as far away. This is because the positive samples in the low-quality negative samples are from the positive samples in the semantic space and are prone to the problem of gradient disappearance during model training.

In addition, this conclusion can be obtained by analyzing the objective function of the embedding model. The objective function is shown in (1).

$$\min \sum_{(h,r,t)\in S} \sum_{(\bar{h},r,\bar{t})\in \bar{S}} L(h, r, t) \tag{1}$$

where S is the set of real positive triples in the knowledge graph, $\bar{S}$ is the set of artificially constructed negative triples and $L(h, r, t)$ is the loss function of the embedding model. The calculation of it is as:

$$L(h, r, t) = [\gamma + f(h, r, t) - f(\bar{h}, r, \bar{t})] \tag{2}$$

where γ is the marginal distance, $f(h, r, t)$ is the score of positive samples, $f(\bar{h}, r, \bar{t})$ is the score of negative samples.

Assuming that the embedding model is negatively sampled from within the blue region of Fig. 1, the score of the negative sample triple $f(\bar{h}, r, \bar{t})$ will be relatively large, while the training target of $f(h, r, t)$ is a very small score tending to 0. When the model executes

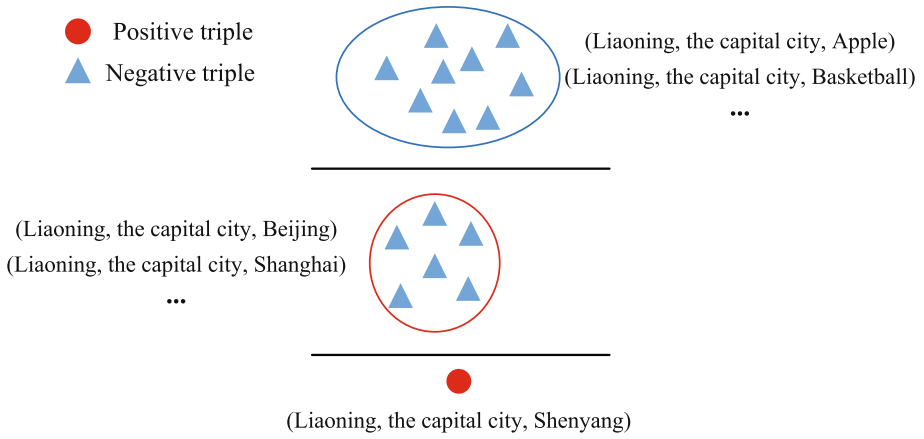


Figure 1 Illustration of the quality of negative sample triples in model training

negative sampling from the blue region will obviously result in the score of $\|f(h,r,t) - f(\bar{h},r,\bar{t})\|$ being too large. Once the score exceed the marginal distance, it will trigger the problem of gradient disappearance.

The sample algorithm is preferred for selecting the triples inside the red dashed circle (high-quality negative samples) in Fig. 1, which can provide more detailed features in the embedding model training process.

3.3 Problem analysis

A high-quality negative sample triple can accelerate model convergence and improve model performance. Given a positive triple *(Liaoning, the capital city, Shenyang)*, if random negative sampling is used, there is a high probability that some unrelated entities with less information, such as *"basketball"* or *"apple"*, will be sampled, and these entities are very different from *"shenyang"* at the semantic level. Therefore, these negative sample triples can only guide the model to distinguish tail entities different from non-city ones during the training process, but cannot help the model to make more detailed distinctions among city-like tail entities.

The two most widely used static sampling methods, i.e. random negative sampling and Bernoulli negative sampling, cannot control the quality of negative triples, and the low-quality negative triples are prone to the problem of loss of 0 for the embedding model based on the marginal loss function $L(h,r,t) = [\gamma + f(h,r,t) - f(\bar{h},r,\bar{t})]$, leading to the phenomenon of gradient disappearance and thus hinders the training of the model. The dynamic negative sampling algorithm applied in KBGAN and IGAN attempts to model the distribution of entities to avoid selecting low-quality triples, however, extra generators and discriminators are introduced, which makes the training process complicated and the model performance unstable.

Moreover, low-quality negative sample triples do not sufficiently help model training and can slow down the convergence rate. Therefore, it is essential to mine high-quality negative triples during the training of the embedding model.

4 Our method

4.1 Completion framework

Figure 2 shows the overall framework of knowledge graph completion that includes three key components: the dynamic semantic similarity negative sampling module, KG embedding module, and completion module.

For the random negative sampling methods easy to generate low-quality negative samples, we design a dynamic semantic similarity negative sampling module to guarantee the quality of negative samples. First, the module sets the probability of different candidate sets according to different types of relations, so as to avoid replacing the original entity with entities of excessive semantic differences. Then the negative triples with high semantic similarity to positive triples are provided for the knowledge graph embedding model, and the negative sample triples participate in the training process of the KG embedding model together with the positive triples.

In order to make better use of the information inside the triples, we design a correlation embedding module to learn the sequential correlation and the convolutional correlation of different relations to entities. First, the module used RNN to capture the sequential correlation

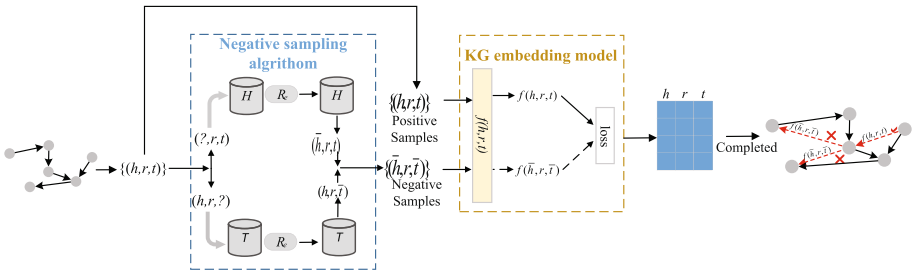


Figure 2 The framework of knowledge graph completion

information. Meanwhile, a CNN is adopted to mine the convolutional correlation of different relations on entities, so as to enhance the embedding representation of knowledge. As the training progresses, the embeddings of the triple (h, r, t) are adjusted with feedback and the candidate set in the negative sampling algorithm is dynamically updated.

Finally, the knowledge graph was complemented based on the scores of the triples. The three key components are described in detail in the following sections.

4.2 Dynamic semantic similarity negative sampling algorithm

In order to provide high-quality negative samples and dynamically update the embedding representation of negative samples to match the current model training, we design the Dynamic Semantic Similarity Negative Sampling algorithm (DSS_Sample) which consists of two procedures: determine the replacing entity and dynamic constructing the negative candidate set. The algorithm can dynamically construct negative samples that are semantically similar to the positive triples from the candidate set. These high-quality negative samples can prevent the problem of gradient disappearance during model training.

4.2.1 Determine the replacing entity

In the process of constructing negative sample triples for the corresponding positive triples, the random negative sampling algorithm generates negative sample triples by replacing the head and tail entities of the positive triples with an equal probability. Then, missing triples were filled by random sampling from the entity set. However, an equal-probability approach to determine head or tail candidate sets may introduce many low-quality negative sample triples during training because real-world knowledge graphs are often incomplete. To minimize this occurrence, we used the Bernoulli negative sampling [18] to calculate the probabilities of replacing the head and tail entities.

When the probability of replacing the head entity is greater than that of replacing the tail entity, we choose to sample from the head entity candidate set H; Conversely, we choose to sample from the tail entity candidate set T. Also, we randomly sample from the set of identified candidates to guarantee the generalizability of the samples.

4.2.2 Dynamic constructing the negative candidate set

The negative-sampling algorithm is guaranteed to maintain high-quality negative samples as the embedding model training proceeds by dynamically constructing the negative candidate

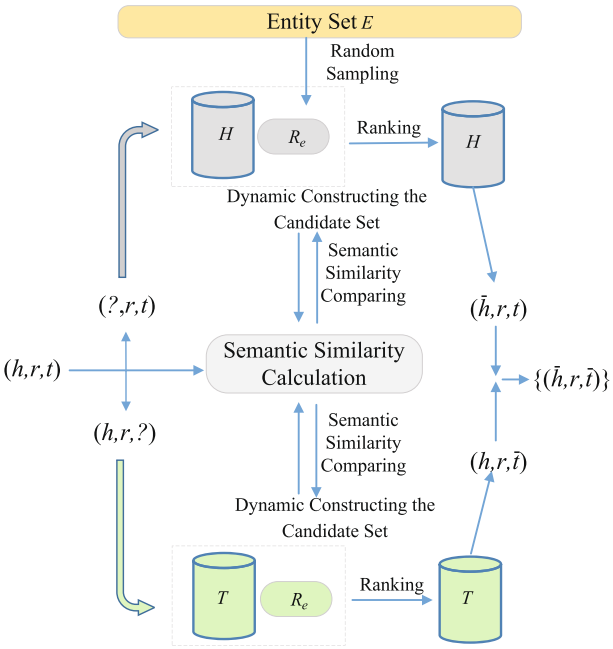


Figure 3 Schematic diagram of constructing and updating the candidate set

set, as shown in Fig. 3. Because the design of the head entity candidate set H and the tail entity candidate set T are the same when dynamically constructing the negative candidate set, we take the head entity candidate set H as an example in this section.

First, we initialize the candidate set H to an empty set and open a supplementary sample cache R_e with a space size of m. Then we randomly sample m entities from the entity set E without putting back at a time. We perform a negative sample filtering operation on each entity to filter out the low-quality negative triples that directly conflict with the existing real triples in the KG. Finally, we execute a union operation between R_e and H to construct a head entity candidate set H containing m entity samples.

$$Sim(h,\bar{h}) = \frac{\sum_{i=1}^{n} h_i \times \bar{h}_i}{\sqrt{\sum_{i=1}^{n} h_i^2} \times \sqrt{\sum_{i=1}^{n} \bar{h}_i^2}} \tag{3}$$

Then, we measure the semantic similarity between the negative samples $\bar{h}$ and the positive samples h in the head candidate set $H^{'}$ by calculating their cosine similarity using (3). The semantically closer entities have cosine values closer to 1. Next, we sort the negative samples in the candidate set in descending order according to their semantic similarity and keep *top-m* negative samples as the updated candidate set. In each epoch, a sample from the candidate set is randomly selected to participate in the training, and the negative sample is removed from the candidate set $H^{'}$. After n epoch iterations, $m - n$ negative samples remain in $H^{'}$. We then repeat the construction of the candidate set, that is, resample R_e containing m negative samples. We compare the similarity with the remaining samples of $m - n$ in the current candidate set, sort them, and select the top m samples. Finally, the candidate set is

constructed and updated to ensure that it can maintain high-quality negative samples as the embedding model training proceeds.

The general process is presented in Algorithm 1. Firstly, the entire data of the knowledge graph is sliced into t copies of size m (lines 1-2). The candidate set is constructed using the Bernoulli probability method and is randomly sampled (lines 3-6). The above steps (lines 1-6) correspond to determine the replacing entity. Subsequently, the semantic similarity between the negative samples $\bar{h}$ and the positive samples h in the candidate set $H^{'}$ is calculated (line 8). The top m negative samples were retained (lines 9-10). Finally, the negative sample triple was constructed for model training and the candidate set was updated (lines 11-15). And these above steps (lines 7-15) correspond to dynamic constructing the negative candidate set.

Algorithm 1 Negative sampling strategy based on dynamic semantic similarity.

Input: training set $S = \{(h, r, t)\}$, head entity candidate set H, tail entity candidate set T
Output: negative sample triple set $\bar{S}$

1: **for** $i = 1$âe¦t **do**
2: Sample a batch $S_{batch} \in S$ of size of m;
3: **for** each $(h, r, t) \in S_{batch}$ **do**
4: Bernoulli: calculate and compare probability p_1 with p_2; ▷ Calculate the probability p to decide whether to replace the head entity or the tail entity.
5: Initialize $H^{'} \leftarrow \varnothing$; ▷ Take the replacement of head entity as an example.
6: Randomly sample a subset $R_e \in H$ with m entities;
7: $H^{'} = H^{'} \cup R_e$;
8: Compute the similarity between h and $\bar{h}$ for all $\bar{h} \in H$;
9: Ranking the similarity in $H^{'}$ in descending order;
10: $H^{'} \leftarrow H^{'}_{top_m}$; ▷ Retain the top m negative samples and update the candidate set
11: **for** i=1,...,n **do**
12: Randomly sample $\bar{h} \in H^{'}$;
13: Construct negative sample $(\bar{h}, r, t)$ for training;
14: $\bar{S} \leftarrow (\bar{h}, r, t)$;
15: Remove $\bar{h}$ from $H^{'}$,
16: **end for**
17: Return $\bar{S}$;
18: **end for**
19: **end for**

4.3 Correlation embedding model

To address the problem that the existing methods make incomplete use of information inside the triples, we consider the sequence information inside the triple, and the correlational information of different relations on entities to enhance the embeddings. And we design the correlation embedding (CorE) model which mainly includes two parts: the sequential correlation capture module and the convolutional correlation module, as shown in Fig. 4.

The initialized embedding representation first passes through a sequential correlation capture module to capture the sequence information inside the triple, and the interactions between entities and relations to enhance the semantic representation of uncommon entities. Then, it uses a relational convolution layer to capture the semantics of the same entities expressed under different relations to enrich the embedding of factual triples representation.

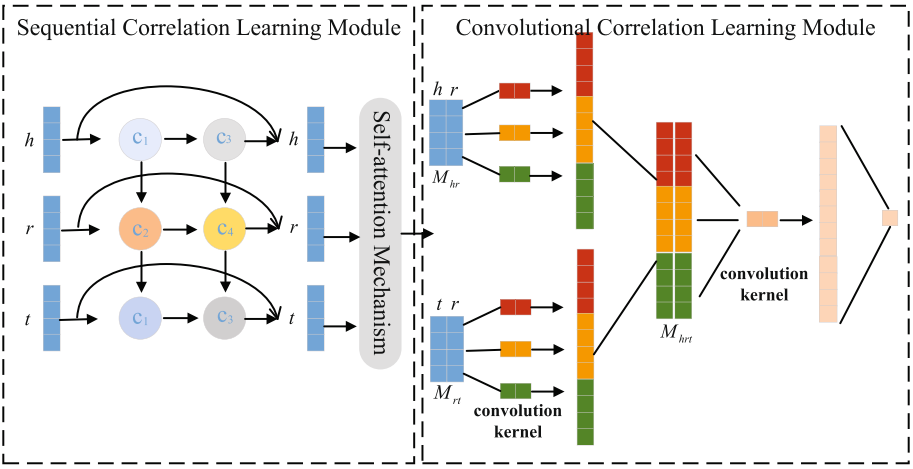


Figure 4 Structure of the correlation embedding model

4.3.1 Sequential correlation capture module

To extract the whole sequential information of the triple, as well as the partial correlation between elements inside the triple, we design a sequential correlation capture medule.

For the whole sequential information of the triple, we use RNN and take the whole triple (h, r, t) as input to capture it inspired by DSKG [38]. Note that triples are not exactly equivalent to sentences with a sequence length of three, especially when the direction of some of the relations does not point explicitly, such as the two triples (*China, City, Shanghai*) and (*Shanghai, City, China*), both of which hold in the knowledge graph, despite the completely different positions of the head and tail entities. Based on this, the initialized embedding of the triples indicates that when acquiring layers through sequence information, the model should capture such information instead of directly learning the current sequence values in relation to the historical sequence values. Therefore, we use the residual network to encapsulate the recurrent neural network to avoid network degradation.

For the partial correlation between elements inside the triple, we design a self-attentive reinforcement layer to capture it, which can enhance the semantic representation of uncommon entities.

We take the relation embeddings $\mathbf{r}$ as an example to explain self-attention learning. First, we perform the dot product operation on the transpose of $\mathbf{r}$ with the head entity, itself, and the tail entity, respectively, to obtain the initialization weight w_{rh}, w_{rr}, w_{rt}. The calculation of w_{rh} is as:

$$w_{rh} = \mathbf{r}^T \mathbf{h} \tag{4}$$

where w_{rh} is the attention of the relation to the head entity, $\mathbf{r}$ and $\mathbf{h}$ are the embeddings of the relation and head entity. The weight w_{rr}, w_{tr} are calculated as above. Then we normalize these weights as:

$$w'_{rh} = \frac{e^{w_{rh}}}{e^{w_{rh}} + e^{w_{rr}} + e^{w_{rt}}} \tag{5}$$

And the normalized weights w'_{rr}, w'_{rt} are calculated in the same way.

By repeating the above steps, we obtain the attention of the head entity for itself w'_{hh}, the relation w'_{hr}, and the tail entity w'_{ht}. And we also get the attention of the tail entity for head entity w'_{th}, the relation w'_{tr}, and itself w'_{tt}.

To extract the sequence-correlated information of the triple, we multiply the normalized attention with the initialized input sequence $\mathbf{X}$ respectively, and obtain the z-vector that incorporates the connection with other elements, as shown in Equation:

$$z_{i\in h,r,t} = \sum_{j\in h,r,t} w'_{ij}\mathbf{X} \tag{6}$$

where w'_{ij} represents the normalized attention, $\mathbf{X} = (\mathbf{h}, \mathbf{r}, \mathbf{t})$.

4.3.2 Convolutional correlation module

To improve the embedding representation of triples with complex relations such as "one-to-many" and "many-to-one"[39], we design a convolutional correlation module. First, this layer extracts the relational transition features from the head entity to the relation and transition features from the relation to the tail entity. Subsequently, the global relational features of the head entity, relation, and tail entity are extracted from the two transition features. The structure is shown in Fig. 5.

The initialization input module takes the output data (embeddings of the head entity $\mathbf{h}$, relation $\mathbf{r}$, and tail entity $\mathbf{t}$) of the self-attention reinforcement layer in the correlation embedding model as the input of this layer.

The first-layer splicing module performs splicing processing on the head entity embeddings $\mathbf{h}$ and the relation embeddings $\mathbf{r}$, the relation embeddings $\mathbf{r}$ and the tail entity

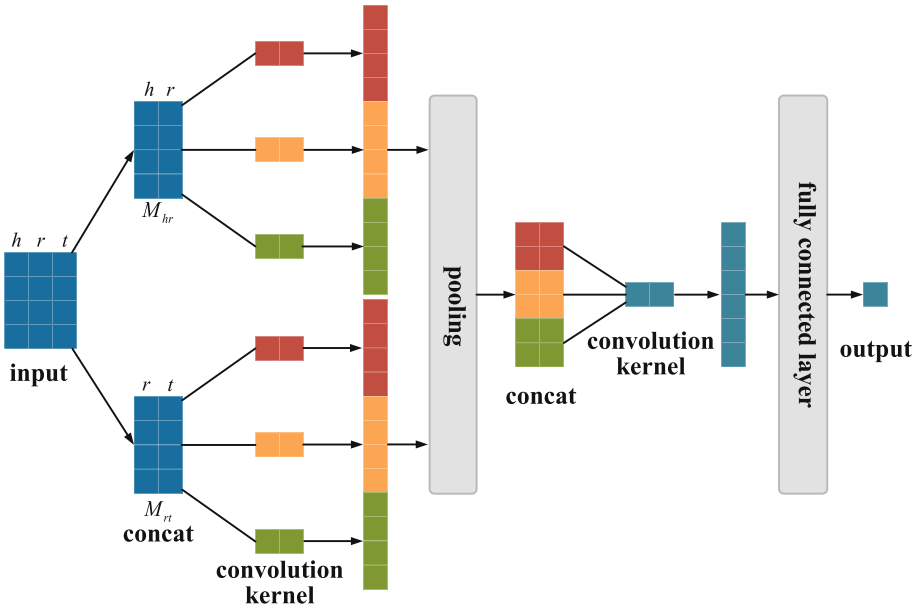


Figure 5 Structure of the relation convolutional layer

embeddings $\mathbf{t}$, and two embeddings matrices $\mathbf{M}_{hr} \in \mathbb{R}^{k\times 2}$ and $\mathbf{M}_{rt} \in \mathbb{R}^{k\times 2}$ with dimension k are obtained.

The first-layer convolution module uses multiple convolution kernels w to act on the matrix splicing M_{hr} and M_{rt} to extract the transition features from the head entity to the relation and the transition feature from the relation to the tail entity, respectively. The convolution kernel w is repeatedly computed on each row of the matrices and generates feature embeddings $\mathbf{v}_{hr} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k]$ and $\mathbf{v}_{hr} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k]$. The embeddings $\mathbf{v}_i$ are calculated as:

$$\mathbf{v}_i = S(w \cdot \mathbf{M} + \mathbf{b}) \tag{7}$$

where $S(\cdot)$ is the ReLU activation function, and $\mathbf{b}$ is the bias term.

The pooling module applies a maximum pooling operation to the feature mapping embeddings $\mathbf{v}_{hr}$ and $\mathbf{v}_{rt}$ to extract the maximum value in the local sensory domain. The size of the pooling window is 2×1 and the step size is 2. After pooling, the feature mapping vectors are updated to $\mathbf{v}^{'}_{hr} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{k/2}] \in \mathbb{R}^{k/2}$ and $\mathbf{v}^{'}_{rt} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{k/2}] \in \mathbb{R}^{k/2}$.

The second-layer splicing module continues the splicing operation on the pooled mapping embeddings $\mathbf{v}^{'}_{hr}$ and $\mathbf{v}^{'}_{rt}$ to obtain a triple mapping matrix $\mathbf{M}_{hrt} \in \mathbb{R}^{k/2}$.

The second-layer convolution module also uses multiple convolution kernels w to act on the triple mapping matrix $\mathbf{M}_{hrt}$ to extract global information, and then obtain the feature embeddings $\mathbf{v}^{'}_{hrt} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{k/2}] \in \mathbb{R}^{k/2}$. This module can extract the features of triple entities and relations in each dimension as a whole.

Finally, after $\mathbf{v}^{'}_{hrt}$ is flattened, it is sent to the fully connected layer. The flattened triple feature embeddings $\mathbf{v}^{'}_{hrt}$ and parameter w perform a dot product operation to obtain the triple score. This score can be used as the basis to judge whether the triplet is true. The score is calculated as:

$$f(h, r, t) = flatten(\mathbf{v}^{'}_{hrt}) \cdot w \tag{8}$$

4.4 Loss function

The correlation embedding model adopts a pair-wise training method, which needs to be learned by minimizing the score of positive triples (h, r, t) and maximizing the score of unobserved negative triples $(\bar{h}, r, t)$ or$(h, r, \bar{t})$. Therefore, we adopt the loss function which taking into account the score of positive and negative examples in TransE [14]. The loss function is as:

$$L(h, r, t) = [\gamma + f(h, r, t) - f(\bar{h}, r, \bar{t})] \tag{9}$$

The training objective is to minimize the loss function.

5 Experiments

In this section, the negative sampling algorithm based on dynamic semantic similarity and the correlation embedding knowledge graph embedding model are experimentally verified and analyzed.

5.1 Dataset

We evaluate the dynamic semantic similarity based negative sampling algorithm DSS_Sample with correlation embedding model on two benchmark datasets WN18RR and FB15K-237.

Table 2 The statistic details of the datasets

Dataset	#Entity	#Relation	#Training set	#Validation set	#Testing set
WN18RR	40943	11	86835	3034	3134
FB15K-237	14541	237	272115	17535	20466

Among them, FB15K-237 is a subset of the relational database Freebase and contains a large number of fact triples. WN18RR is a subset of WordNet and contains the lexical relation between two words. The statistical details of the datasets used in this paper are shown in Table 2.

5.2 Evaluation metrics

Three common evaluation metrics are used in the evaluation process for the link prediction task described above.

(1) MR: MR is the average ranking of all positive triples in the test set. The higher the ranking of the positive triples, the higher the accuracy of the algorithm. Therefore, a smaller MR indicates a better experimental result.
(2) MRR: MRR is calculated from the inverse of the ranking of each positive triple and is the average inverse of the ranking of all positive triples in the test set. Therefore, a larger MRR indicates a more satisfactory experimental result.
(3) HIT@n: HIT@n is the proportion of positive triples in the test set in the top n rankings of the predicted scores. Therefore, a larger HIT@n indicates a higher accuracy of link prediction.

5.3 Experimental setup

5.3.1 Baseline

First, we compare our negative sampling algorithm with the static and dynamic negative sampling algorithms.

(1) *Static negative algorithm*: Random negative sampling [14] is the most traditional sampling method in the KG embedding model. The Bernoulli negative [18] sampling algorithm reduces the generation of low-quality negative sample triples by setting the probability of replacing the head entity or the tail entity differently.
(2) *Dynamic negative sampling algorithm*: KBGAN first randomly draws a candidate set $Neg = (\bar{h}, r, \bar{t})$ from the entire set E. Then, it samples triples from Neg. In the framework of GAN, the generator in KBGAN can approximate the score distribution of the triples in the set Neg and sample a higher quality triplet.

Second, we compare the correlation embedding with the following knowledge graph embedding methods to verify the performance of the embedding completion model.

(1) *Translation-based models*: TransE [14] adopts the idea of translation $h + r \approx t$ to model the relation between entities. TransH [18] extends TransE by projecting the head entity and the tail entity into the relation hyperplane. TransD [17] considers the diversity of entities and relations based on TransE.

(2) *Semantic matching-based models*: DISTMULT [27] uses a diagonal matrix to represent relations. CompIEx [28] extends DisMult to complex space.
(3) *Neural network-based models*: KBGAN [21] learns the embeddings of knowledge facts based on GAN. DSKG [38] uses recurrent neural networks to capture the sequence information of triples. ConvE [33] uses a 2D CNN to capture the interactions between the entities and relations. ConvKB [34] extends ConvE by using 1D CNN to capture global relations between entities. CapsE [36] uses capsule neural networks to learn the embeddings. KBGAT [37] incorporates an attention mechanism to capture the influence of neighboring nodes on the entity.

5.3.2 Data initialization and parameter setting

For the negative sampling experiments, we adopt two training strategies, which are training from scratch and training with pre-trained data, on dynamic negative sampling algorithm.

(1) Training from scratch: The embedding representations of relations and entities are initialized by the Xavier unified initialization procedure, and the initialization data of the given KG are directly used in the four negative sampling algorithms, including random negative sampling, Bernoulli negative sampling, KBGAN+scratch and DSS_Sample+scratch.
(2) Using pre-trained data: Each KG embedding model is first pre-trained under the baseline algorithm (random negative sampling algorithm), and training is stopped after reaching the optimal performance. In this paper, we use the KG pre-trained data as a base in a hot-start manner, and then apply different negative sampling algorithms to continue the training.

The parameters of the DSS_Sample algorithm proposed in this paper are as follows: The size of the set of candidate entities m is 50 and the number of iterations n is 20 (i.e., the set of candidates is negatively sampled and updated every 20 epochs).

For the experiments on embedding models, we use the Adam optimizer and chooses its initial learning rate to be 0.001, save the results every 5 epochs, and initialize the embedding dimensionality of entities and relations to be 100 dimensions. The detailed parameters corresponding to the two datasets are as follow:

(1) WN18RR dataset: *batch_size* is 1024; the number of convolutional kernels per layer of the convolutional correlation module is 100; the λ is set to 1; *num_epoch* is 60.
(2) FB15K-237 dataset: *batch_size* is 2048; the number of convolutional kernels per layer of the convolutional correlation module is 50; the λ is set to 1; *num_epoch* is 60.

5.4 Experimental results and analysis

5.4.1 Negative sampling algorithm experiments

To demonstrate the wide applicability of the DSS_Sample algorithm, the negative sampling algorithm is experimented on the link prediction tasks of four embedding models, such as TransE, TransD, TransH and CorE, respectively. In the above experimental results, the bolded metrics denote the experimental results of the optimal embedding model corresponding to the negative sampling algorithm, and the underlined metrics denote the results of the sub-optimal corresponding embedding model. The experimental results are shown in Table 3.

Table 3 Results of the comparison of negative sampling performance

Models	Dataset	WN18RR			FB15K-237		
	Metrics	MR	MRR	HIT@10	MR	MRR	HIT@10
TransE	Random	4038	0.175	44.5	237	0.226	38.6
	Bernoulli	3924	0.178	45.1	197	0.256	41.9
	KBGAN+pretrain	4420	0.186	45.4	628	0.294	46.7
	KBGAN+scratch	5356	0.181	43.2	722	0.293	46.6
	DSS_Sample+pretrain	3982	**0.216**	46.9	181	0.300	47.0
	DSS_Sample+scratch	**3902**	0.205	**47.3**	**178**	**0.301**	**47.1**
TransD	Random	4955	0.178	42.2	215	0.224	39.5
	Bernoulli	**3555**	0.190	46.4	197	0.256	41.9
	KBGAN+pretrain	3785	0.192	46.5	628	0.294	46.7
	KBGAN+scratch	4083	0.188	46.4	722	0.293	46.6
	DSS_Sample+pretrain	3745	**0.235**	50.0	238	0.280	**45.8**
	DSS_Sample+scratch	3736	0.234	**50.2**	**179**	**0.281**	45.7
TransH	Random	5646	0.175	43.3	223	0.222	38.8
	Bernoulli	4113	0.186	45.1	202	0.233	40.1
	KBGAN+pretrain	4708	0.192	45.3	401	0.281	46.4
	KBGAN+scratch	4881	0.187	44.8	455	0.278	46.2
	DSS_Sample+pretrain	4347	**0.288**	50.5	219	**0.300**	**47.5**
	DSS_Sample+scratch	**4005**	0.283	**50.6**	**199**	0.297	47.5
CorE	Random	729	0.578	63.8	173	0.603	66.0
	Bernoulli	704	0.590	63.9	156	0.606	66.3
	KBGAN+pretrain	809	0.533	62.3	453	0.538	64.5
	KBGAN+scratch	939	0.512	61.8	653	0.504	63.5
	DSS_Sample+pretrain	701	**0.599**	64.1	**149**	**0.636**	**66.7**
	DSS_Sample+scratch	**684**	0.593	**64.2**	169	0.612	66.4

Overall analysis Compared with the random negative sampling algorithm, the DSS_Sample algorithm shows a general performance improvement over each embedding model. For example, in the FB15K-237 dataset, link prediction experiments on the TransE embedding model show that the DSS_Sample algorithm has 59, 0.075, and 8.5% improvements in MR, MRR, and HIT@10, respectively. Meanwhile, the DSS_Sample algorithm has the same advantages as the Bernoulli sampling algorithm and KBGAN negative sampling algorithm.

The above experimental results illustrate that the dynamic semantic similarity-based negative sampling algorithm proposed in this paper can effectively extract high-quality negative sample triples and help the embedding model obtains a higher performance than other negative sampling algorithms. Similarly, the embedding model considering the sequential correlation information of triples and the convolutional correlation information of different relations to entities has a higher accuracy compared with other models.

Pre-training analysis In this paper, we compare the performance of negative sampling algorithms under two different data initialization modes, pretrain and scratch. The results of comparing KBGAN+pretrain and KBGAN+scratch show that KBGAN with pre-training has a better performance. This indicates that dynamic negative sampling algorithm requires additional pre-training to improve performance.

In contrast, the performance of the KG embedding model trained by the DSS_Sample algorithm proposed in this paper outperforms the GAN-based negative sampling algorithm in either state (pretrained or scratch). Moreover, the results of DSS_Sample +pretrain and DSS_Sample+scratch roughly converge. This result verifies that DSS_Sample does not require additional pre-training operations, and has higher accuracy than GAN-based sampling method.

Stability analysis When testing the link prediction effect of the TransH on the WN18RR dataset, it was found that the KBGAN negative sampling algorithm did not outperform the Bernoulli negative sampling algorithm on the MR metric. It performs even worse than the random negative sampling method on the CorE model. Therefore, KBGAN does not exhibit consistent performance over different embedding models. This observation further confirms that GAN-based methods suffer from instability. In contrast, the DSS_Sample algorithm designed in this paper performs consistently across embedding models with stable performance.

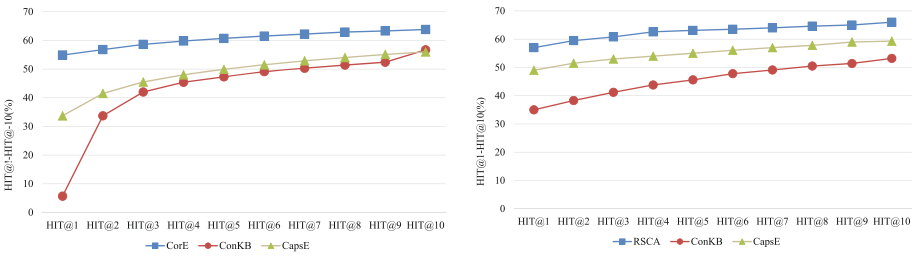
5.4.2 Embedding model experiments

Table 4 shows the overall performance of the CorE model and its comparison models on top of the link prediction task. The CorE model designed in this paper shows almost optimal results on all three metrics for both datasets (WN18RR and FB15K-237), except that the MR metric is slightly lower than that of the CapsE model on the WN18RR dataset.

Embedding performance results on HIT@1 to HIT@10 In order to reflect the performance of the CorE model on the link prediction task, the experiments calculate the metrics HIT@1 to HIT@10 based on the test set. We consider that the HIT@1 metric is very important, as a higher HIT@1 means that the model is more capable of directly predicting the correct answer at once, which better reflects the accuracy of the model. Therefore, this paper compares two current embedding models with higher accuracy, ConvKB and CapsE, on the WN18RR and FB15K-237 datasets. Figure 6a and b show the experimental results on the WN18RR

Table 4 Results of comparing the embedding performance

Methods	WN18RR			FB15K-237		
	MR	MRR	HIT@10	MR	MRR	HIT@10
KBGAN	3785	0.192	46.5	825	0.247	44.4
DISTMULT	5100	0.430	49.0	254	0.241	41.9
CompIEx	5261	0.440	51.1	339	0.247	42.8
ConvE	5277	0.460	48.0	246	0.316	49.1
TransE	4038	0.175	44.5	237	0.226	38.6
TransH	5646	0.175	43.3	223	0.222	38.8
TransD	4955	0.178	42.2	215	0.224	39.5
ConvKB	763	0.253	56.7	254	0.418	53.2
CapsE	**719**	0.415	56.0	303	0.523	59.3
DSKG	-	-	-	175	0.339	52.1
KBGAT	1921	0.412	55.4	270	0.157	33.1
CorE	729	**0.578**	**63.8**	**173**	**0.603**	**66.0**



(a) WN18RR (b) FB15K-237

Figure 6 Embedding performance evaluation of HIT@1 to HIT@10

and FB15K-237 datasets, respectively. The metrics in bold font indicate the best current experimental results.

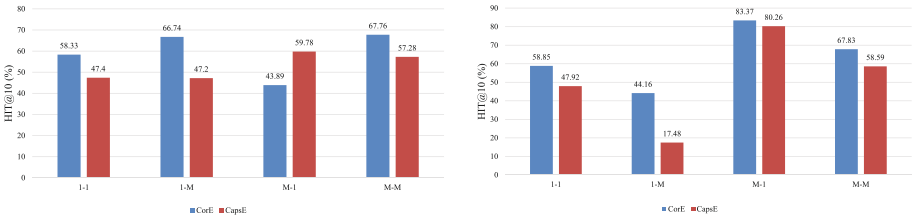
The results presented in Fig. 6a and b show that, compared with the current ConvKB model and CapsE model, the CorE model shows a significant improvement in the HIT@1. Especially on the WN18RR data, the HIT@1 metric of the CorE model increased by 49.2% when compared with the ConvKB. The probability of hitting the correct answer at one time far exceeds that of other models on both datasets. The evaluation index became more relaxed as n increased. Clearly, for each index from HIT@1 to HIT@10, the CorE model is better than the other models.

Embedding performance results on different types of relation To verify the performance of the CorE model on different relations, we divide the relation categories in the dataset FB15K-237, which are 1-1, 1-M, M-1, and M-M. After classifying the relations, the CorE model is evaluated using HIT@10 and MRR metrics. The model performance of the HIT@10 metric on the four different relation types for predicting head entities and tail entities as shown in Fig. 7a and b, respectively.

The results in Fig. 7a and b show that, the CorE model is more effective than the current best-performing CapsE model in all tasks, except for the task of predicting the head entity in the M-1 relation type. This implies that the CorE model can effectively distinguish different relation types, and obtain a good embedding representation of the triples.

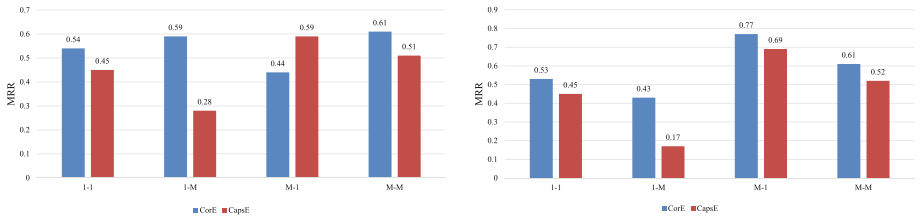
The performance of the MRR metrics is shown in Fig. 8a and b.

The proposed CorE model has stable performance on the four types of relations in the FB15K-237 dataset, as shown in Fig. 8a and b. For example, in the prediction task of tail



(a) Predicting head entity (b) Predicting tail entity

Figure 7 Embedding performance comparison of HIT@10 with different types of relation



(a) Predicting head entity (b) Predicting tail entity

Figure 8 Embedding performance comparison of MRR with different types of relation

entities, the MRR index extreme value difference for the four relation types of the CorE model is 0.34, which is smaller than the extreme value difference of CapsE (0.52). The CorE model does not exhibit large fluctuations in performance because of the different types of data relations and has high stability.

6 Conclusion

Most studies on knowledge graph completion tasks focus on the design of embedding models, and only few studies were conducted on negative sampling techniques. However, the quality of negative samples directly affects the performance of embedding models. In this paper, we propose a negative sampling algorithm based on the dynamic semantic similarity to solve the problem of gradient disappearance caused by low-quality triples in the training process of the embedding model, which can gradually explore high-quality negative samples to participate in the training as the embedding model learns to help improve the performance of the embedding model. Meanwhile, we design the correlation embedding model by extracting the correlation between different entities and relations and sequence information inside the triples, to enrich the embeddings of the triples. The experimental results on the two benchmark datasets verify the performance of proposed method.

Acknowledgements This work is partially supported by the National Natural Science Foundation of China under Grants No.62072087, 62002054, 61972077, 61932004, U2001211.

Author Contributions Haojie Nie: Conceptualization, Methodology, Project administration. Xiangguo Zhao: Software, Formal analysis, Writing-Original Draft. Xin Bi: Validation, Resources. Yuliang Ma: Visualization, Investigation. George Y. Yuan: Writing-Review&Editing.

Availability of data and materials The dataset we use is the public benchmark dataset.

Declarations

Ethical Approval not applicable

Competing interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Wang, J., Shi, Y., Li, D., Zhang, K., Chen, Z., Li, H.: Mcha a multistage clustering-based hierarchical attention model for knowledge graph-aware recommendation. World Wide Web **25**(3), 1103–1127 (2022)
2. Huang, Y., Zhao, F., Gui, X., Gui, H.: Path-enhanced explainable recommendation with knowledge graphs. World Wide Web. **24**(5), 1769–1789 (2021)
3. Liao, J., Zhao, X., Tang, J., Zeng, W., Tan, Z.: To hop or not, that is the question Towards effective multi-hop reasoning over knowledge graphs. World Wide Web. **24**(5), 1837–1856 (2021)
4. Mehmood, Q., Saleem, M., Jha, A., d'Aquin, M.: Efficient distributed path computation on RDF knowledge graphs using partial evaluation. World Wide Web. **25**(2), 1005–1036 (2022)
5. Cai, B., Xiang, Y., Gao, L., Zhang, H., Li, Y., Li, J.: Temporal knowledge graph completion: A survey. CoRR arXiv:2201.08236 (2022)
6. Xue, G., Zhong, M., Li, J., Chen, J., Zhai, C., Kong, R.: Dynamic network embedding survey. Neurocomputing. **472**, 212–223 (2022)
7. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250 (2008)
8. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia A nucleus for a web of open data. In: The Semantic Web, pp. 722–735 (2007)
9. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706 (2007)
10. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proc. IEEE **104**(1), 11–33 (2016)
11. Le, T., Le, N., Le, B.: Knowledge graph embedding by relational rotation and complex convolution for link prediction. Expert Syst. Appl. **214**, 119122 (2023)
12. Liang, S., Shao, J., Zhang, D., Zhang, J., Cui, B.: DRGI deep relational graph infomax for knowledge graph completion. IEEE Trans. Knowl. Data Eng. **35**(3), 2486–2499 (2023)
13. Shen, T., Zhang, F., Cheng, J.: A comprehensive overview of knowledge graph completion. Knowl. Based Syst. **255**, 109597 (2022)
14. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in neural information processing systems. **26** (2013)
15. Stoica, G., Stretcu, O., Platanios, E.A., Mitchell, T., Póczos, B.: Contextual parameter generation for knowledge graph link prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3000–3008 (2020)
16. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI Conference on Artificial Intelligence (2015)
17. Ji, G., He, S., Xu, L., Liu, K., Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing vol.1, pp. 687–696 (2015)
18. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol.28 (2014)
19. Xiao, C., Li, B., Zhu, J.-Y., He, W., Liu, M., Song, D.: Generating adversarial examples with adversarial networks. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 3905–3911 (2018)
20. Wang, P., Li, S., Pan, R.: Incorporating gan for negative sampling in knowledge representation learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
21. Cai, L., Wang, W.Y.: Kbgan Adversarial learning for knowledge graph embeddings. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol.1, pp. 1470–1480 (2018)
22. Zhang, Y., Yao, Q., Shao, Y., Chen, L.: Nscaching simple and efficient negative sampling for knowledge graph embedding. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 614–625 (2019)
23. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems **30** (2017)
24. Yang, S., Tian, J., Zhang, H., Yan, J., He, H., Jin, Y.: Transms Knowledge graph embedding for complex relations by multidirectional semantics. In: IJCAI, pp. 1935–1942 (2019)
25. Cui, Z., Liu, S., Pan, L., He, Q.: Translating embedding with local connection for knowledge graph completion. In: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, pp. 1825–1827 (2020)

26. Zhang, Z., Cai, J., Zhang, Y., Wang, J.: Learning hierarchy-aware knowledge graph embeddings for link prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3065–3072 (2020)
27. Yang, B., Yih, S.W.-t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: Proceedings of the International Conference on Learning Representations (ICLR) 2015 (2015)
28. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning, pp. 2071–2080 (2016)
29. Nickel, M., Rosasco, L., Poggio, T.: Holographic embeddings of knowledge graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
30. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
31. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line Large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1067–1077 (2015)
32. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1225–1234 (2016)
33. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
34. Nguyen, T.D., Nguyen, D.Q., Phung, D., et al.: A novel embedding model for knowledge base completion based on convolutional neural network. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. **2**, pp. 327–333 (2018)
35. Shang, C., Tang, Y., Huang, J., Bi, J., He, X., Zhou, B.: End-to-end structure-aware convolutional networks for knowledge base completion. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol.33, pp. 3060–3067 (2019)
36. Vu, T., Nguyen, T.D., Nguyen, D.Q., Phung, D., et al.: A capsule network-based embedding model for knowledge graph completion and search personalization. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1, pp. 2180–2189 (2019)
37. Nathani, D., Chauhan, J., Sharma, C., Kaul, M.: Learning attention-based embeddings for relation prediction in knowledge graphs. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4710–4723 (2019)
38. Guo, L., Zhang, Q., Ge, W., Hu, W., Qu, Y.: Dskg A deep sequential model for knowledge graph completion. In: China Conference on Knowledge Graph and Semantic Computing, pp. 65–77 (2018)
39. Bi, X., Nie, H., Zhang, X., Zhao, X., Yuan, Y., Wang, G.: Unrestricted multi-hop reasoning network for interpretable question answering over knowledge graph. Knowl. Based Syst. **243**, 108515 (2022)