Knowledge-Based Systems 256 (2022) 109889

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A knowledge graph completion model based on contrastive learning and relation enhancement method

LinYu Li [a], Xuan Zhang [a,b,c,*], YuBin Ma [a], Chen Gao [d], Jishu Wang [d], Yong Yu [a,b,c], Zihao Yuan [a], Qiuying Ma [a]

[a] School of Software, Yunnan University, Yunnan, 650091, China
[b] Yunnan Key Laboratory of Software Engineering, Yunnan 650091, China
[c] Engineering Research Center of Cyberspace, Yunnan, 650091, China
[d] School of Information Science & Engineering, Yunnan University, Yunnan, 650091, China

ARTICLE INFO

Article history:
Received 7 August 2022
Received in revised form 7 September 2022
Accepted 11 September 2022
Available online 16 September 2022

Keywords:
Knowledge graph
Knowledge graph completion
Contrastive learning
Graph attention network
Link prediction

ABSTRACT

The rapid development in knowledge graph (KG) technology and its popularity in the field of artificial intelligence (AI) have significantly increased the support for similar KG-based applications. However, there is a concerning problem regarding KGs; most of them are often incomplete. This motivated us to study knowledge graph completion (KGC). Some recent studies have used graph neural networks (GNN) such as graph convolutional networks (GCN) to model graph-structured data, providing good results on KGC tasks. However, the edge weights in GCN models are controlled by degree, a measure that moderately ignores the differences among relation information. To address the above limitations and obtain better KGC, we propose a model based on graph attention networks (GATs) and contrastive learning (CL), called the CLGAT-KGC model. This model introduces the graph attention mechanism and adds different representations of entities under the same entity corresponding to different relations to enhance the entity-relation message function. Additionally, a new CL method is proposed under the CLGAT-KGC model to better learn the embedding of entities and relations in the KG domain. We have completely verified the effectiveness of this model through extensive experiments.

1. Introduction

With the gradual popularization of the Internet and emergence of various new applications, plentiful data resources have been generated; and the problem of organizing and expressing the useful data sources from these has attracted the attention of many researchers. Thus, Google proposed the concept of knowledge graph (KGs) [1] and KG technology has gained great attention in recent years. To put it simply, a KG is a directed graph with entities as nodes and relations as edges. KGs model real-world information in the form of triples. Each triple can be expressed as (*head entity*, *relation*, *tail entity*) or (h, r, t), where h, r, and t correspond to the head entity, relation, and tail entity, respectively, e.g., (*The Great Wall, IsLocatedIn, China*).

KGs extract, organize, and manage knowledge from many data resources to provide users with intelligent services that can meet their needs, which includes understanding semantic search and providing more accurate search answers or appropriate recommendation services. There are many KGs; the representative ones are KnowItAll [2], Yet Another Great Ontology (YAGO) [3,4], DBpedia [5], Freebase [6], Never-Ending Language Learning (NELL) [7], and Probase [8]. Even widely-used KGs such as these are still far from being complete and comprehensive. The incompleteness of KGs greatly affects their quality, hindering KG support for artificial intelligence (AI)-related applications. Thus, knowledge graph completion (KGC), also known as the link prediction task, has become deeply important in KG research. The core of KGC involves logically understanding KGs, with the main aim of predicting the missing elements in the triples. More formally, the goal of KGC is predicting the head entity (?, r, t) or tail entity (h, r, ?) in a given query.

Currently, any advanced inference models are based on embedding approaches. Knowledge graph embedding (KGE) models

Abbreviations: AI, Artificial intelligence; KG, Knowledge graph; KGC, Knowledge graph completion; GAT, Graph attention network; CL, Contrastive learning; CLGAT-KGC, Contrastive learning graph attention network-knowledge graph completion; GNN, Graph neural network; GCN, Graph convolutional network; YAGO, Yet Another Great Ontology; NELL, Never-Ending Language Learning; KGE, Knowledge graph embedding; NTN, Neural tensor network; r-GCN, Relational graph convolutional network; SACN, Searching architecture calibration network; CompGCN, Composition-based multi-relational graph convolutional network; MRR, Mean reciprocal rank; MR, Mean rank; A2N, Attention in attention; MRGAT, Multi-relational graph attention network; HRAN, Hierarchical residual attention network; LTE, Linearly-transformed entity embedding; DeepER, Deep Entity Resolution

* Corresponding author at: School of Software, Yunnan University, Yunnan, 650091, China.
E-mail address: zhxuan@ynu.edu.cn (X. Zhang).

https://doi.org/10.1016/j.knosys.2022.109889
0950-7051/© 2022 Elsevier B.V. All rights reserved.

are generally classified into four categories [1]: linear/bilinear models (TransE [9], DistMult [10], Complex [11], HolE [12], RotatE [13]); factorization models (TuckER [14], LowFER [15]), neural network models (neural tensor network (NTN) [16], ConvE [17], ConvKB [18]); and ontology models (TransO) [19]. Typically, these models embed entities and relations into a low-dimensional distributed representation based on existing triples in KGs. Entity and relational embeddings are obtained by optimizing the scoring function defined on each fact triple (h, r, t) to measure the plausibility of the embedding. Nevertheless, most of these embedding models only study the triadic information in isolation, ignoring the valuable information that is embedded and available in the whole KG. The largest typical drawback of these embedding models is that they do not fully exploit the rich neighborhood structure of each entity and the additional information that may be hidden in the KG. This limits both the inference capability and interpretability of the embedding space. Additionally, we should consider that the locations of these models in the embedding space are not fixed in the entity representation – the locations have been called "uncontrollable"– and this can lead to similar or dissimilar entities with similar embedding locations incorrectly becoming the correct inference when we make inference predictions; this seriously limits the inference and its effect.

Motivated to solve these issues, we made full use of the domain information of each entity and the most representative transfer information. Regarding neighborhood learning, using graph neural networks (GNNs) [20–22] is the most ideal choice, with their excellent performance in representing the neighborhood information from a given node. These networks are widely used in computer vision, natural language processing, and knowledge tracking [23]. The edge weights in common graph convolutional network (GCN)-based models [20] are only controlled by degree, moderately ignoring the difference between each piece of relational information.

The following relations are shown in Fig. 1: (Lionel Messi, plays forward for, Paris Saint-Germain); (Kylian Mbappé, plays forward for, Paris Saint-Germain); and (Ángel Di María, plays forward for, Paris Saint-Germain). Three players simultaneously play for the same club, but each player is not necessarily equally important for the club; so, they should have different weights when representing the entities and relations. Based on this and graph attention networks (GATs) [21], we have introduced a graph attention mechanism. In Fig. 1, there may be multiple relations between the entities. For example, between the entities Lionel Messi and Argentina, there are multiple relations such as *plays forward in*, *born in*, and *be captain*. In this case, when link prediction is performed under the traditional KGC model, which is based on GNNs, it is often affected by the noise of these multiple relations. Inspired to solve this problem and based on the working of CoPER [24], we propose a relation-enhanced entity representation method, which can enhance the representation of entities under different relations. We wanted the embedding representation of each entity to be slightly different when the same entity was linked to different relations. Accordingly, we propose a method for augmenting the message function to enhance the different representations of entities under multiple relations. Here, unlike how it is represented in the traditional GCN model, not only a fixed matrix is used to obtain messages. Specifically, in addition to the weight matrix shared across different relations, we separately learned a specific weight matrix for each relation to enhance the representation of entities for different relational links.

A recent paper [25] has stated, "The transformation of entity representations can effectively improve the performance of KGE models as long as the GCN can distinguish entities with different semantics through the entity representations it generates". Certain KGC methods have not been able to fully tap the potential of contrastive learning (CL). Thus, we thought of introducing CL into the field of KGC to provide KGC tasks with a more powerful knowledge representation capability. Specifically, we used CL to augment the representations of entities and relations. Inspired by CL methods in the computer vision domain, such as SimSiam [26]; MoCo [27]; SimCLR [28]; the recommender system domain [29–31]; and the natural language processing domain [32–34]; we propose two node-level CL methods to aggregate and model rich neighborhood information to improve entity and relation representation learning. Unlike traditional graph augmentation types that involve the random sampling of nodes to the graph or the construction of augmented graph CL by destroying subgraphs, we preferred to mine potential neighborhood relations from node-level contrast representation to enhance the learning ability of the node representation. The first contrast representation aimed to extract entity nodes as well as representations of relations and their hierarchical neighbor embeddings for contrast. The second contrast representation was based on the potential semantic relations of nodes – which may not be reachable on the graph but have some similar semantics – that were to be compared to capture the correlation between entity node representations and their corresponding prototype embeddings. These two contrast representations were used to improve the representation of entity node embeddings with relational embeddings. Although distance-based KGE models can achieve a better performance in link prediction, most only model either connection patterns or relation mapping properties and not both. For example, RotatE [11] is good at modeling connection patterns but weak at modeling the mapping properties of relations; and it cannot handle knowledge representation under more relations. Composition-based multi-relational graph convolutional network (CompGCN), like the GCN-based model, only relies on the degree of nodes to judge its importance; lacks the ability to handle multiple relations; and does not have a better representation for distinguishing entities and relations. Also, our experimental results showed that our proposed contrastive learning graph attention network-knowledge graph completion (CLGAT-KGC) model performed more comprehensively on both datasets than other existing models did.

The contributions of our model CLGAT-KGC are summarized in the following three aspects.

1. Our proposed relation-enhanced entity representation method enriches different types of relations and makes the representation of each entity unique according to its combination with different relations, enhancing knowledge representation learning.
2. To improve the learning of entity and relation representations, we propose two CL strategies for KGC, one based on the hierarchy of neighbors in the graph and the other based on potential semantic relations.
3. The effectiveness of our model for link prediction tasks under KGC has been verified through extensive experiments on both the FB15K-237 and WN18RR datasets.

The remainder of this paper is structured as follows. We first review the related work in Section 2. Then, we thoroughly describe our proposed CLGAT-KGC model in Section 3. Finally, the experiments are discussed in Section 4, and the conclusion and scope for future work are presented in Section 5.

2. Related work

For easier understanding, first, we have introduced some mainstream KG inference methods for KGC such as linear/bilinear models, factorization models, neural network-based models, and ontology models. Secondly, in Section 2.2, we briefly introduce GNNs and introduce the most relevant mainstream models for KGC based on GNNs for comparison purposes.

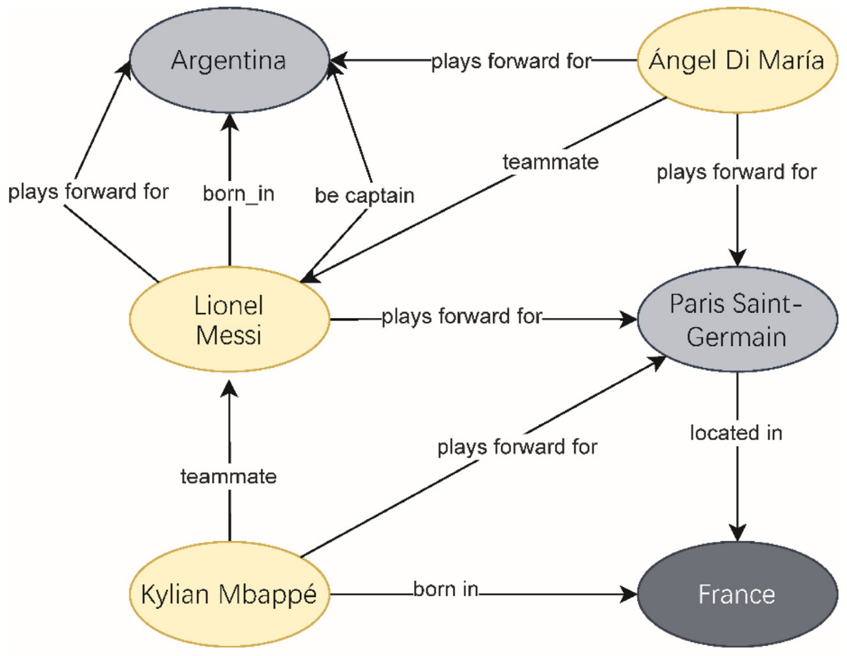

Fig. 1. Example of a multi-relational KG.

2.1. Representation learning of KGs

KG representation learning has been attracting attention as a common means for KGC. Among KG representation learning methods, the most widely used are embedding models, which have shown high efficiency in many KG-based applications. Most of these embedding models work by learning the representation of entities and relations in a low-dimensional space and then using the learned tensors to predict the missing elements. We generally classify these embedding models into four categories: translation-based models, tensor decomposition-based models, neural network-based models, and ontology models.

For each triple (h, r, t), translation-based transformation models treat the relation r as a transformation from the head entity h to the tail entity t. TransE [11], which depicts a hyperplane transformation, is an early representative of this type of model. For each triple (h, r, t) in the KG, the existential approximation equation $h+r\approx t$ is used to express it. Some subsequently derived column models have generally tried to project the representation of entities and relations into other spaces to overcome the disadvantage of the TransE [9] model in dealing with complex relations such as many-to-many, one-to-many, and many-to-one relations. Examples of these models include TransH [35]; TransR [36]; TransD [37]; and the current state-of-the-art embedding model RotatE [13], which defines each relation as a rotation from head entities to tail entities in a complex space. An example of the ontology-based model is TransO [19], which can effectively model relationships explicitly and seamlessly to improve the model performance and keep the model complexity low.

A tensor-based decomposition model is a model that decomposes knowledge representation learning into three tensors. Models such as these usually reduce high-dimensional KG data into low-dimensional tensors, making complex problems simple. An example of this model is RESCAL [38], which uses vectors and matrices to represent entities and relations, respectively. Subsequently, a series of models based on tensor decomposition were derived: ComplEx [11], DistMult [10], and TuckER [14].

The neural network-based model includes the neural tensor network (NTN) model [16] and the convolutional neural network (CNN)-based typical models ConvE [17] and ConvKB [18]. Among them, NTN [16] is a vector embedding that projects the entity to the input layer. The embeddings of the head and tail entities are then combined through a specific tensor and used as an input to the non-linear layer to compute the score. CNNs have many advantages, such as efficient parameters and fast training. Owing to these excellent properties, they have been widely used in KGEs. ConvKB [18] improves the state-of-the-art model by using other CNNs, capturing global relations and knowledge between entities, while ConvE [17] uses convolutional feature filters to filter reshaped matrices of feature and relation embeddings.

2.2. GNN-based KGC model

Unlike traditional graph embedding models such as Node2Vec [39] and DeepWalk [40], GNNs use deep neural networks for representation learning under the single-node aggregation approach for intent GNNs. The core formulation of GNNs is shown in Table 1.

GCN [20] is a GNN that has produced prominent results. It aggregates messages from one-hop neighbors by performing local convolutional operations on graph structure data, as shown in the equation in Table 1. In a GCN, all the neighbors pass messages of equal weight. Therefore, inspired by the attention mechanism in Transformer [41], many applications based on the multi-head attention mechanism were born. For example, using the most popular GAT in GNNs, the local attention mechanism was introduced into GCNs and very good results were obtained [21,42].

Since the introduction of relational-GCNs (R-GCNs) [43], GNNs have been used in KGEs to address the limitations of traditional neural networks; their structures are limited to processing traditional Euclidean data. Also, the R-GCN introduced the GCN and utilized it for processing multi-relational data in KGs. Then, the weighted GCN was introduced using the searching architecture calibration network (SACN) [45]; this GCN defined the strength of two neighboring nodes with the same relation type and captured structured information by using node structures, node features, and relation types. Fig. 2 shows the encoder–decoder framework

Table 1
Graph neural network and message functions.

GNN		GNN-based KGC-model	
GNNs	Node aggregation	Models	Message function
GCN [20]	$H^{(l+1)} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$	R-GCN [43]	$W_r e_h$
		VR-GCN [44]	$W\left[(e_h - e_r) \text{ or } (e_h + e_r)\right]$
GAT [21]	$\vec{h}_i' = \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij} W \vec{h}_j\right)$	SACN [45]	$W\alpha_r e_h$
		CompGCN [46]	$W_{\mathrm{dir}(r)}\left(e_h \star e_r\right)$

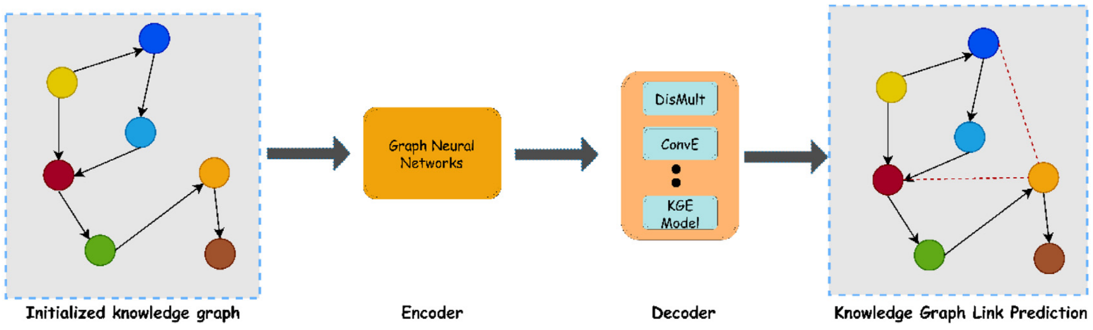

Fig. 2. Encoder–decoder framework of GCN-based KGC [25].

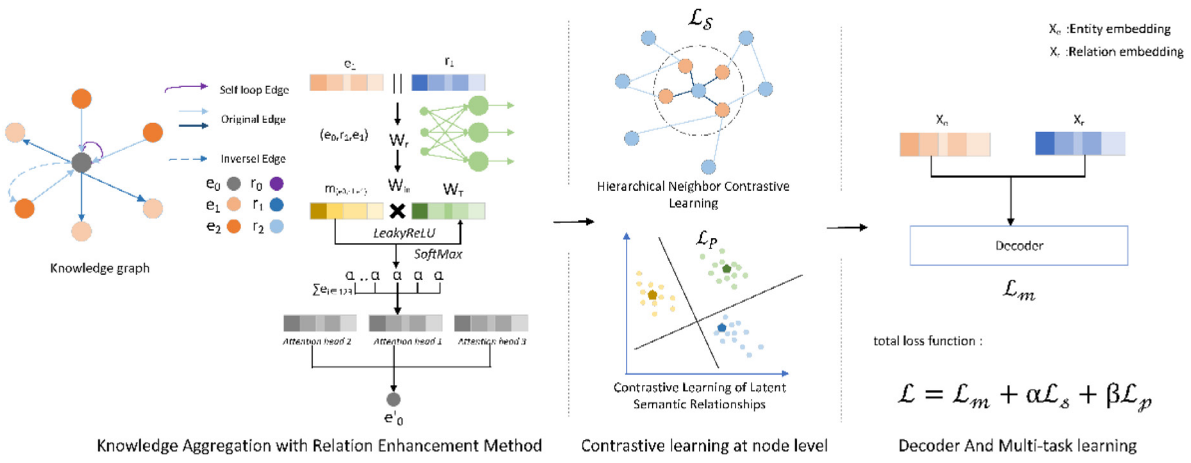
Self loop Edge
Original Edge
Inversel Edge
Knowledge graph
LeakyReLU
SoftMax
Attention head 1
Attention head 2
Attention head 3
Knowledge Aggregation with Relation Enhancement Method
$\mathcal{L}_S$
Hierarchical Neighbor Contrastive Learning
$\mathcal{L}_P$
Contrastive Learning of Latent Semantic Relationships
Contrastive learning at node level
X_e :Entity embedding
X_r :Relation embedding
Decoder
$\mathcal{L}_m$
total loss function :
$\mathcal{L} = \mathcal{L}_m + \alpha\mathcal{L}_s + \beta\mathcal{L}_p$
Decoder And Multi-task learning

Fig. 3. Proposed model with three attention heads and one attention layer. The embeddings of the entities and relations are updated, the representations of the entities and relations are updated through CL, and finally, the encoder is used to calculate the score.

of the GCN-based KGC. Here, the GCN is an encoder that generates entity and relation representations based on the graph structure; and the KGE model is a decoder that recovers the KG structure based on the entity and relation representations generated by the GCN. In the figure, the partially invisible links predicted by the links are indicated by undirected red dashed lines.

The latest and most influential KGC model that uses the encoder–decoder architecture is CompGCN [46], which was proposed to jointly learn node embeddings and relational representations in a multi-relational graph and solve the parameter overload problem that existed in previous work on multi-relational graph representation (GNN aspect). Almost all the current mainstream GNN-based KGC models follow the encoder–decoder architecture shown in Fig. 2 [25].

3. CLGAT-KGC model

In this section, we present our proposed CLGAT-KGC model. We have first introduced the general framework of the model, followed by the relation-enhanced entity representation module, the CL module, and the decoder layer scoring module.

3.1. Overall architecture

The architectural process of our proposed CLGAT-KGC model is shown in Fig. 3. The overall process is shown in Algorithm 1. The architecture and workflow of the model is shown from left to right in Fig. 3. First, the entire triple of the KGs was taken as the input. Each value in the triad was embedded into a continuous vector space while preserving the semantic information. To fully capture the higher-order structural information of the KG, we used a graph attention-based embedding to propagate the information approach and used our proposed method that worked on relationally enhanced entity representation to obtain the aggregated entity representation.

After obtaining the entity and relation representations in the GNN layer, we inputted the initialized representations of the entities and new representations to the CL layer based on the hierarchy of the neighbors. Then, we inputted the new representations to the potential semantic relation-based layer and partitioned them into K clusters for contrastive learning to enhance the knowledge representation. Finally, we inputted the entity and relation representations to our decoder layer to obtain the scores of the link prediction.

3.2. Methods for relation-enhanced entity representation

Through our model, we propose a relation-enhanced entity representation in a multi-relational setting based on the three KG information flow directions followed by CompGCN [46], namely original, self-loop, and inverse. We defined the KG as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where ε denotes the set of entities, R denotes the set of relations, and T denotes the set of edges. Each edge (h, r, t) indicates that the head entity h to tail entity t is real in the case of the relation $r \in R$. We used concepts introduced under CompGCN. Our same extension extended ε and R with the homologous self-loop and inverse relations, i.e., $\mathcal{T}' = \mathcal{T} \cup \{h, r^{-1}, t \mid (h, r, t) \in \{\mathcal{T}\}\} \cup \{h, \top, h \mid h \in \mathcal{E}\}$, and $R' = R \cup R_{inv} \cup \{\top\}$, where $\top$ and $R_{inv} = \{r^{-1} \mid r \in R\}$ represent the self-loop and inverse relations, respectively. The right side of Table 1 shows how some common models use message functions to represent learning information from the neighbor nodes and edges. If the head entity h is connected to the tail entity t through the relation r, the combination is represented as Eq. (1):

$$c_{(h,r,t)} = \phi\left(e_h, e_r, e_t\right) \tag{1}$$

Consistent with the approach followed under CompGCN and unlike most approaches that embed nodes only in the graph, our model simultaneously learned the entity embedding representation of the d-dimensional representation; $e_h, e_t \in R^d$ and the d-dimensional relation is expressed as $e_r \in R^d$. Additionally, $\phi(\cdot)$ represents a combinatorial operator, which serves to merge relational embeddings into entity embeddings in some way. We used the combination of entities and relations $c_{(h,r,t)}$ to generate the delivery message, as shown in Eq. (2):

$$m_{(h,r,t)} = M\left(c_{(h,r,t)}\right) \tag{2}$$

where $m_{(h,r,t)}$ denotes the messages from the head entity h through the relation r to the tail entity t; and $M(\cdot)$ denotes the filters for the three different message flow directions (to be described later). As we stated in Section 1, we propose that when an entity is connected to different relations, it is not represented exactly alike; we wanted different relations to have different effects on the central entity t. To implement this type of message passing, we extracted the relational features for augmenting the entity representation by introducing a learnable parameter w_r for each relation. The message passing formula under a particular relation is shown in Eq. (3):

$$c^r_{(h,r,t)} = \phi_r\left(e_h, e_r, e_t, w_r\right) \tag{3}$$

To reduce the computational complexity and memory consumption, we defined w_r as a diagonal matrix, $w_r \in R^{d_0 \times 1}$.

As followed under CompGCN, we used w_r for three combinatorial operators for message functions, inspired by TransE [9], DistMult [10], and HolE [11]. Our augmentation of each operator is shown here.

(1) Subtraction (Sub): $\Phi_r\left(e_h, e_r, e_t, w_r\right) = w_r e_h - e_r$
(2) Multiplication (Mult): $\Phi_r\left(e_h, e_r, e_t, w_r\right) = \left(w_r e_h\right) \oplus e_r$, where $\oplus$ represents the Hadamard product.
(3) Circular-correlation (Corr): $\Phi_r\left(e_h, e_r, e_t, w_r\right) = \left(w_r e_h\right) \star e_r$, where $\star$ stands for the cyclic operator operation in HolE [12].

Inspired by the approach under CompGCN, we defined a separate filter for each of the three different directions of the original, inverse, and self-loop edges in Eq. (4):

$$m_{(h,r,t)} = M\left(c_{(h,r,t)}\right) = w_{\text{dir}(r)} c^r_{(h,r,t)} \tag{4}$$

where the weights of the three different directional edges are defined as follows:

$$\mathbf{w}_{\text{dir}(r)} = \begin{cases} \mathbf{w}_O & \text{if } r \in \mathcal{R} \\ \mathbf{w}_I & \text{if } r \in \mathcal{R}_{\text{inv}} \\ \mathbf{w}_S & \text{if } r \in \{\top\} \end{cases} \tag{5}$$

The terms O, I, and S in the above Eq. (5) represent the original, inverse, and self-loop directions, respectively.

Typically, traditional GCN models model the message update function for obtaining the embedding representation after node t is updated, as shown in Eq. (6):

$$e'_t = f\left(\sum_{(h,r)\in\mathcal{N}(t)} W m_{(h,r,t)}\right) \tag{6}$$

where f represents the nonlinear activation function, and $N(t)$ represents the set of entities and relations that are directly adjacent to the central entity t. Based on the parameters of the GCN model being summed using a fixed weight parameter W, we introduced an attention mechanism to achieve a different attention level for each message. Like in GATs [21], we used a single layer feedforward neural network to implement the attention mechanism module and defined the weight matrix as $w_T \in R^{1\times d_1}$. We used Leaky ReLu as the activation function, as shown in Eq. (7):

$$b_{h,r} = \text{LeakyReLU}\left(\mathbf{W}_{\mathrm{T}} \mathbf{m}_{(h,r,t)}\right) \tag{7}$$

where $b_{h,r}$ represents the absolute attention coefficient of each message from entity h to entity t. We used the softmax function on the message attention $b_{h,r}$ to obtain the relative attention values in Eq. (8):

$$\alpha_{h,r} = \text{softmax}\left(b_{h,r}\right) = \frac{\exp\left(b_{h,r}\right)}{\sum_{i\in\mathcal{N}(t)} \sum_{r\in\mathcal{R}_{i,h}} \exp\left(b_{i,r}\right)} \tag{8}$$

where $R_{i,h}$ denotes the set of relations connecting the entities h and i. To obtain the output after the entity update, we obtained the linear combination of the normalized attention coefficients with the message function in Eq. (9):

$$e'_t = f\left(\sum_{(h,r)\in\mathcal{N}(t)} \alpha_{h,r} m_{(h,r,t)}\right) \tag{9}$$

Inspired by the approach of the NLP domain transformer [41], we also employed multi-headed attention, enabling the model to capture subspace parameters from different relational parameters for stabilizing and improving the learning and effectiveness of the model, respectively. We used the averaging method for N independent attention heads and chose TanH as the activation function whose output entity t embedding is represented as Eq. (10):

$$e'_t = f\left(\frac{1}{N}\sum_{h=1}^{N} \sum_{(h,r)\in\mathcal{N}_t} \alpha^h_{h,r} m^h_{(h,r,t)}\right) \tag{10}$$

To use the same dimensionality in the decoder layer for embedding the entities and relations, we propose using a weight matrix $W_t \in R^{d_1 \times d}$ for transforming the relational dimension in Eq. (11):

$$e'_r = W_t e_r \tag{11}$$

where W_t is a learnable weight matrix for transforming the embedding space of the relations into the same as that of the entities.

Algorithm 1 Training process for link prediction task in our model

```
Input: Knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$; training set $\mathcal{S} = (h, r, t) | h, t \in \mathcal{E}, r \in \mathcal{R}$;
batch size b; temperature parameter $\tau$;
number of clusters K, regularization parameter $\lambda$;
learning rate $\alpha$; message function $\phi(\cdot)$; and score function $\psi(\cdot)$.
Output: Triplet prediction probability $\hat{y}(\cdot)$.
1  Relation-enhanced entity-representation module:
2  Initialize node embedding $e_h, e_t \in R^d, \forall h, t \in \mathcal{E}$.
3  Initialize relation embedding $e_r \in R^d, \forall r \in \mathcal{R}$.
4  for $n$ = 1 to $N$ do
5      //$N$ is the number of attention heads
6      for $r \in \mathcal{R}$ do
           for $t \in \mathcal{E}$ do
               $c_{(h,r,t)} = \phi(e_h, e_r, e_t)$; //Incorporating relational embeddings into entity embeddings
               using combinators $\phi(\cdot)$
               $c^r_{(h,r,t)} = \phi_r(e_h, e_r, e_t, w_r)$; //Entity-enhanced representation under a specific relation
               $m_{(h,r,t)} = w_{dir(r)} c^r_{(h,r,t)}$; //Messaging under a specific relation
               Eq. (12), (13), and (14)// Computational relative attention mechanism
           $e'_r = W_t e_r$ //Transformation of embedding space of relations into the same one as that
           of the entities
       $e'_t = f\left(\frac{1}{N}\sum_{h=1}^{N} \sum_{(h,r)\in\mathcal{N}_t} \alpha^h_{h,r} m^h_{(h,r,t)}\right)$ //Multi-head attention mechanism for averaging
7  $\boldsymbol{E} \leftarrow$ concat $\{\boldsymbol{e}'_t, \forall t \in \mathcal{E}\}$
8  $\boldsymbol{R} \leftarrow$ concat $\{\boldsymbol{e}'_r, \forall r \in \mathcal{R}\}$
9  Node-level contrastive learning module:
10 The loss function based on hierarchical neighbors $\mathcal{L}_S$ can be obtained from Section 3.3 and
   Eq. (12), (13), and (14).
11 The loss function based on latent semantics $\mathcal{L}_P$ can be obtained from Section 3.3 and Eq.
   (15).
12 Decoder and training process of link prediction task:
13 for $S_{batch} \leftarrow$ sample$(\mathcal{T}, b)$ do
       for $(s, r, o) \in \mathcal{T}_{batch}$ do
           for $o' \in \mathcal{E}$ do
               $\mathcal{T}'_{\text{batch}} \leftarrow (s, r, o') \wedge (o', r, s) \notin \mathcal{T}$ ;
               // Generate the negative triplets set.
           $S_{\text{batch}} \leftarrow S_{\text{batch}} \cup \{\mathcal{T}_{\text{batch}}, \mathcal{T}'_{\text{batch}}\}$ ;// Generate the trained triplets set.
14 From Eq. (21), $\mathcal{L}_m$ is obtained as the main loss function for the task.
15 From Eq. (22), $\mathcal{L}$ is obtained as the total loss function for the task.
16 Backpropagation is conducted and the parameters are updated by Adam optimizer.
17 Repeat: Training is repeated to get best predicted value.
18 Until: converges
19 Return: Triplet prediction probability $\hat{y}(\cdot)$.
```

3.3. Node-level contrastive learning modules

Inspired by recent CL-based applications in computer vision, CL is developing quickly in various fields. A common tool used regarding the components of GNNs is graph augmentation, which both adds edges to and subtracts them from graphs [47–49]. We propose two node-level CL methods for KGC. The first one is a hierarchical-neighbor based CL approach. The second one is a CL method based on potential semantic relations.

3.3.1. CL approach based on hierarchical neighbors

We want to further improve the representation of entity and relation embeddings by CL; so, we propose enabling entity/relation learning through contrasts with their hierarchical neighboring layers whose representations are propagated through GNN layers to aggregate messages. We used the initialized feature embeddings or learnable embedding representations of entities as a positive contrast to their own embedding representations after K GNN layer inputs were used to improve the entity-relation representations by closing the distance between the two positive

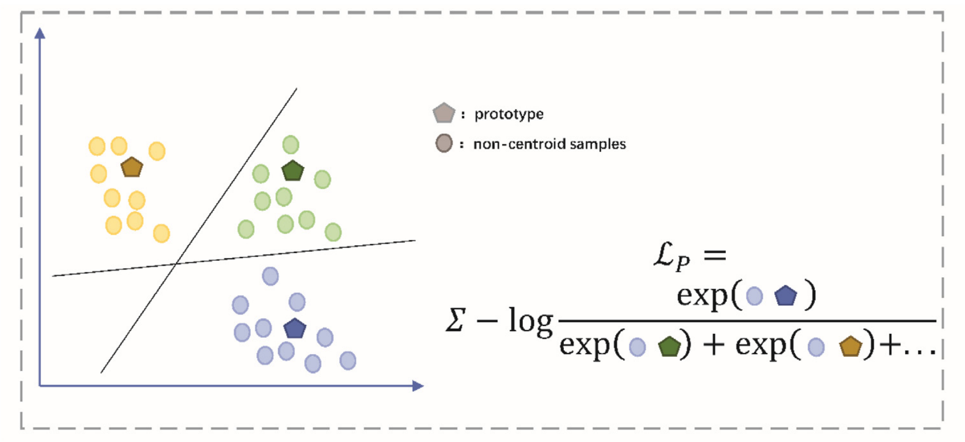
: prototype
: non-centroid samples
$\mathcal{L}_P =$
$\Sigma - \log \frac{\exp(\cdot)}{\exp(\cdot) + \exp(\cdot) + \ldots}$

Fig. 4. Overview of CL of potential semantic relations.

samples, which we define as the loss formula (Eq. (12)):

$$\mathcal{L}_S^E = \sum_{e \in \mathcal{E}} -\log \frac{\exp\left(\left(\mathbf{z}_e^{(k)} \cdot \mathbf{z}_e^{(0)} / \tau\right)\right)}{\sum_{v \in \mathcal{E}} \exp\left(\left(\mathbf{z}_e^{(k)} \cdot \mathbf{z}_v^{(0)} / \tau\right)\right)} \tag{12}$$

where $Z_e^{(k)}$ is the entity representation of the output after the introduction of the Kth layer of the GAT and normalization; and τ is the temperature parameter, which is an important hyperparameter in CL.

Similarly, the initialized embedding of the relation was used as a positive contrast with its own transformed embedding, as shown in Eq. (13):

$$\mathcal{L}_S^R = \sum_{r \in R} -\log \frac{\exp\left(\left(\mathbf{z}_r^{(k)} \cdot \mathbf{z}_r^{(0)} / \tau\right)\right)}{\sum_{j \in R} \exp\left(\left(\mathbf{z}_r^{(k)} \cdot \mathbf{z}_j^{(0)} / \tau\right)\right)} \tag{13}$$

The complete hierarchy-based neighborhood loss function is a weighted sum of the two loss functions mentioned, which is shown in Eq. (14):

$$\mathcal{L}_{\mathcal{S}} = \mathcal{L}_{\mathcal{S}}^{\mathcal{E}} + \Omega \mathcal{L}_{\mathcal{S}}^{\mathcal{R}} \tag{14}$$

where Ω is the weight that controls the entity-based contrast as well as the relation-based contrast.

3.3.2. *CL methods based on potential semantic relations*

The above hierarchy-based neighbor CL approach explicitly mines the hierarchical neighbor relations in the KG. This approach treats the hierarchical neighbors of entity nodes and relations equally, which inevitably introduces noise. To reduce the impact of this noise on the hierarchical-neighbor comparison and further explore the potential semantic relations between entity nodes to improve the entity node representation. We then propose a comparison based on potential semantic relations to extend the hierarchical contrast. Potential semantic relations are entity nodes that may not be reachable in the KG but have similar features at the semantic level.

Inspired by [30], w e could identify and mine potential semantic relations by learning each entity representation and generalizing them using CL to better capture the potential semantic relations between entity nodes. Usually, entity nodes with similar features tend to be represented in the adjacent embedding space. Therefore, we applied the clustering algorithm to the embedding representation of the entity nodes after GNN initialization. There are many new clustering algorithms [50]; for convenience, we used the classic K-means clustering algorithm. It was used to cluster all the entity nodes into different classes, allowing us to apply CL very well. In the K-means clustering algorithm, for each category that is divided, there is a cluster center, which is called the prototype. Within a category, the cluster center is the positive sample, while the other cluster centers are the negative samples of the nodes, as shown in Fig. 4. The contrastive learning loss function is shown in Eq. (15):

$$\mathcal{L}_P = \sum_{u \in \mathcal{E}} -\log \frac{\exp\left(\mathbf{e}_u \cdot \mathbf{c}_i / \tau\right)}{\sum_{\mathbf{c}_j \in C \neq i} \exp\left(\mathbf{e}_u \cdot \mathbf{c}_j / \tau\right) + \exp\left(\mathbf{e}_u \cdot \mathbf{c}_i / \tau\right)} \tag{15}$$

where C_i is the prototype of the entity set $\mathcal{E}$, which is obtained by performing K-means clustering on the embedding of all the entity nodes. K clusters were obtained after the clustering algorithm was performed to cover all the entity nodes.

3.4. *Introduction to the decoder layer*

In our study, we used three different decoders to verify the validity of our model, namely DistMult [10], ConvE [17], and InteractE [46].

DistMult was obtained by restricting the relation matrix M_r in the Rescal [38] model to a diagonal matrix, which reduced the number of parameters of the bilinear model to the same as that of TransE. Its scoring function is given by Eq. (16):

$$g_r^b\left(y_{e_1}, y_{e_2}\right) = y_{e_1}^T M_r y_{e_2} \tag{16}$$

where $y_{e1} = f(Wx_{e1})$, $y_{e2} = f(Wx_{e2})$ are the head and tail entities, respectively.

ConvE is one of the most common decoders used for computing the evaluation probabilities of triples. ConvE models the interactions between input entities and relations through convolutional and fully connected layers. First, it reshapes the embeddings of head and tail entities into a two-dimensional tensor, and then, applies the standard convolution operation to the reshaped tensor to compute the score of the triple. Given a triple (h,r, t), ConvE has the following triplet scoring function shown in Eq. (17):

$$p_{(h,r,t)} = \mathrm{ReLU}\left(\mathrm{vec}\left(\mathrm{ReLU}\left(\left[\mathbf{e_h}; \mathbf{e_r}\right] * \omega\right)\right) \mathbf{W}\right) \mathbf{e_t} \tag{17}$$

where the vec($\cdot$) operation flattens the tensor into a vector.

InteractE is an improved version of the ConvE model. It enhances the expression of ConvE through feature permutation, checkered feature reshaping, and circular convolution. For the input (e_h, e_r), a random permutation is first generated, as shown in Eq. (18):

$$\mathcal{P}_n = \left[\left(e_h^1, e_r^1\right), \ldots, \left(e_h^n, e_r^n\right)\right] \tag{18}$$

Table 2
Dataset statistics for FB15K-237 and WN18RR.

Dataset	#entity	#relation	#training	#valid	#test
FB15K-237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3034	3134

Then, InteractE performs the checkered feature reshaping operation $\phi_{chk}(e_h, e_r)$, ∀i ∈ {1, …, n}, as shown in Eq. (19):

$$\phi_{chk}\left(\mathcal{P}_n\right) = \left[\phi_{chk}\left(e_h^1, e_r^1\right), \ldots, \phi_{chk}\left(e_h^n, e_r^n\right)\right] \tag{19}$$

The implementation formula for evaluating the probability of a real triple (h, r, t) implemented using InteractE can be written formally as Eq. (20):

$$p_{(h,r,t)} = g\left(\text{vec}\left(f\left(\phi_{chk}\left(\mathcal{P}_t\right) \circledast \boldsymbol{\omega}\right)\right)\mathbf{W}\right)\mathbf{e}_t \tag{20}$$

where $vec(\cdot)$ denotes the tilting of the tensor into a vector; and $\circledast$ represents circular convolution in the depth direction; ω denotes the convolution filter; $\mathbf{W}$ is a weight matrix; and f and g represent the ReLu and Sigmoid activation functions, respectively.

To train our model, we used the cross-entropy loss function optimized for label smoothing as the master function for our task, as shown in Eq. (21):

$$\mathcal{L}_m = -\frac{1}{N}\sum_i \left(t_i \cdot \log\left(p_i\right) + \left(1 - t_i\right) \cdot \log\left(1 - p_i\right)\right) \tag{21}$$

where t_i denotes the label of the triplet; and p_i denotes the corresponding score.

In summary, our model can be described as a multi-task learning approach, and the total loss of the model tasks is defined in Eq. (22):

$$\mathcal{L} = \mathcal{L}_m + \alpha\mathcal{L}_s + \beta\mathcal{L}_p \tag{22}$$

where α and β are used to control the weights between the losses of multi-task learning.

4. Experiments

To verify the validity of our proposed model, we conducted many experiments and reported the detailed analytical results. Then, for the link prediction task, we evaluated the inference capability of our model, compared it with those of baseline models, analyzed the sensitivity of the model parameters, and finally verified the validity of our model by further analysis through ablation experiments.

4.1. Experimental setup

4.1.1. Datasets

We evaluated our model using two commonly used datasets in the KG domain, namely WN18RR and FB15K-237; the corresponding statistics are shown in Table 2. WN18 and FB15K [9] were pointed out in later work [17,51]; these datasets had the drawback of test set leakage. The WN18RR and FB15K-237 datasets were released after the inverse relations were eliminated. Our experimental dataset does not include the WN18 and FB15K datasets.

The WN18RR dataset comprises 41K grammar sets from WordNet and 11 relational collections. FB15K-237 contains 15K entities and 237 relations from Freebase. Table 2 shows the detailed descriptions of these two datasets.

4.1.2. Evaluation metrics

The evaluation protocol for the link prediction task was the same as that followed under CompGCN, where a ranking criterion was used for evaluation. For each test triple (h, r, t), we randomly replaced h and t with all the entities in the dataset to generate a corrupted triple as our negative sample for score calculation. The input triad was then used by the decoding layer to calculate the score of its triads and rank them in descending order. We obtained the ranking of the correct triad among the candidate triad entities. Like mainstream baselines, we reported the experimental results in the "filter" setting, as in TransE; we removed the corrupted triples already present in the dataset during the ranking. Our evaluation metrics were Mean Reciprocal Ranking (MRR), Mean Rank (MR) and Hits@n (n = 1, 3, 10).

The specific calculation formula for MRR is Eq. (23):

$$\mathbf{MRR} = \frac{1}{|S|}\sum_{i=1}^{|S|}\frac{1}{\text{rank}_i} = \frac{1}{|S|}\left(\frac{1}{\text{rank}_1} + \frac{1}{\text{rank}_2} + \cdots + \frac{1}{\text{rank}_{|S|}}\right) \tag{23}$$

where S is the set of triples, $|S|$ is the number of triple sets, and $rank_i$ is the link prediction rank of the ith triple. The larger the MRR value, the better the model is.

The specific calculation formula of MR is Eq. (24):

$$\mathbf{MR} = \frac{1}{|S|}\sum_{i=1}^{|S|}\text{rank}_i = \frac{1}{|S|}\left(\text{rank}_1 + \text{rank}_2 + \cdots + \text{rank}_{|S|}\right) \tag{24}$$

The symbols in the MR and MRR formulas represent the same variables. The smaller the MR value is, the better the model is.

Hits@n refers to the proportion of correct entities ranked in the top N in the link prediction, which is calculated in Eq. (25):

$$\text{HITS@n} = \frac{1}{|S|}\sum_{i=1}^{|S|}\mathbb{I}\left(\text{rank}_\text{i} \leqslant \text{n}\right) \tag{25}$$

where II(·) is the indicator function. If the condition is true, then the function value is 1; otherwise, it is 0. The larger the value of this indicator, the better the model is.

4.1.3. Training protocol and comparison of models

We used Pytorch [52] and the Adam [53] optimizer to implement our model and performed experiments on a PC server with a Nvidia RTX 3090 GPU. The final hyperparameters of our model were selected by grid search, which were determined based on the combined MRR and MR metrics evaluated on the validation set. Some of the important hyperparameters that performed well for our link prediction task on the FB15K-237 and WN18RR datasets are shown in Table 3.

To demonstrate the effectiveness of our proposed model for the link prediction task under the domain of KGC, we used the models proposed in the following papers as baselines.

(1) TransE [9]: The earliest KGC model.
(2) DistMult [10]: A tensor decomposition-based KGC model, which uses a bilinear scoring function approach for calculating the scores of triples.
(3) RotatE [13]: The model follows a new idea, defining each relation as a rotation from the source entity to the target entity in the complex vector space.
(4) ConvE [17]: A classical KGC model based on CNNs.
(5) SACN [45]: The model uses an end-to-end graph structure-sensitive convolutional network, which can preserve the translation properties between entities to relational embeddings based on the working of ConvE.

Table 3
Hyperparameters that performed well on both datasets.

Dataset → Hyperparameters ↓	FB15K-237	WN18RR
Learning rate	0.001	0.0005
Epoch	1000	1000
GCN_drop	0.4	0.4
Batch_size	1024	256
Entity_embedding	200	200
Relation_embedding	200	200
GNN_layer	1	1
nheads	2	1
Ω	0.5	0.5
α	0.4	0.4
β	0.6	0.6

(6) InteractE [54]: An extension of ConvE that adds interaction between entities and relational embeddings.
(7) R-GCN [43]: An advanced extension to GCN that efficiently models multi-relational data.
(8) A2N [55]: A novel model for learning the query-related representations of entities based on GNN structures.
(9) MRGAT [56]: The model can selectively aggregate information features and perform feature weighting completely. The learned entities and relational embeddings can then be used for downstream tasks such as KGC.
(10) HRAN [57]: Each relational path-based feature is aggregated with the learned weight values to generate an embedding representation to accomplish the KGC task.
(11) LTE [25]: The model proposes a framework for equipping existing KGE models with linearly transformed entity embeddings.
(12) DeepER [58]: The model utilizes rotation and reflection transformations of group convolutions to generate more expressive feature maps for entities and relations.
(13) CompGCN [46]: The model uses a new CNN framework and embeds the representation of nodes and relations together in the relational graph.

4.2. Results and analysis

4.2.1. Performance comparison on link prediction

In this subsection, we present a comprehensive evaluation and analysis of our proposed model, comparing its performance on two datasets with five evaluation metrics with those of the baseline models. Table 4 summarizes the experimental results of our proposed model under three different decoder layers, namely InteractE, ConvE, and DistMult; and compares it with the experimental results of the baseline models. The experimental results of the baseline models are taken from [46,57], and the respective source papers of the models. The best results are depicted in bold, and the suboptimal results are underlined.

Compared to CompGCN [46], our CLGAT-KGC-InteractE model made greater progress on the FB15K-237 dataset; the MRR improved by 5.3%, MR improved by 18.7%, Hits@1 improved by 3.4%, Hits@3 improved by 2.5%, and Hits@10 improved by 2.9%. For the WN18RR dataset, MRR improved by 1.2%, MR improved by 40%, Hits@1 improved by 0.4%, Hits@3 improved by 1.2%, and Hits@10 improved by 3.6%. Additionally, our model showed excellent results compared to the remaining 12 baseline models, with a 1%–2% improvement in most metrics compared to the baselines.

For the FB15K-237 dataset, we obtained optimal results for four of the five metrics and suboptimal results for one. For the WN18RR dataset, we obtained optimal results for three metrics and suboptimal results for one. Generally, we found that our model performed better on FB15K-237 wherein the number of relations was more, while an embedding-based model like RotatE [13] performed better for Hits@10 on WN18RR wherein the number of relations was less. Although we obtained good results on the WN18RR dataset for Hits@10, our results were still slightly inferior to those under RotatE. We attributed this to the main relation patterns in WN18RR being symmetric/anti-symmetric and inversed, whereas in WN18RR, almost all the entities were words and belonged to the same entity type, making them more suitable for the RotatE model. The effect of our model on FB15K-237 for the MR indicator and on WN18RR for the Hits@1 indicator was very different from that of HRAN, showing that the new heterogeneous GNN inspired by HRAN also had special effects on a very few indicators.

The general validity of our model was demonstrated after experiments on both datasets. Our proposed model had considerable advantages over the distance-based embedding model and GCN-based CompGCN model. Compared with models such as A2N [55], MRGAT [56] and HRAN [57] that also used attention to aggregate messages, our model had better experimental results and more effectively demonstrated the effectiveness of the proposed relation enhancement method and CL method.

4.2.2. Comparison of different message functions and different decoders

In this subsection, we compare the different results of our model on two datasets based on five metrics under different message functions and different decoder layers for evaluating the effectiveness of our model. To more precisely verify the effectiveness of different decoder and message functions, we used ConvE, DistMult, and InteractE as our decoders. Similarly, we used three message functions, namely sub, corr, and mult, which were consistent with CompGCN, to conduct experiments at each decoder layer separately. The experimental results of different decoder layers and their corresponding message functions are shown in Table 5. The best results are depicted in bold, and the suboptimal results are underlined.

Table 5 shows the specific and clear statistical results of the experiments. The results of different message functions and decoder layers on the performance of the models can be observed.

In the decoder layer, our model had the best overall effect when using InteractE as the decoder. ConvE was in second place, while DistMult had a large gap with the previous two decoder models. This also showed that the CNN-based model could be better adapted to our model than the tensor decomposition-based decoder model could. This moderately indicated that the more advanced decoder layer could achieve better results for the GNN-based KGC model.

In terms of message functions, the simplest message function sub provided better results in both datasets when InteractE was used as the decoder. When using ConvE and DistMult as the decoder, the more complex corr function was often used to achieve better results. The results revealed that the message function should neither be too complex nor simple. The message function should be selected for the model according to the specific situation to obtain better experimental results.

4.2.3. Ablation experiments

To explore the contribution of each module, we conducted an ablation experiment for our model. In the experiments, we used InteractE as the decoder and corr as the message function. The experimental results under the FB15K-237 and WN18RR datasets are shown in Tables 6 and 7, respectively.

We deleted the potential semantic relation CL module, the hierarchical neighbor CL module, and the relation-enhanced entity representation module, layer by layer. Tables 6 and 7 show

Table 4
Experimental results of proposed model performing link prediction on three decoder layers and two datasets.

Model	FB15K-237					WN18RR				
	MRR↑	MR↓	Hits@1	Hits@3	Hits@10	MRR↑	MR↓	Hits@1	Hits@3	Hits@10
TransE [9]	0.294	357	–	–	0.465	0.226	3384	–	–	0.501
DistMult [10]	0.241	254	0.155	0.263	0.419	0.430	5110	0.390	0.440	0.490
RotatE [13]	0.358	177	0.241	0.375	0.533	0.476	3340	0.428	0.492	**0.571**
SCAN [45]	0.35	–	0.26	0.39	0.54	0.47	–	0.43	0.48	0.54
ConvE [17]	0.325	244	0.237	0.356	0.501	0.430	4187	0.400	0.440	0.520
R-GCN [43]	0.248	–	–	–	0.417	–	–	–	0.137	–
InteractE [54]	0.354	172	0.263	–	0.535	0.463	5202	0.430	–	0.528
A2N [55]	0.317	–	0.232	0.348	0.486	0.450	–	0.420	0.460	0.510
HRAN [57]	0.355	**156**	0.263	0.390	0.541	0.479	2113	**0.450**	0.494	0.542
MRGAT [56]	0.355	–	0.266	0.392	0.539	0.481	–	0.449	0.495	0.544
LTE [25]	0.355	249	0.264	0.389	0.535	0.472	3434	0.437	0.485	0.544
DeepER [58]	0.345	–	0.255	0.379	0.525	0.476	–	0.446	0.490	0.535
CompGCN [46]	0.355	197	0.264	0.390	0.535	0.479	3533	0.443	0.494	0.546
Ours-InteractE	**0.364**	160	**0.274**	**0.402**	**0.551**	**0.484**	**2104**	0.448	**0.550**	0.566
Ours-ConvE	0.362	165	0.269	0.399	0.549	0.483	2411	0.447	0.495	0.556
Ours-DistMult	0.348	192	0.255	0.382	0.530	0.450	2823	0.409	0.460	0.535

Table 5
Comparison of experimental results under different decoders.

Decoder+message	FB15K-237					WN18RR				
function	MRR↑	MR↓	Hits@1	Hits@3	Hits@10	MRR↑	MR↓	Hits@1	Hits@3	Hits@10
InteractE+sub	0.364	160	**0.274**	**0.402**	**0.551**	0.484	**2104**	0.448	0.500	**0.566**
InteractE+corr	**0.365**	171	0.273	0.400	0.550	0.486	2252	**0.449**	0.504	0.562
InteractE+mult	0.363	176	0.270	0.399	0.546	**0.487**	2393	0.447	**0.505**	0.560
ConvE+sub	0.361	**156**	0.268	0.397	0.547	0.477	2322	0.431	0.485	0.555
ConvE+corr	0.362	165	0.269	0.399	0.549	0.483	2411	0.447	0.495	0.556
ConvE+mult	0.363	163	0.270	0.399	0.546	0.471	2925	0.426	0.490	0.554
DistMult+sub	0.338	195	0.248	0.370	0.517	0.438	3366	0.398	0.453	0.523
DistMult+corr	0.348	192	0.255	0.382	0.530	0.450	2823	0.409	0.460	0.535
DistMult+mult	0.344	199	0.254	0.390	0.527	0.453	2967	0.410	0.468	0.539

Table 6
Results of ablation experiments under FB15K-237 dataset.

Perspective detail	MRR↑	MR↓	Hits@1	Hits@3	Hits@10
·Full model	0.365	160	0.274	0.402	0.551
·w/0 Potential semantic relations	0.364	190	0.272	0.399	0.549
·w/0 Potential semantic relations ·w/0 Hierarchical neighbor	0.363	197	0.271	0.398	0.548
·w/0 Potential semantic relations ·w/0 Hierarchical neighbor ·w/0 Relation enhancement method	0.361	207	0.269	0.397	0.545
CompGCN [46]	0.355	197	0.264	0.390	0.535

Table 7
Results of ablation experiments under the WN18RR dataset.

Perspective detail	MRR↑	MR↓	Hits@1	Hits@3	Hits@10
·Full model	0.486	2252	0.449	0.504	0.562
·w/0 Potential semantic relations	0.485	2398	0.448	0.500	0.561
·w/0 Potential semantic relations ·w/0 Hierarchical neighbor	0.483	2418	0.448	0.500	0.556
·w/0 Potential semantic relations ·w/0 Hierarchical neighbor ·w/0 Relation enhancement method	0.483	2495	0.448	0.495	0.553
CompGCN [46]	0.479	3533	0.443	0.494	0.546

the corresponding contributions of each module. Among them, the potential semantic relation-based module provided better MR values; we assumed it improves the model's ability to rank unseen triples with potential relations. The other modules had different degrees of improvement for each indicator. The experimental results in Tables 6 and 7 clearly show the effectiveness of

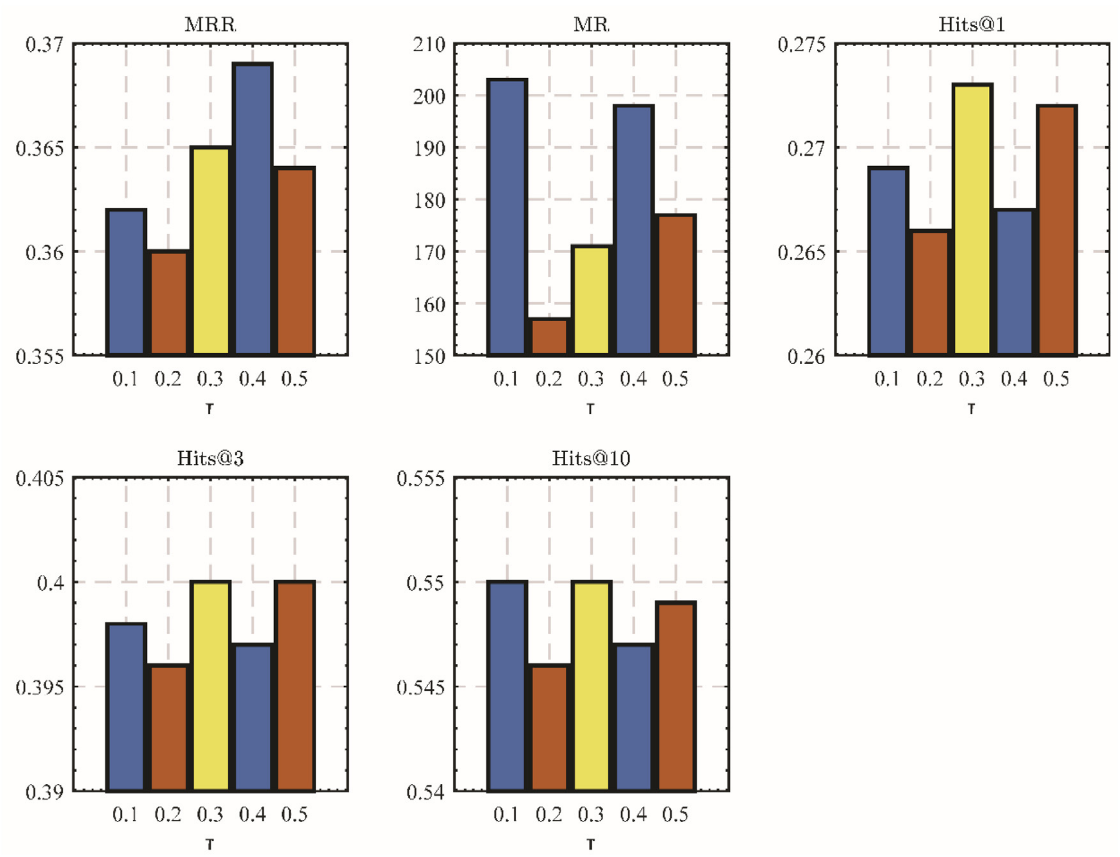

Fig. 5. Parameter sensitivity of different τ values on FB15K-237 dataset.

our model. The ablation experiments also proved that embedding the CL module between the encoder and decoder layers of our model could perfectly fit the link prediction task and improve the experimental effect of the model, confirming the applicability and effectiveness of our proposed introduction of CL into the field of KGC.

4.2.4. *Parameter sensitivity*

We conducted extensive experiments on the FB15K-237 dataset to investigate the parameter sensitivity of three important parameters in our model.

(1) Impact of temperature τ: The temperature parameter mainly regulated the degree of attention to difficult samples [59]. Considering the more extreme two cases, when τ tended to 0, the contrast loss function degenerated to a loss function that focused only on difficult negative samples, while when τ tended to infinity, all the negative samples were treated equally under the contrast loss and the focus property of difficult negative samples was lost. To verify which τ was more suitable, we chose $\tau = 0.1$, 0.2, 0.3, 04, and 0.5; the experimental results are shown in Fig. 5. Our model had the best results when $\tau = 0.3$. This also reaffirmed the principle that τ should neither be too large nor too small [59].

(2) Impact of prototype number K: To check the impact of the aggregation category number K on the model, we conducted experiments using different aggregation numbers. The experimental results are shown in Fig. 6. The results showed that the model had the best effect when $K = 20$, and the model effect decreased to different degrees when K was larger or smaller than 20. This was mainly because too many aggregated categories would increase the noise of the model, while too few of them would be insufficient to explore the potential semantic relations in the KG.

(3) Impact of hierarchical-neighbor based CL loss weight α and potential semantic relation-based CL loss weight β:To investigate the importance of the two CL losses in our multitask learning, we set several proportional weights to control the importance of the two CL losses. The results are shown in Fig. 7. The model worked best when α: $\beta = 0.4$: 0.6.

4.2.5. *Depth of network layers L*

To check the effect of network layers L, we explored the performance of the proposed method with different depths. The results are shown in Fig. 8. The results showed that our model had the best performance when the depth of the GNN layers was 1. As the number of network layers increased, the experimental results became worse. This was mainly because when the number of GNN layers increased, the aggregated features would become too smooth, degrading the performance of the model.

4.2.6. *Multi-head attention mechanism*

As shown in Fig. 9, we used 1, 2, and 3 attention heads on the FB15K-237 and WN18RR datasets. The graph for the FB15K-237 dataset has a sharp angle shape, and the best results were obtained when the number of attention heads was 2. For the WN18RR dataset, the experimental results deteriorated as the number of attention heads increased, and the best results were obtained when there was 1 attention head.

5. Conclusion and future work

In this study, we propose a KGC model based on CL and the relation-enhanced entity representation method for link prediction under KGC. The model could obtain different representations

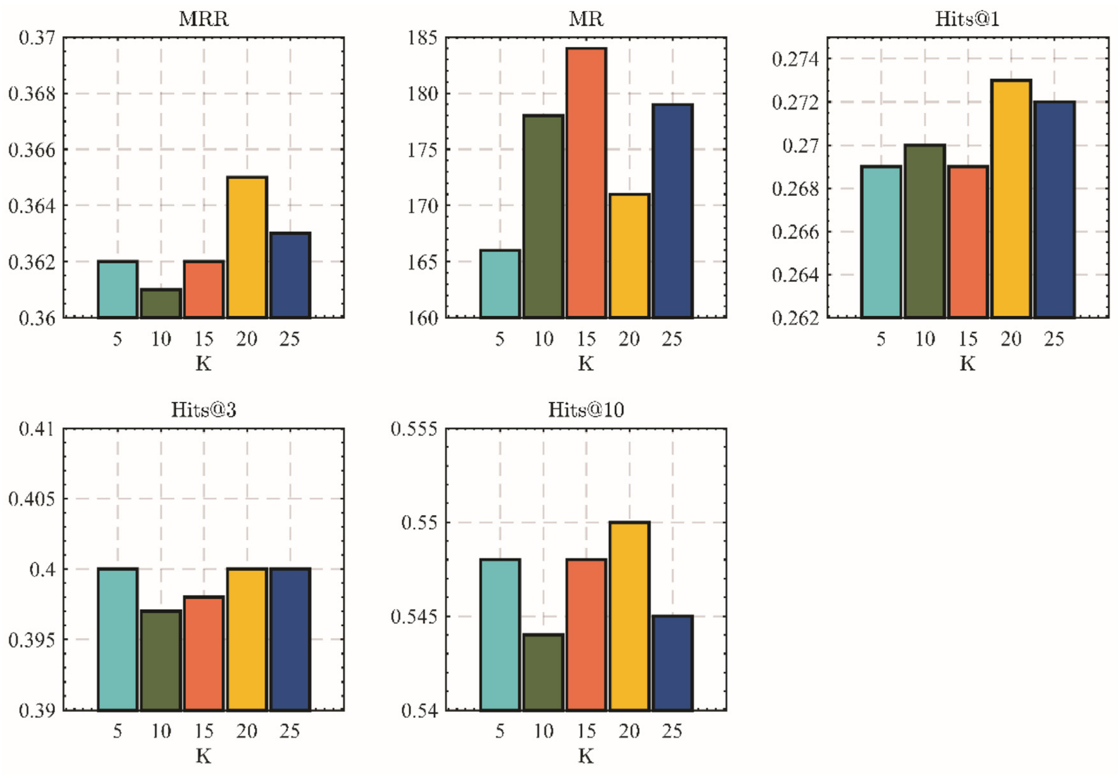

Fig. 6. Parameter sensitivity of different K values on FB15K-237 dataset.

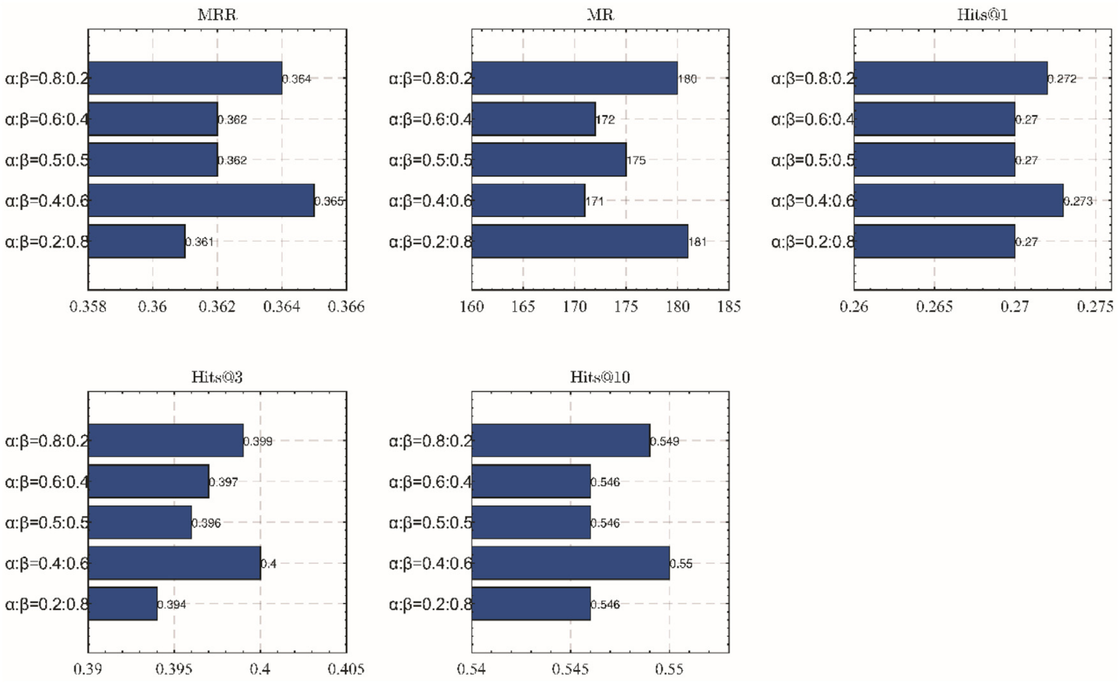

Fig. 7. Parameter sensitivity of different α and β values on FB15K-237 dataset.

for entities when the same entity was combined with different relations, and the representation of the entities and relations was further improved after using CL at the node level. The experimental analysis of our model on both FB15K-237 and WN18RR

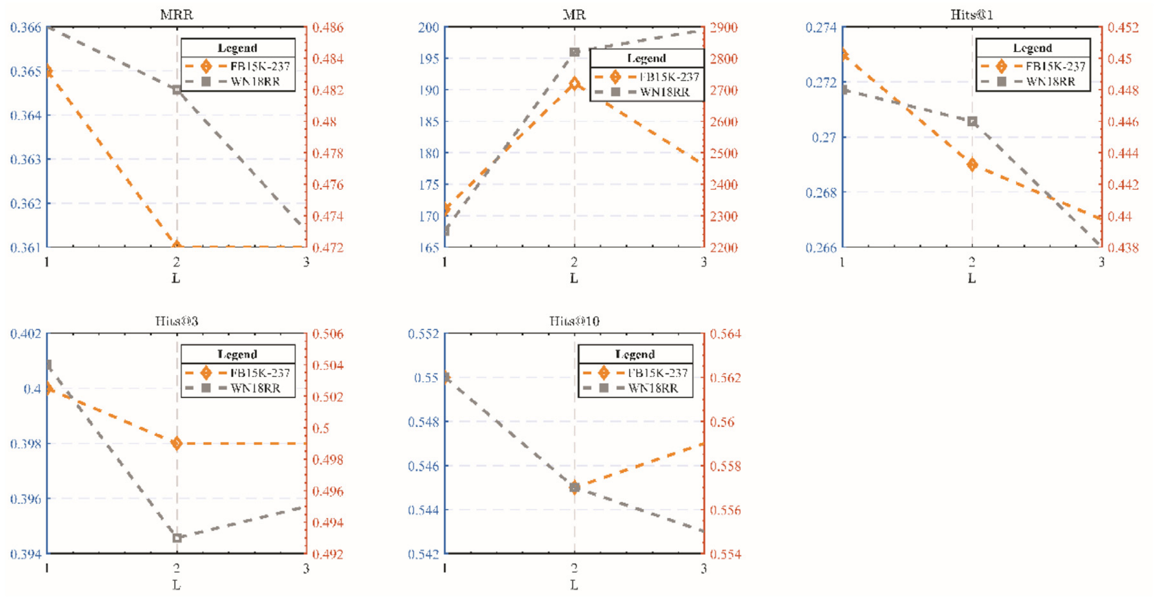

Fig. 8. Experimental results of proposed model on FB15K-237 and WN18RR datasets for different depths of network layers.

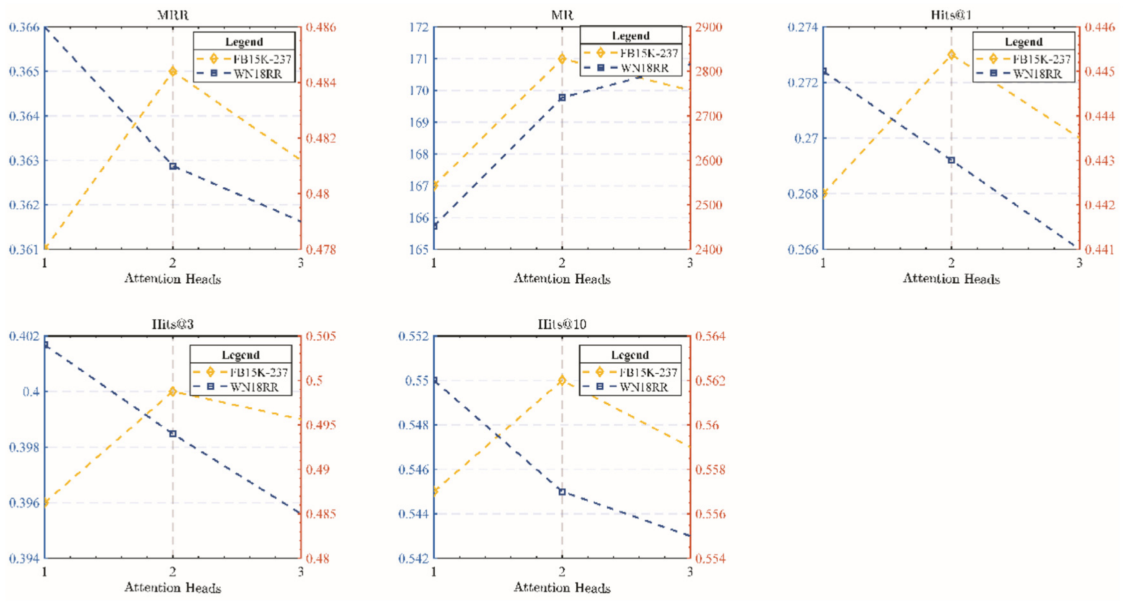

Fig. 9. Experimental results of proposed model on FB15K-237 and WN18RR datasets when multihead attention mechanism is implemented using different numbers of heads.

datasets with multiple decoder layers and multiple message functions showed that our model could achieve better results in link prediction tasks. In the future, we intend to design a more efficient graph CL approach and more improved GNNs for extending our model.

CRediT authorship contribution statement

LinYu Li: Conceptualization, Methodology, Software, Writing – original draft. **Xuan Zhang:** Supervision, Writing – review & editing. **YuBin Ma:** Visualization, Investigation. **Chen Gao:** Investigation. **Jishu Wang:** Writing – review & editing. **Yong Yu:** Writing – review & editing. **Zihao Yuan:** Writing – review & editing. **Qiuying Ma:** Writing – review & editing.

Declaration of competing interest

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.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 61862063, 61502413, 61262025; the National Social Science Foundation of China under Grant No. 18BJL104; the Science Foundation of Young and

Middle-aged Academic and Technical Leaders of Yunnan, China under Grant No. 202205AC160040; the Science Foundation of Yunnan Jinzhi Expert Workstation, China under Grant No. 202205AF150006; the Open Foundation of Yunnan Key Laboratory of Software Engineering, China under Grant No. 2020SE301; and the Science Foundation of the "Knowledge-driven intelligent software engineering innovation team, China".

References

[1] S. Ji, S. Pan, E. Cambria, P. Marttinen, S.Y. Philip, A survey on knowledge graphs: Representation, acquisition, and applications, IEEE Trans. Neural Netw. Learn. Syst. 33 (2021) 494–514, http://dx.doi.org/10.1109/TNNLS.2021.3070843.

[2] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D.S. Weld, A. Yates, Web-scale information extraction in knowitall: (preliminary results), in: Proceedings of the 13th International Conference on World Wide Web, New York, USA, 2004, pp. 100–110. http://dx.doi.org/10.1145/988672.988687.

[3] F.M. Suchanek, G. Kasneci, G. Weikum, Yago: A core of semantic knowledge, in: Proceedings of the 16th International Conference on World Wide Web, Banff, Canada, 2007, pp. 697–706. http://dx.doi.org/10.1145/1242572.1242667.

[4] J. Hoffart, F.M. Suchanek, K. Berberich, G. Weikum, YAGO2: A spatially and temporally enhanced knowledge base from wikipedia, Artificial Intelligence 194 (2013) 28–61, http://dx.doi.org/10.1016/j.artint.2012.06.001.

[5] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, DBpedia: A nucleus for a web of open data, in: The Semantic Web, Springer, Berlin, 2007, pp. 722–735, http://dx.doi.org/10.1007/978-3-540-76298-0_52.

[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, 2008, Vancouver, Canada, pp. 1247–1250. http://dx.doi.org/10.1145/1376616.1376746.

[7] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka, T.M. Mitchell, Toward an architecture for never-ending language learning, in: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, 2010, Atlanta, USA, pp. 1306–1313.

[8] W. Wu, H. Li, H. Wang, K.Q. Zhu, Probase: A probabilistic taxonomy for text understanding, in: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, Scottsdale, USA, 2012, pp. 481–492. http://dx.doi.org/10.1145/2213836.2213891.

[9] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, Adv. Neural Inf. Process. Syst. 26 (2013).

[10] B. Yang, W.-T. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, 2014, http://dx.doi.org/10.48550/arXiv.1412.6575, arXiv preprint arXiv:1412.6575.

[11] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: Proceedings of the 33rd International Conference on Machine Learning, PMLR, New York, USA, 2016, pp. 2071–2080.

[12] M. Nickel, L. Rosasco, T. Poggio, Holographic embeddings of knowledge graphs, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA, Vol. 30, 2016. http://dx.doi.org/10.1609/aaai.v30i1.10314.

[13] Z. Sun, Z.-H. Deng, J.-Y. Nie, J. Tang, RotatE: Knowledge graph embedding by relational rotation in complex space, 2019, http://dx.doi.org/10.48550/arXiv.1902.10197, arXiv preprint arXiv:1902.10197.

[14] I. Balažević, C. Allen, T.M. Hospedales, TuckER: Tensor factorization for knowledge graph completion, 2019, http://dx.doi.org/10.48550/arXiv.1901.09590, arXiv preprint arXiv:1901.09590.

[15] S. Amin, S. Varanasi, K.A. Dunfield, G. Neumann, LowFER: Lowrank bilinear pooling for link prediction, in: Proceedings of the 37th International Conference on Machine Learning, PMLR, 2020, pp. 257–268.

[16] R. Socher, D. Chen, C.D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, Adv. Neural Inf. Process. Syst. 26 (2013).

[17] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2D knowledge graph embeddings, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, USA, Vol. 32, 2018. http://dx.doi.org/10.1609/aaai.v32i1.11573.

[18] D.Q. Nguyen, T.D. Nguyen, D.Q. Nguyen, D. Phung, A novel embedding model for knowledge base completion based on convolutional neural network, 2017, http://dx.doi.org/10.48550/arXiv.1712.02121, arXiv preprint arXiv:1712.02121.

[19] Z. Li, X. Liu, X. Wang, P. Liu, Y. Shen, TransO: a knowledge-driven representation learning method with ontology information constraints, World Wide Web (2022) 1–23, http://dx.doi.org/10.1007/s11280-022-01016-3.

[20] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, Adv. Neural Inf. Process. Syst. 29 (2016).

[21] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, Stat 1050 (2017) 20.

[22] Y. Yang, Z. Guan, J. Li, W. Zhao, J. Cui, Q. Wang, Interpretable and efficient heterogeneous graph convolutional network, IEEE Trans. Knowl. Data Eng. (2021) http://dx.doi.org/10.1109/TKDE.2021.3101356.

[23] X. Song, J. Li, Y. Tang, T. Zhao, Y. Chen, Z. Guan, JKT: A joint graph convolutional network based deep knowledge tracing, Inform. Sci. 580 (2021) 510–523, http://dx.doi.org/10.1016/j.ins.2021.08.100.

[24] G. Stoica, O. Stretcu, E.A. Platanios, T. Mitchell, B. Póczos, Contextual parameter generation for knowledge graph link prediction, in: Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, New York, USA, Vol. 34, 2020, pp. 3000–3008. http://dx.doi.org/10.1609/aaai.v34i03.5693.

[25] Z. Zhang, J. Wang, J. Ye, F. Wu, Rethinking graph convolutional networks in knowledge graph completion, in: Proceedings of the ACM Web Conference 2022, Lyon, France, 2022, pp. 798–807. http://dx.doi.org/10.1145/3485447.3511923.

[26] X. Chen, K. He, Exploring simple siamese representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15750–15758. http://dx.doi.org/10.1109/CVPR46437.2021.01549.

[27] K. He, H. Fan, Y. Wu, S. Xie, R. Girshick, Momentum contrast for unsupervised visual representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9729–9738. http://dx.doi.org/10.1109/CVPR42600.2020.00975.

[28] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: Proceedings of the 37th International Conference on Machine Learning, PMLR, 2020, pp. 1597–1607.

[29] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, X. Xie, Self-supervised graph learning for recommendation, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 726–735. http://dx.doi.org/10.1145/3404835.3462862.

[30] S. Lin, P. Zhou, Z.-Y. Hu, S. Wang, R. Zhao, Y. Zheng, L. Lin, E. Xing, X. Liang, Prototypical graph contrastive learning, 2021, http://dx.doi.org/10.48550/arXiv.2106.09645, arXiv preprint arXiv:2106.09645.

[31] Y. Yang, C. Huang, L. Xia, C. Li, Knowledge graph contrastive learning for recommendation, 2022, http://dx.doi.org/10.48550/arXiv.2205.00976, arXiv preprint arXiv:2205.00976.

[32] Y. Yan, R. Li, S. Wang, F. Zhang, W. Wu, W. Xu, ConSERT: A contrastive framework for self-supervised sentence representation transfer, 2021, http://dx.doi.org/10.48550/arXiv.2105.11741, arXiv preprint arXiv:2105.11741.

[33] T. Gao, X. Yao, D. Chen, SimCSE: Simple contrastive learning of sentence embeddings, 2021, http://dx.doi.org/10.48550/arXiv.2104.08821, arXiv preprint arXiv:2104.08821.

[34] J. Liao, X. Zhao, X. Li, J. Tang, B. Ge, Contrastive heterogeneous graphs learning for multi-hop machine reading comprehension, World Wide Web 25 (2022) 1469–1487, http://dx.doi.org/10.1007/s11280-021-00980-6.

[35] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Proceedings of the Twenty-Eight AAAI Conference on Artificial Intelligence, Québec, Canada, Vol. 28, 2014, pp. 1112–1119. http://dx.doi.org/10.1609/aaai.v28i1.8870.

[36] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, Texas, USA, 2015, pp. 2181–2187.

[37] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, 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 (Volume 1: Long Papers), Beijing, China, 2015, pp. 687–696. http://dx.doi.org/10.3115/v1/P15-1067.

[38] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: Proceedings of the 28th International Conference on Machine Learning, Bellevue, Washington, United States, 2011, pp. 809–816.

[39] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, USA, 2016, pp. 855–864. http://dx.doi.org/10.1145/2939672.2939754.

[40] B. Perozzi, R. Al-Rfou, S. Skiena, DeepWalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, USA, 2014, pp. 701–710. http://dx.doi.org/10.1145/2623330.2623732.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[42] J. Li, H. Zeng, L. Peng, J. Zhu, Z. Liu, Learning to rank method combining multi-head self-attention with conditional generative adversarial nets, Array 15 (2022) 100205, http://dx.doi.org/10.1016/j.array.2022.100205.

[43] M. Schlichtkrull, T.N. Kipf, P. Bloem, R.v.d. Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: European Semantic Web Conference, Springer, Cham, 2018, pp. 593–607, http://dx.doi.org/10.1007/978-3-319-93417-4_38.

[44] R. Ye, X. Li, Y. Fang, H. Zang, M. Wang, A vectorized relational graph convolutional network for multi-relational network alignment, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macau, 2019, pp. 4135–4141. http://dx.doi.org/10.24963/ijcai.2019/574.

[45] Y.-C. Wu, F. Yin, X.-Y. Zhang, L. Liu, C.-L. Liu, SCAN: Sliding convolutional attention network for scene text recognition, 2018, http://dx.doi.org/10.48550/arXiv.1806.00578, arXiv preprint arXiv:1806.00578.

[46] S. Vashishth, S. Sanyal, V. Nitin, P. Talukdar, Composition-based multi-relational graph convolutional networks, 2020, http://dx.doi.org/10.48550/arXiv.1911.03082, arXiv preprint arXiv:1911.03082.

[47] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, L. Wang, Deep graph contrastive representation learning, 2020, http://dx.doi.org/10.48550/arXiv.2006.04131, arXiv preprint arXiv:2006.04131.

[48] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, L. Wang, Graph contrastive learning with adaptive augmentation, in: Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 2021, pp. 2069–2080. http://dx.doi.org/10.1145/3442381.3449802.

[49] X. Song, J. Li, Q. Lei, W. Zhao, Y. Chen, A. Mian, Bi-CLKT: Bigraph contrastive learning based knowledge tracing, Knowl. Based Syst. 241 (2022) 108274, http://dx.doi.org/10.1016/j.knosys.2022.108274.

[50] C. Xu, Z. Guan, W. Zhao, H. Wu, Y. Niu, B. Ling, Adversarial incomplete multi-view clustering, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, Macau, 2019, pp. 3933–3939. http://dx.doi.org/10.24963/ijcai.2019/546.

[51] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, M. Gamon, Representing text for joint embedding of text and knowledge bases, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 2015, pp. 1499–1509. http://dx.doi.org/10.18653/v1/D15-1174.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, Adv. Neural Inf. Process. Syst. 32 (2019).

[53] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, http://dx.doi.org/10.48550/arXiv.1412.6980, arXiv preprint arXiv:1412.6980.

[54] S. Vashishth, S. Sanyal, V. Nitin, N. Agrawal, P. Talukdar, InteractE: Improving convolution-based knowledge graph embeddings by increasing feature interactions, in: Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, New York, USA, Vol. 34, 2020, pp. 3009–3016. http://dx.doi.org/10.1609/aaai.v34i03.5694.

[55] T. Bansal, D.-C. Juan, S. Ravi, A. McCallum, A2N: Attending to neighbors for knowledge graph inference, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 2019, pp. 4387–4392. http://dx.doi.org/10.18653/v1/P19-1431.

[56] Z. Li, Y. Zhao, Y. Zhang, Z. Zhang, Multi-relational graph attention networks for knowledge graph completion, Knowl. Based Syst. 251 (2022) 109262, http://dx.doi.org/10.1016/j.knosys.2022.109262.

[57] Z. Li, H. Liu, Z. Zhang, T. Liu, N.N. Xiong, Learning knowledge graph embedding with heterogeneous relation attention networks, IEEE Trans. Neural Netw. Learn. Syst. 33 (2021) 3961–3973, http://dx.doi.org/10.1109/TNNLS.2021.3055147.

[58] A. Zeb, S. Saif, J. Chen, D. Zhang, Learning knowledge graph embeddings by deep relational roto-reflection, Knowl. Based Syst. 252 (2022) 109451, http://dx.doi.org/10.1016/j.knosys.2022.109451.

[59] F. Wang, H. Liu, Understanding the behaviour of contrastive loss, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, Tennessee, USA, 2021, pp. 2495–2504. http://dx.doi.org/10.1109/CVPR46437.2021.00252.