Contents lists available at ScienceDirect

Information Processing and Management

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

Knowledge graph representation learning with simplifying hierarchical feature propagation

Zhifei Li [a], Qi Zhang [b], Fangfang Zhu [c], Duantengchuan Li [d,*], Chao Zheng [d], Yan Zhang [a,*]

[a] *School of Computer Science and Information Engineering, Hubei University, Wuhan, Hubei 430062, China*
[b] *School of Information Management, Central China Normal University, Wuhan, Hubei 430072, China*
[c] *Faculty of Artificial Intelligence in Education, Central China Normal University, Wuhan, Hubei 430079, China*
[d] *School of Computer Science, Wuhan University, Wuhan, Hubei 430072, China*

ARTICLE INFO

Keywords:
Knowledge graphs
Representation learning
Knowledge graph embedding
Link prediction

ABSTRACT

Graph neural networks (GNN) have emerged as a new state-of-the-art for learning knowledge graph representations. Although they have shown impressive performance in recent studies, how to efficiently and effectively aggregate neighboring features is not well designed. To tackle this challenge, we propose the simplifying heterogeneous graph neural network (SHGNet), a generic framework that discards the two standard operations in GNN, including the transformation matrix and nonlinear activation. SHGNet, in particular, adopts only the essential component of neighborhood aggregation in GNN and incorporates relation features into feature propagation. Furthermore, to capture complex structures, SHGNet utilizes a hierarchical aggregation architecture, including node aggregation and relation weighting. Thus, the proposed model can treat each relation differently and selectively aggregate informative features. SHGNet has been evaluated for link prediction tasks on three real-world benchmark datasets. The experimental results show that SHGNet significantly promotes efficiency while maintaining superior performance, outperforming all the existing models in 3 out of 4 metrics on NELL-995 and in 4 out of 4 metrics on FB15k-237 dataset.

1. Introduction

Knowledge graphs (KGs), such as WordNet (Miller, 1995), NELL (Carlson et al., 2010), and Freebase (Bollacker et al., 2008), are used in a variety of downstream applications, including recommendation systems (Wang, He, et al., 2019; Wang, Zhang, Wang, et al., 2019; Wang, Zhang, Zhang, et al., 2019; Wu et al., 2022), question answering (Hao et al., 2018; Hu et al., 2018; Huang et al., 2019; Zhang, Weng et al., 2022). KGs are multi-relational graphs made up of (subject entity, relation, object entity) triplets. Although such triplets successfully store structured information, their underlying symbolic composition makes them difficult to manipulate by most machine learning methods. To address this problem, knowledge graph representation learning (KGRL) (Ji et al., 2022) (a.k.a., KG embedding (Li et al., 2022a; Wang et al., 2017; Zhang et al., 2022b)) has quickly gained intensive attention, which attempts to embed conceptual entities and relations into continuous vector spaces. The acquired features can then retain KGs' fundamental structure.

There is a significant challenge for KGs in predicting missing links (Chen et al., 2020; Li et al., 2022b; Sun et al., 2020). Recent years have witnessed massive studies on the link prediction problem and performed remarkable progress in KGRL (Rossi et al., 2021).

* Corresponding authors.
E-mail addresses: zhifei1993@hubu.edu.cn (Z. Li), dtclee1222@gmail.com (D. Li), zhangyan@hubu.edu.cn (Y. Zhang).

https://doi.org/10.1016/j.ipm.2023.103348
Received 13 November 2022; Received in revised form 3 February 2023; Accepted 13 March 2023
Available online 27 March 2023
0306-4573/© 2023 Elsevier Ltd. All rights reserved.

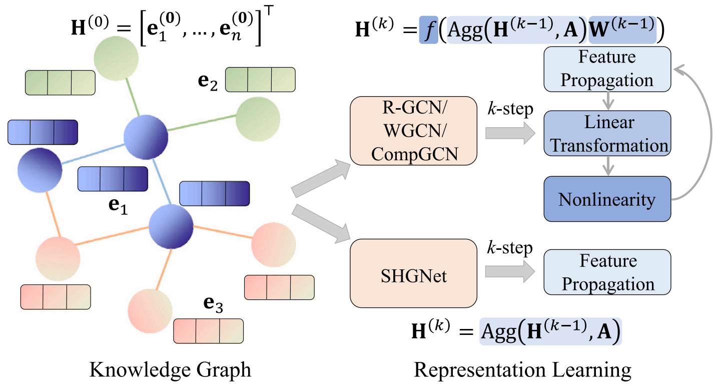
$\mathbf{H}^{(0)} = [\mathbf{e}_1^{(0)}, \ldots, \mathbf{e}_n^{(0)}]^{\top}$
$\mathbf{e}_2$
$\mathbf{e}_1$
$\mathbf{e}_3$
Knowledge Graph
$\mathbf{H}^{(k)} = f(\mathrm{Agg}(\mathbf{H}^{(k-1)}, \mathbf{A})\mathbf{W}^{(k-1)})$
R-GCN/ WGCN/ CompGCN
k-step
Feature Propagation
Linear Transformation
Nonlinearity
SHGNet
k-step
Feature Propagation
$\mathbf{H}^{(k)} = \mathrm{Agg}(\mathbf{H}^{(k-1)}, \mathbf{A})$
Representation Learning

Fig. 1. Knowledge graph representation learning with SHGNet and other models. SHGNet only keeps the feature propagation as opposed to redundant computation for other models.

The main categories of KGRL methods are translation distance-based models (e.g.: TransE (Bordes et al., 2013) and RotatE (Sun et al., 2019)), matrix factorization-based models (e.g.: RESCAL (Nickel et al., 2011) and DistMult (Yang et al., 2014)), and neural network-based models (e.g.: ConvE (Dettmers et al., 2018) and InteactE (Vashishth et al., 2020a)). Most previous models focused on representing knowledge triplets separately in KGs and ignored the heterogeneity of an entity's local neighborhood. To solve this issue, many studies (e.g.: R-GCN (Schlichtkrull et al., 2018), WGCN (Shang et al., 2019), and CompGCN (Vashishth et al., 2020b)) apply the graph neural network (GNN) (Wu et al., 2021) to model multi-relational KGs and achieve superior performance.

As shown in Fig. 1, it can conclude that previous GNN-based KGRL models follow the same propagation rule to learn the features: feature propagation, linear transformation, and nonlinear activation. However, most operations are directly inherited from GNN, which may generate unnecessary complexity and redundant computation. Inspired by the simplified GNN model such as SGC (Wu et al., 2019) and APPNP (Klicpera et al., 2019), this study is aimed at simplifying the design of GNN and making it more succinct and adaptable for KGRL. SGC and APPNP abandon nonlinear activation and linear transformation while maintaining feature propagation between GNN layers. However, SGC and APPNP ignore the feature propagation of relations which is not suitable for modeling KGs. Thus, it needs to design a simplified GNN framework for KGRL to achieve computation efficiently and incorporate the relation feature.

Furthermore, KGs generally contain multiple entities and relations and are thought to be heterogeneous information networks (HIN) (Shi et al., 2017). In general, various types of entities and relations have different attributes, which can be segmented into disparate semantic spaces. Moreover, We must gather valuable semantic characteristics using multiple relation types. Treating diverse relation types equally is unworkable since it weakens the semantic features gathered by certain essential relation types. As a result, we should explore the importance of each relationship type and give appropriate weights to them. Thus, it is critical to understand how to simultaneously combine numerous semantic knowledge sources and choose integrate informative features.

In light of the above discussion, it is advisable to consider the computation efficiency and graph heterogeneity in feature propagation. In the present study, a novel simplifying heterogeneous graph neural network (SHGNet) for KGRL is proposed, which could effectively and efficiently learn knowledge graph representation learning. In SHGNet, we remove the two standard operations including transformation matrix and nonlinear activation, which enables the model more effective and easier to train. Specifically, the proposed model incorporates the relation feature into the feature propagation formulation. And we design a hierarchical aggregation architecture for KGs to deal with the heterogeneity of local neighborhoods. It includes node aggregation to aggregate each relation type-based node feature. The features with different semantic information are then combined using relational weighting. Finally, the self-connection is utilized to aggregate the feature of the node itself. The acquired features are employed for numerous downstream tasks such as link prediction and can better model the complicated structure of KGs.

To summarize, our main contributions are set out as follows:

- We propose a simplified GNN-based KGRL model SHGNet, which designs the non-parametric feature propagation and only keeps the essential feature propagation operation. It makes our model easier to train and more effective.
- We design a hierarchical aggregation architecture for feature propagation in SHGNet. The presented method incorporates neighbor information via node aggregation and relation weighting, which can selectively aggregate informative features.
- We undertake extensive experiments to assess the effectiveness of the proposed methodology. The results on three benchmark datasets show that SHGNet has significant scalability while maintaining prediction quality.

The subsequent sections of the present paper are organized as follows. We first review the related work in Section 2. Then we present the details of the proposed model in Section 3. Section 4 presents the experimental results and analysis, and Section 5 concludes the paper.

2. Related work

Previous studies connected to our method, such as knowledge graph representation learning and graph neural networks, are discussed in this section.

2.1. Knowledge graph representation learning

The main objective of knowledge graph representation learning (KGRL), also known as Knowledge Graph Embedding (KGE), is to acquire the embedded representation of entities and relations. The success of the KGRL model based on linear or neural network operations has been noticed. These models are basically divided into three categories:

Relations are regarded as translations from a subject entity to an object entity in translation distance-based models. TransE (Bordes et al., 2013) and its variants, such as TransH (Wang et al., 2014), TransR (Lin et al., 2015), and TransD (Ji et al., 2015), are examples of representative translational models. Moreover, TorusE (Ebisu & Ichise, 2020) also keeps the translational characteristic of a lie group. RotatE (Sun et al., 2019) designs a rotation operation in complex vector space. PairRE (Chao et al., 2021) presents paired vectors for every relation representation to enable dynamic margin modification in loss function.

Matrix factorization-based models use linear or bilinear transformation procedures to compute the semantic similarity scores of entity–relation triplets. The typical models RESCAL (Nickel et al., 2011) and DistMult (Yang et al., 2014) encode interactions of entities and relations through a bilinear operation. ComplEx (Trouillon et al., 2016) is a DistMult extension that models asymmetric relations in complex vector space. TuckER (Balazevic et al., 2019) is a linear model that decomposes triplets into low-rank matrices for entity–relation feature representation. It is based on the TuckER tensor decomposition.

Entities and relations are utilized as input in neural network-based models for embedding and semantic matching modeling. The recent ConvE (Dettmers et al., 2018) is a neural network-based architecture that achieves state-of-the-art link prediction results by employing a multi-layer of 2D convolution over embeddings. ConvR (Jiang et al., 2019) creates convolutional features by building convolution filters adaptively using relation representations. ConvKB (Nguyen et al., 2018) investigates global linkages between same-dimensional entity and relation embeddings using a convolutional neural network. InteractE (Vashishth et al., 2020a) is a unique KGRL model that extends ConvE capabilities by capturing extra heterogeneous feature interactions. LTE-ConvE (Zhang, Wang, et al., 2022) enhances KGE models with linearly transformed entity embeddings and achieves comparable performance.

2.2. Graph neural network

Graph neural network (GNN) is designed to extend the deep neural networks to process data with arbitrary graphical structures. GCN (Kipf & Welling, 2017) first uses graph convolutional networks (GCN) to simplify graph convolutions. Several extensions have been developed in subsequent studies. FastGCN (Chen et al., 2018) is an efficient form of GCN that uses significance sampling to improve inductive node classification. On the basis of node sampling and a disparate aggregating mechanism, GraphSAGE (Hamilton et al., 2017) combines neighbor feature information. GAT (Velickovic et al., 2018) evaluates the attention values between neighbor nodes for node classification via the self-attention method (Vaswani et al., 2017). We recommend the reader to recent GNN surveys (Wu et al., 2021; Zhang, Cui, & Zhu, 2022; Zhang, Weng, et al., 2020) for a more comprehensive study.

To learn expressive knowledge graph representations, GNN has recently been presented to model KGs in an encoder–decoder framework. Representative GNN-based models include R-GCN (Schlichtkrull et al., 2018), WGCN (Shang et al., 2019), and CompGCN (Vashishth et al., 2020b). R-GCN introduces relation information into the modeling of GNNs and designs aggregation functions that incorporate relation information. WGCN adopts an aggregation function weighted the information of each subgraph to yield entity information. CompGCN designs a variety of information aggregation functions and implements a unified framework that generalizes multiple GNN-based modeling. However, most of them follow the same propagation rule to learn the features which leads to unnecessary complexity and redundant computation. Actually, it remains a challenge to effectively and efficiently aggregate KGs across large node features. Recently, some simplified GNN models such as SGC (Wu et al., 2019) and APPNP (Klicpera et al., 2019) have been proposed. SGC offers a way to reduce redundant complexity. It considers a nonlinearity transformation to be cumbersome and unnecessary and proposes an alternative to a single linear transformation. APPNP utilizes a propagation architecture built from modified PageRank to achieve linear computing cost in terms of edge number. LightGCN (He et al., 2020) attempts to simplify the structure of GNN in order to make it more concise and appropriate for recommendation. To modify pre-trained KG representations using graph context, REP-OTE (Wang et al., 2022) presents the relation-based embedding propagation approach. However, the aforementioned models concentrate on homogeneous graphs, it should develop a simplified heterogeneous GNN framework for KGs to achieve computation efficiently.

3. Methodology

The technical details of the proposed SHGNet are described in this section. First, the preliminaries and overall framework of SHGNet are introduced. Then the detailed procedures of SHGNet are presented. Finally, the loss function and training information are given. The main notations and explanations are presented in Table 1.

Table 1
Notations and explanations.

Notations	Explanations	Notations	Explanations
$\mathcal{G}$	Knowledge graph	$\mathcal{V}$	Set of nodes/entities
$\mathcal{E}$	Set of edges/triplets	$\mathcal{R}$	Set of relation types
e_s/e_o	Subject/object entity	r	Relation type
$\mathbf{e}$	Feature of entity	$\mathbf{r}$	Feature of relation
$\mathbf{H}$	Entity feature matrix	$\mathbf{R}$	Relation feature matrix
$\mathbf{A}$	Adjacency matrix	$\mathbf{D}$	Degree matrix
$\star$	Circular correlation	$\mathcal{N}$	Local neighbors
$\phi(\cdot)$	Score function	$*$	Convolution operation

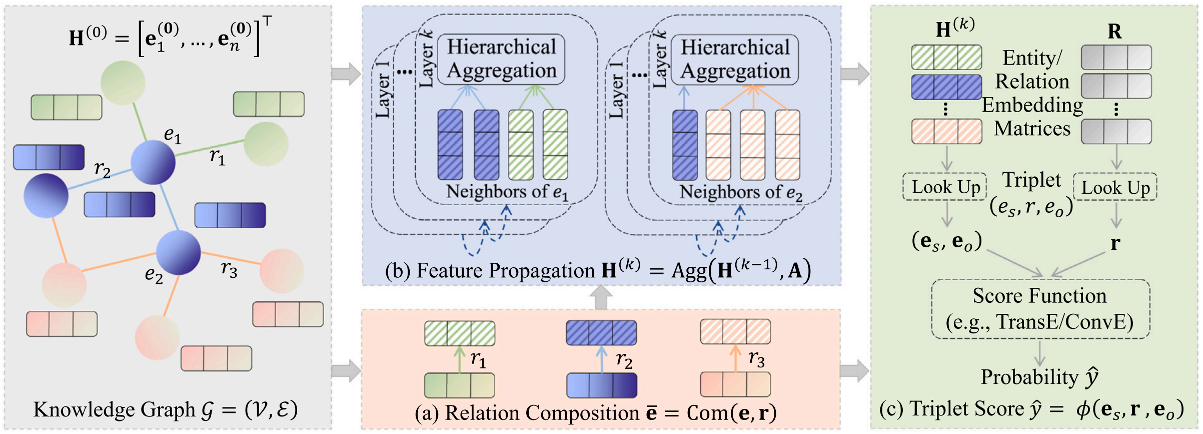
$\mathbf{H}^{(0)} = \left[\mathbf{e}_1^{(0)}, \dots, \mathbf{e}_n^{(0)}\right]^\top$
e_1
r_1
r_2
e_2
r_3
Knowledge Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Layer 1
Layer k
Hierarchical Aggregation
Neighbors of e_1
Neighbors of e_2
(b) Feature Propagation $\mathbf{H}^{(k)} = \text{Agg}(\mathbf{H}^{(k-1)}, \mathbf{A})$
(a) Relation Composition $\bar{\mathbf{e}} = \text{Com}(\mathbf{e}, \mathbf{r})$
$\mathbf{H}^{(k)}$
$\mathbf{R}$
Entity/ Relation Embedding Matrices
Look Up
Triplet (e_s, r, e_o)
$(\mathbf{e}_s, \mathbf{e}_o)$
$\mathbf{r}$
Score Function (e.g., TransE/ConvE)
Probability $\hat{y}$
(c) Triplet Score $\hat{y} = \phi(\mathbf{e}_s, \mathbf{r}, \mathbf{e}_o)$

Fig. 2. An illustration of SHGNet model architecture. Given entity and relation features, SHGNet adopts a composition operation to incorporate relation features into feature propagation. Then multi-layer simplifying hierarchical aggregation are utilized to fuse rich semantic information for KGs. Finally, a score function is performed to compute the triplet probability.

3.1. Problem formulation

Knowledge graphs (KGs) are directed graphs that could be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The node set is represented by the symbol $\mathcal{V}$, while the edge set is denoted by the symbol $\mathcal{E}$. The nodes represent entities, while the edges represent subject-relation-object triplet facts. Each edge is associated with a relation type $r \in \mathcal{R}$, where $\mathcal{R}$ represents the relation type set. An edge such as subject-relation-object (denoted as $(e_s, r, e_o) \in \mathcal{E}$) represents a relation r from subject entity e_s to object entity e_o. The bold letter $\mathbf{e}_s, \mathbf{r}, \mathbf{e}_o \in \mathbb{R}^d$ denotes their features with dimension d.

Simplified Feature Propagation. The goal of simplified feature propagation is to eliminate the requirement for feature transformation and nonlinear activation in GNN. Let $\mathbf{H}^{(k)}$ denote the node feature matrix, the equation $\mathbf{H}^{(k)} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(k-1)}$ denotes the forward propagation. However, the aforementioned technique is developed for handling homogeneous graph-structured data. A novel simplified feature propagation framework for heterogeneous KG is necessary.

Link Prediction for KGs. For KGs, link prediction is predicting another reasonable entity to compose a correct triplet. For instance, given the subject entity e_s and the relation r, predicting the object entity e_o. To achieve this goal, a general methodology is to define a score function $\phi(\mathbf{e}_s, \mathbf{r}, \mathbf{e}_o) \in \mathbb{R}$ for the triplet. The goal of the optimization is to rank correct triplets higher than incorrect triplets. In the real application area, link prediction with the proposed model can automatically predicting missing links for large-scale knowledge graphs. Thus, we can complete the knowledge graph and construct a larger knowledge graph.

3.2. Overall framework

An encoder–decoder architecture underpins the proposed SHGNet architecture. Fig. 2 depicts the SHGNet encoder workflow, which consists of two components: relation composition and feature propagation. In order to incorporate relation features into feature propagation, the encoder first leverages the entity–relation composition operation. Second, we adopt the simplifying hierarchical feature propagation for heterogeneous KGs, which can deal with the rich semantic information and complex structure effectively. Finally, the learned entity features are passed to the decoder, which could be modified by one of numerous KGRL models. In this paper, TransE (Bordes et al., 2013), DistMult (Yang et al., 2014), and ConvE (Dettmers et al., 2018) are chosen as the decoder.

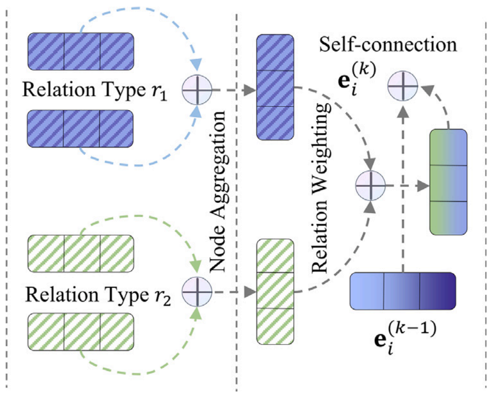

Fig. 3. An illustration of hierarchical aggregation architecture. SHGNet deals with heterogeneous node features in two steps including node aggregation and relation weighting. The initial step is to combine the features of each relation type-based node. Then the next step will aggregate features from different relation types. Furthermore, the self-connection operation is adopted to aggregate the entity feature itself.

3.3. *Relation composition*

Recently, GNN has proven to be highly effective at modeling graph-structured data. However, KGs are multi-relational graphs with edges that have entities and relations. Thus, it is necessary to design a novel GNN framework to systematically leverage entity–relation composition operations for KGRL. Specifically, SHGNet first initializes a d-dimensional feature representation $\mathbf{e} \in \mathbb{R}^d$ for entity and $\mathbf{r} \in \mathbb{R}^d$ for relation. To incorporate the relation feature into the feature propagation formulation, the entity–relation composition operation is introduced as follows:

$$\overline{\mathbf{e}} = \mathrm{Comp}(\mathbf{e}, \mathbf{r}), \forall r \in \mathcal{N}_e, \tag{1}$$

where $\mathrm{Comp}(\cdot)$ denotes the composition operator. We also limit the composition operator to be non-parameterized in this paper. Specifically, inspired by HolE (Nickel et al., 2016), the composition operator is denoted as follows:

$$\mathrm{Comp}(\mathbf{e}, \mathbf{r}) = \mathbf{e} \star \mathbf{r}, \tag{2}$$

where $\star : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ denotes *circular correlation*, and it can be rewritten as:

$$[\mathbf{e} \star \mathbf{r}]_k = \sum_{i=0}^{d-1} \mathbf{e}_i \mathbf{r}_{(k+i) \bmod d}. \tag{3}$$

In comparison to *addition* composition in TransE (Bordes et al., 2013) and *multiplication* composition in DistMult (Yang et al., 2014), the *circular correlation* composition has two main advantages. First, it is not commutative, i.e., $\mathbf{e} \star \mathbf{r} \neq \mathbf{r} \star \mathbf{e}$. Thus, the *circular correlation* composition is appropriate to model directed graphs for KGs. Second, the *circular correlation* composition is also a non-parameterized operation, which can further reduce the number of parameters for SHGNet.

3.4. *Feature propagation*

As shown in Fig. 3, distinct types of entities may exist in distinct feature spaces as a result of the heterogeneity of KGs. Directly incorporating all neighboring characteristics for every entity is not suitable. Thus, SHGNet proposes a hierarchical aggregation architecture for KGs. Specifically, SHGNet first aggregates each relation type-based node. Then, using relation weighting, features containing diverse semantic information are fused. Finally, the self-connection is utilized to aggregate the feature of the node itself. Moreover, we remove the transformation matrix and nonlinear activation to efficiently and effectively aggregate features.

The entity feature matrix is initialized as follows:

$$\mathbf{H}^{(0)} = \left[\mathbf{e}_1^{(0)}, \ldots, \mathbf{e}_n^{(0)}\right]^\top, \tag{4}$$

taking the ith entity feature $\mathbf{e}_i \in \mathbb{R}^d$ as the input, each relation type-based entity feature is first aggregated by SHGNet as follows:

$$\mathbf{e}_{\mathcal{N}_i^r}^{(k-1)} = \mathrm{Agg}\left(\mathbf{e}_j^{(k-1)}\right), \forall j \in \mathcal{N}_i^r, \tag{5}$$

where $\mathcal{N}_i^r$ indicates a group of entities based on relation types. $\mathrm{Agg}(\cdot)$ represents the aggregation function. The objective of this work is to enhance the effectiveness of feature propagation, so we choose the averaging operation for feature aggregation. It can also keep features on the same scale. Thus, the aggregation function is defined as:

$$\mathbf{e}_{\mathcal{N}_i^r}^{(k-1)} = \frac{1}{\left|\mathcal{N}_i^r\right|} \sum_{j \in \mathcal{N}_i^r} \mathbf{e}_j^{(k-1)}, \forall j \in \mathcal{N}_i^r, \tag{6}$$

Algorithm 1: Simplifying Heterogeneous Graph Neural Network Algorithm.

```
Input: The knowledge graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
       the set of relation types $\mathcal{R}$
       the dimension of feature $d$
       the number of network layers $K$
Output: The entity feature matrix $\mathbf{E}$ and the relation feature matrix $\mathbf{R}$
1  Initialize the node feature $\mathbf{e}_i^{(0)} \in \mathbb{R}^d,\ \forall e_i \in \mathcal{V}$
2  Initialize the relation feature $\mathbf{r} \in \mathbb{R}^d,\ \forall r \in \mathcal{R}$
3  for network layer k in [1,K] do
4      for each relation type $r$ in $\mathcal{R}$ do
5          for the i-th entity $e_i$ in $\mathcal{V}$ do
6              Execute the entity–relation composition operation $\mathrm{Comp}\left(\mathbf{e}_i, \mathbf{r}\right) = \mathbf{e}_i \star \mathbf{r}$
7              Aggregate each relation type-based entity features $\mathbf{e}_{\mathcal{N}_i^r}^{(k-1)} = \frac{1}{\left|\mathcal{N}_i^r\right|} \sum_{j \in \mathcal{N}_i^r} \mathbf{e}_j^{(k-1)}, \forall j \in \mathcal{N}_i^r$
8          end
9          Aggregate all relation type-based entity features $\mathbf{e}_{\mathcal{N}_i}^{(k-1)} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \left(\alpha_r^{(k-1)} \mathbf{e}_{\mathcal{N}_i^r}^{(k-1)}\right)$
10     end
11     Integrate the self-loops into the feature propagation. $\mathbf{e}_i^{(k)} = \frac{1}{2}\left(\mathbf{e}_{\mathcal{N}_i}^{(k-1)} + \mathbf{e}_i^{(k-1)}\right)$
12 end
13 Return the entity feature matrix $\mathbf{H}^{(k)} = \left[\mathbf{e}_1^k, \dots, \mathbf{e}_n^k\right]^\top$
14 Return the relation feature matrix $\mathbf{R} = \left[\mathbf{r}_1, \dots, \mathbf{r}_n\right]^\top$
```

where $\mathbf{e}_j^{(k-1)}$ represents the neighboring node features based on r. These characteristics are combined and put through a normalizing term $\left|\mathcal{N}_i^r\right|$. Then the aggregated feature $\mathbf{e}_{\mathcal{N}_i^r}^{(k-1)}$ from relation r can be obtained. For each entity, given all relation types $\mathcal{R}$, $|\mathcal{R}|$ classes of aggregated features can be constructed. And each class will be semantically specialized, capturing only one piece of semantic information.

It can be observed that various relation types reflect different semantic information. Thus, each relation type-based feature is aggregated, and the aggregation operation can be represented as follows:

$$\mathbf{e}_{\mathcal{N}_i}^{(k-1)} = \mathrm{Agg}\left(\mathbf{e}_{\mathcal{N}_i^r}^{(k-1)}, r\right), \forall r \in \mathcal{R}. \tag{7}$$

Nevertheless, the semantic characteristics fused by significant relation types are weakened when each relation is treated equally. A critical concern is how to integrate various semantic features and choose the most appropriate relation type. Thus, SHGNet proposes the relation weighting operation to selectively aggregate informative features, and it is defined as follows:

$$\mathbf{e}_{\mathcal{N}_i}^{(k-1)} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \left(\alpha_r^{(k-1)} \mathbf{e}_{\mathcal{N}_i^r}^{(k-1)}\right), \tag{8}$$

where $\alpha_r^{(k-1)}$ denotes the weight value of relation r. After determining the significance of each relation type, each aggregated feature $\mathbf{e}_{\mathcal{N}_i^r}^{(k-1)}$ based on that relation type can be weighted with the learned $\alpha_r^{(k-1)}$ as a coefficient. These features are combined and put through a normalization term $|\mathcal{R}|$.

However, it can see that all nearby features $\mathbf{e}_{\mathcal{N}_i}^{(k-1)}$ are aggregated without the entity characteristics $\mathbf{e}_i^{(k-1)}$ itself. As a result, the self-loops must be incorporated. The self-connection operation is defined as follows:

$$\mathbf{e}_i^{(k)} = \mathrm{Agg}\left(\mathbf{e}_{\mathcal{N}_i}^{(k-1)}, \mathbf{e}_i^{(k-1)}\right). \tag{9}$$

Here, the averaging operation for self-connection is adopted to improve the efficiency of feature propagation. Then (9) can be redefined as follows:

$$\mathbf{e}_i^{(k)} = \frac{1}{2}\left(\mathbf{e}_{\mathcal{N}_i}^{(k-1)} + \mathbf{e}_i^{(k-1)}\right). \tag{10}$$

Finally, we can concatenate each entity feature $\mathbf{e}_i^{(k)}$ in the last layer to acquire the entity feature matrix $\mathbf{H}^{(k)}$. And it can be defined as:

$$\mathbf{H}^{(k)} = \left[\mathbf{e}_1^k, \dots, \mathbf{e}_n^k\right]^\top. \tag{11}$$

In summary, the proposed feature propagation framework of SHGNet has two main advantages. On the one hand, SHGNet removes the transformation matrix and nonlinear activation in traditional GNN, which can largely simplify the model design. On the other hand, SHGNet adopts a hierarchical feature propagation for heterogeneous KGs, which can deal with rich semantic information and complicated structure efficiently. The overall of feature propagation for SHGNet is shown in Algorithm 1.

Table 2
The statistics of the datasets.

Dataset	# Entities	# Relations	# Edges				Mean degree	Graph heterogeneity	Grap scale
			# Train	# Valid	# Test	# Total			
WN18RR	40,943	11	86,835	3,034	3,134	93,003	2.12	Low	Small
NELL-995	74,536	200	149,678	543	2,818	153,039	2.05	Low	Medium
FB15k-237	14,541	237	272,115	17,535	20,466	310,116	18.71	High	Large

3.5. *Triplet score*

The construction of a score function $\phi\left(\mathbf{e}_s, \mathbf{r}, \mathbf{e}_o\right) \in \mathbb{R}$ is a generic technique for link prediction. Given the triplet $\left(e_s, r, e_o\right)$, it should first look-up the matrices $\mathbf{H}^{(k)}$ and $\mathbf{R}$ for the feature of each entity and relation. The look-up equation is given as:

$$\begin{cases} \mathbf{e}_s = \mathbf{I}_s^{\top}\mathbf{H}^{(k)} \\ \mathbf{r} = \mathbf{I}_r^{\top}\mathbf{R} \\ \mathbf{e}_o = \mathbf{I}_o^{\top}\mathbf{H}^{(k)} \end{cases}, \tag{12}$$

here $\mathbf{I}_s^{\top}$, $\mathbf{I}_r^{\top}$, and $\mathbf{I}_o^{\top}$ denote the high-dimensional index of e_s, r, and e_o, respectively. In SHGNet, TransE (Ebisu & Ichise, 2020), DistMult (Yang et al., 2014), and ConvE (Dettmers et al., 2018) are selected to score the triplet. Taking ConvE as an example, the following is the definition of the score function $\phi\left(\mathbf{e}_s, \mathbf{r}, \mathbf{e}_o\right)$:

$$\begin{aligned} \hat{y} &= \phi\left(\mathbf{e}_s, \mathbf{r}, \mathbf{e}_o\right) \\ &= f\left(\operatorname{vec}\left(f\left(\left[\overline{\mathbf{e}_s}; \overline{\mathbf{r}}\right] * \omega\right)\right)\mathbf{W}\right)\mathbf{e}_o \end{aligned}, \tag{13}$$

where $\hat{y}$ is the triplet probability. The 2D reshaping is denoted by $\bar{\cdot}$, and the concatenation operation is denoted by $[\cdot]$. The notation $*$ represents the convolution operation, and the symbol ω represents the convolution filter. The linear transformation matrix is denoted by $\mathbf{W}$. The probability $\hat{y}$ should be 1 if the triplet $\left(e_s, r, e_o\right)$ is right; otherwise, the probability should be 0. As a consequence, SHGNet's loss function is as follows:

$$\mathcal{L} = -\frac{1}{N}\sum_{e_s, r, e_o \in \mathcal{E}\cup\mathcal{E}'}\left(y\log\hat{y} + (1-y)\log(1-\hat{y})\right), \tag{14}$$

in which

$$y = \begin{cases} 1 & \text{if } \left(e_s, r, e_o\right) \in \mathcal{E} \\ 0 & \text{if } \left(e_s, r, e_o\right) \in \mathcal{E}' \end{cases}, \tag{15}$$

where $\mathcal{E}'$ is a group of inaccurate triplets constructed by altering the accurate triplet group $\mathcal{E}$.

To prevent overfitting and increase generalization in SHGNet and ConvE, the following strategies were included: batch normalization (Ioffe & Szegedy, 2015), dropout technique (Srivastava et al., 2014), and label smoothing (Szegedy et al., 2016). The Adam optimizer (Kingma & Ba, 2015) is employed in the proposed model's learning process.

4. Experiments

In this part, we perform comprehensive experiments using benchmark datasets and compare our proposed model to the state-of-the-art to demonstrate its effectiveness. Before discussing the experiment findings, the general settings are briefly described. The learning process can be shown in Fig. 4.

4.1. *Experimental setup*

4.1.1. *Datasets*

To cover different structures and scales of KGs, comprehensive experiments are conducted on the following benchmark datasets: WN18RR (Dettmers et al., 2018), NELL-995 (Xiong et al., 2017), and FB15k-237 (Dettmers et al., 2018). Table 2 summarizes the statistics of these datasets. The number of edges connected to the entity is the degree of graph heterogeneity. The entity with a higher degree has more neighboring nodes. It could be observed that WN18RR and NELL-995 datasets have relatively low heterogeneity. Compared to the former, the graph heterogeneity and scale of the FB15k-237 dataset are high.

4.1.2. *Baseline models*

The presented model is compared to the following list of KGRL models:

- **TransE** (Bordes et al., 2013): TransE is translational KGRL method that models relations as translation operations between entities in vector semantic spaces.
- **DistMult** (Yang et al., 2014): DistMult utilizes the bi-linear score function to score the knowledge triplets. It is a well-known tensor factorization model.

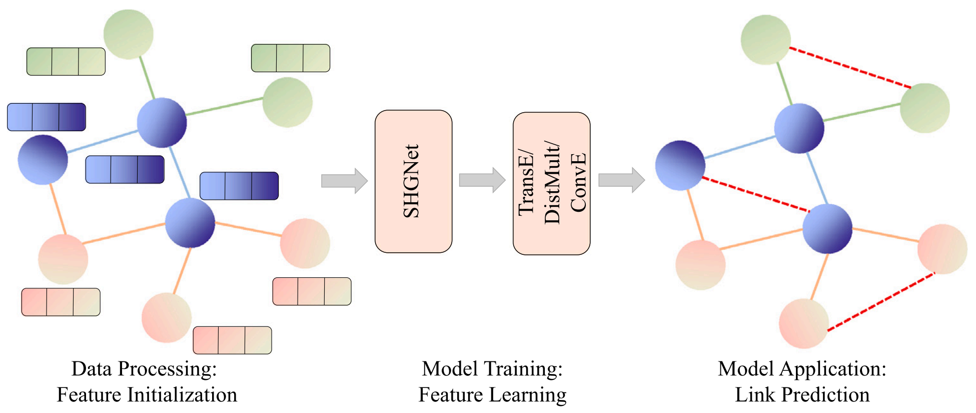
SHGNet
TransE/ DistMult/ ConvE
Data Processing: Feature Initialization
Model Training: Feature Learning
Model Application: Link Prediction

Fig. 4. Learning process of SHGNet model. We first initialize the node and relation feature for data processing. Then the proposed SHGNet is used for model training. Finally, different score functions are adopted to predict missing links in KGs.

- **ComplEx** (Trouillon et al., 2016): ComplEx is a DistMult extension that expresses relations and entities in complex space.
- **RotatE** (Sun et al., 2019): RotatE also defines relations and entities in complicated space. RotatE defines each relationship as a rotation between the entities.
- **PairRE** (Chao et al., 2021): To handle complex relations, PairRE learns the paired relation representations to encode three important relation patterns.
- **ConvE** (Dettmers et al., 2018): ConvE, as the well-known CNN-based KGRL method that performs 2D convolution over the features. The output is then combined with the object entity feature to compute a score for the triplet.
- **ConvR** (Jiang et al., 2019): ConvR creates an adaptive convolutional network to increase relation–entity interactions.
- **InteractE** (Vashishth et al., 2020a): InteractE is proposed to increase the number of entity–relation interactions. It adopts the circular convolution for the subject entity and relation features.
- **LTE-ConvE** (Zhang, Wang, et al., 2022): LTE-ConvE uses linearly transformed entity representations and adopts the ConvE as the score function to achieve comparative performance.
- **R-GCN** (Schlichtkrull et al., 2018): R-GCN is GNN-based KGRL method which can aggregate context information to entities. It is capable of handling multi-relational graph data.
- **WGCN** (Shang et al., 2019): To achieve the joint benefit of ConvE and GNN together, WGCN proposes the weighted graph convolutional.
- **CompGCN** (Vashishth et al., 2020b): CompGCN first learns the entity and relation features by incorporating the multi-relational information into the graph convolutional framework.
- **REP-OTE** (Wang et al., 2022): During the post-training process, REP-OTE makes use of graph context in KGs. The major point is to incorporate information about relational graph structure.

4.1.3. Implementation details

Following previous KGRL models, we use Hits@k and MRR as evaluation metrics. As for the settings in training, we execute a hyper-parameter search on feature dimension, learning rate, network layer, batch size, filter size, filter number, and dropout rate. We report the best hyper-parameters for each dataset as follows: {WN18RR: 200, 0.0001, 4, 128, 2×5, 64, 0.3}, {NELL-995: 200, 0.001, 4, 128, 2×7, 64, 0.4}, {FB15k-237: 200, 0.0001, 4, 128, 2×5, 64, 0.4}. We utilize PyTorch (Paszke et al., 2017) to develop the proposed model.

4.2. Performance comparison

In this subsection, SHGNet is compared to existing state-of-the-art methods on link prediction. The overall results and detailed findings are shown in Tables 3 and 4, respectively. According to the experiment results, it could be observed that:

- The proposed SHGNet outperforms significantly the translation-based KGRL model like TransE on all datasets. The primary explanation could be that SHGNet can aggregate the structural information for entities rather than modeling independent triplets. Furthermore, in most cases, SHGNet performs better than RotatE which models features in complex spaces and leads to more parameters.
- Compared with the bilinear-based models DisMult and ComplEx, our model performs better. ComplEx, despite being based on the bilinear operation, extends DisMult to the complex space, obtaining more information than the latter.
- In a general manner, CNN-based methods outperform translation-based or bilinear-based models in terms of performance. Furthermore, the proposed SHGNet generally outperforms all CNN-based models on the WN18RR and FB15k-237 datasets.

Table 3
Link prediction performance comparison on WN18RR, NELL-995, and FB15k-237 datasets.

Model		WN18RR				NELL-995				FB15k-237			
		MRR	Hits			MRR	Hits			MRR	Hits		
			@1	@3	@10		@1	@3	@10		@1	@3	@10
Shallow	TransE	0.182	0.027	0.295	0.444	0.456	0.514	0.678	0.751	0.257	0.174	0.284	0.420
	DistMult	0.430	0.390	0.440	0.490	0.522	0.610	0.704	0.795	0.241	0.155	0.263	0.419
	ComplEx	0.440	0.410	0.460	0.510	0.652	0.614	0.784	0.815	0.247	0.158	0.275	0.428
	RotatE	0.476	0.428	0.492	0.571	0.754	0.670	0.812	0.854	0.338	0.241	0.375	0.533
	PairRE	0.454	0.411	0.469	0.548	–	–	–	–	0.351	0.256	0.387	**0.544**
CNN	ConvE	0.430	0.400	0.440	0.520	0.747	0.672	0.808	0.864	0.325	0.237	0.356	0.501
	ConvR	0.475	0.443	0.489	0.537	0.749	0.679	0.814	0.869	0.350	0.261	0.385	0.528
	InteractE	0.463	0.430	0.483	0.528	0.751	0.681	0.816	0.872	0.354	0.263	0.386	0.535
	LTE-ConvE	0.472	0.437	0.485	0.544	–	–	–	–	**0.355**	0.264	0.389	0.535
GNN	R-GCN	0.123	0.080	0.137	0.207	0.120	0.082	0.126	0.188	0.249	0.151	0.264	0.417
	WGCN	0.466	0.427	0.478	0.535	0.744	0.668	0.826	0.859	0.352	0.261	0.385	0.536
	CompGCN	0.479	0.443	0.494	0.546	0.753	0.679	0.825	0.876	**0.355**	0.264	0.390	0.535
	REP-OTE	**0.488**	0.439	**0.505**	**0.588**	0.759	**0.690**	0.819	0.869	0.354	0.262	0.388	0.540
Ours	SHGNet	0.476	**0.448**	0.496	0.549	**0.764**	0.677	**0.831**	**0.884**	**0.355**	**0.268**	**0.395**	**0.544**

Table 4
Results on link prediction by relation category on FB15k-237 dataset.

Evaluation Metric	GNN-based Model	Subject entity prediction				Object entity prediction			
		1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N
MRR	R-GCN	0.398	0.086	0.448	0.256	0.527	0.778	0.059	0.379
	WGCN	0.422	0.093	0.454	0.261	0.406	0.771	0.068	0.385
	CompGCN	0.457	0.112	0.471	0.275	0.453	0.779	0.076	0.395
	REP-OTE	0.462	0.114	0.478	0.279	0.462	0.786	0.089	0.402
	SHGNet	**0.470**	**0.116**	**0.481**	**0.286**	**0.467**	**0.795**	**0.112**	**0.415**
Hits@10	R-GCN	0.531	0.179	0.639	0.448	0.534	0.869	0.137	0.589
	WGCN	0.547	0.187	0.647	0.459	0.531	0.875	0.139	0.607
	CompGCN	0.604	0.190	0.656	0.474	0.589	0.885	0.151	0.616
	REP-OTE	0.462	0.114	0.478	0.279	0.462	0.786	0.089	0.402
	SHGNet	**0.615**	**0.207**	**0.672**	**0.495**	**0.599**	**0.891**	**0.176**	**0.634**

- Compared with the latest GNN-based models, our model performs better in most cases. The main reason might be that SHGNet adopts a hierarchical feature propagation for heterogeneous KGs, which can deal with the rich semantic information and complicated structure efficiently.
- Following TransE (Bordes et al., 2013), we investigate the performance of SHGNet with different relation categories. When compared to the most recent GNN-based models, the proposed SHGNet surpasses all baselines in both subject entity prediction and object entity prediction.

4.3. Model efficiency

In this subsection, the model efficiency of SHGNet and GNN-based models is investigated. The model complexity of SHGNet is $\mathcal{O}\left(\left(|\mathcal{V}|d^2 + |\mathcal{R}|d\right)K\right)$, here d is the size of the embedding. The letters $|\mathcal{V}|$ and $|\mathcal{R}|$ represent the entire amount of entities and relationships respectively. K represents the GCN feature propagation layer. As shown in Fig. 5, we compare the running time on small-scale WN18RR and large-scale FB15k-237. It could be found that SHGNet is the fastest while still achieving competitive performance. Specifically, compared with the latest GNN-based model CompGCN, SHGNet achieves nearly three times the relative improvement in running time. The main reason is that we remove all the transformation matrix and nonlinear activation in the feature propagation, which makes our model easier to implement and train.

4.4. Ablation study

To verify the impact of relation composition and relation weighting operations in SHGNet, we conduct an ablation research by examining three variants:

- **SHGNet(w/o rc)**: It is a variant of SHGNet, which removes the relation composition operation and only considers the node feature propagation.

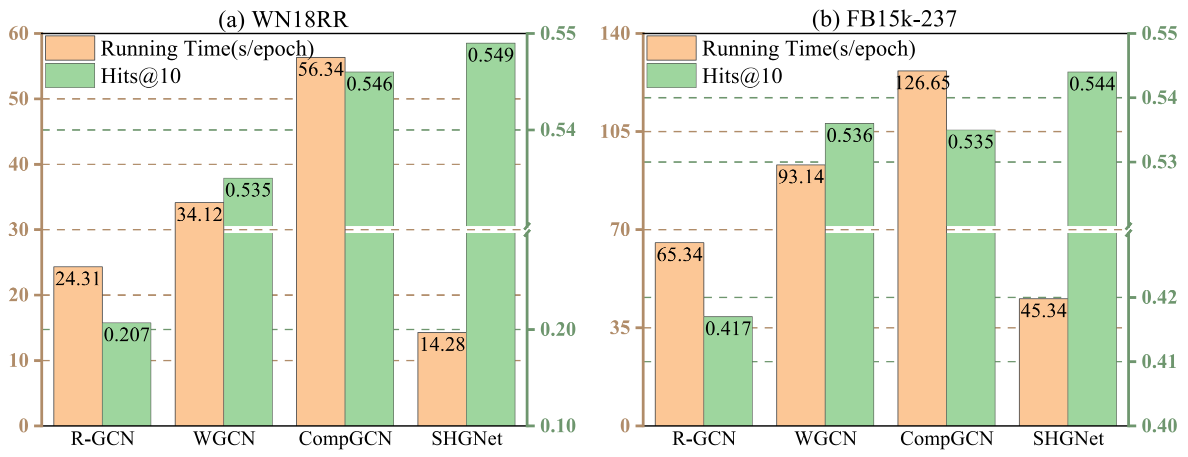

Fig. 5. Performance over running time on (a) WN18RR and (b) FB15k-237 datasets. SHGNet is the fastest while achieving competitive performance.

Table 5
Contributions of each component.

Model	WN18RR		NELL-995		FB15k-237	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
SHGNet(w/o rc&rw)	0.459	0.530	0.748	0.866	0.334	0.528
SHGNet(w/o rc)	0.470	0.542	0.758	0.877	0.351	0.539
SHGNet(w/o rw)	0.465	0.538	0.751	0.870	0.342	0.531
SHGNet	**0.476**	**0.549**	**0.764**	**0.884**	**0.355**	**0.544**

Table 6
Link prediction results on WN18RR and FB15k-237 datasets with different score functions.

Score Function	Model	WN18RR		FB15k-237	
		MRR	Hits@10	MRR	Hits@10
TransE	–	0.182	0.444	0.257	0.420
	CompGCN	0.394	0.482	0.336	0.518
	SHGNet	0.421	0.503	0.341	0.521
DistMult	–	0.430	0.490	0.241	0.419
	CompGCN	0.442	0.521	0.335	0.514
	SHGNet	0.458	0.531	0.345	0.528
ConvE	–	0.430	0.520	0.325	0.501
	CompGCN	0.479	0.546	0.355	0.535
	SHGNet	0.476	0.549	0.355	0.544

- **SHGNet(w/o rw)**: It is a SHGNet variation that removes the relation weighting operation and assigns equal significance to all relation types.
- **SHGNet(w/o rc&rw)**: It is a variant of SHGNet, which removes both relation composition and relation weighting operations in the feature propagation.

The experimental results are summarized in Table 5 and the following observations can be found: (1) Removing the relation composition and relation weighting operations degrades the link prediction performance. SHGNet(w/o rc&rw) consistently underperforms SHGNet(w/o rc) and SHGNet(w/o rw). It can be found that SHGNet(w/o rc&rw) ignores relation feature aggregation and relation type importance, which confirms SHGNet's significant effect. (2) Compared with SHGNet(w/o rw), SHGNet(w/o rc) shows better performance in most instances. One plausible explanation is that relation weighting plays a more valuable role than relation composition for SHGNet.

Furthermore, the impact of various score functions is investigated. As shown in Table 6, it can observe that the experimental results achieve a substantial improvement with different GCN components (CompGCN, SHGNet) as an encoder. And SHGNet performs better performance in most cases on FB15k-237 and WN18RR datasets. When compared to CompGCN, with TransE and DistMult as the scoring functions, SHGNet obtains a relative improvement in MRR of 6.8% and 3.6%, respectively. The outcomes can be attributable to the fact that SHGNet can selectively aggregate neighbor features of entities through the hierarchical aggregation architecture.

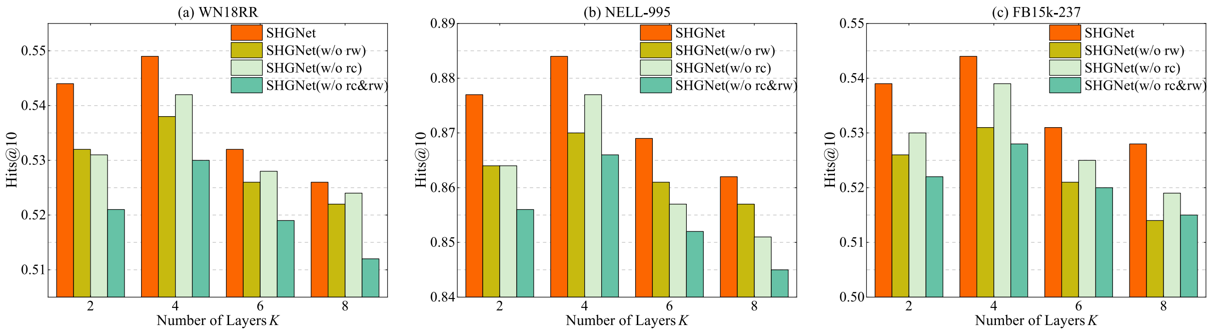

Fig. 6. Hyper-parameter sensitivity of network layer K on (a) WN18RR, (b) NELL-995, and (c) FB15k-237 datasets.

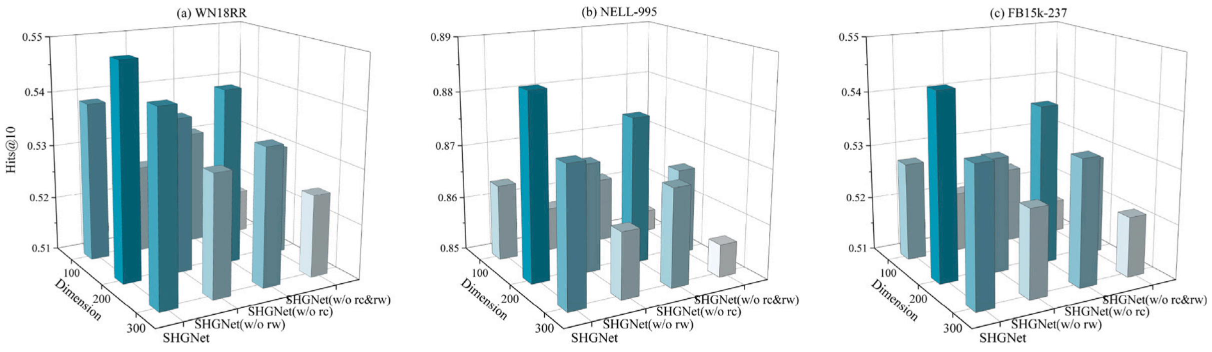

Fig. 7. Hyper-parameter sensitivity of feature dimension d on (a) WN18RR, (b) NELL-995, and (c) FB15k-237 datasets.

4.5. *Hyper-parameter sensitivity*

In this subsection, the influence of network layers K is investigated. Fig. 6 depicts the link prediction performance of SHGNet and its variants. We can observe from the results on three datasets that the proposed model SHGNet and its variants perform better when the layer is 4. Increasing the network layers leads to worse experimental results. The main reason for this may be that as network layers deepen, the aggregated features become too smooth.

Furthermore, the effect of different feature dimensions d is investigated. We report the link prediction results with the dimensionalities $d \in \{100; 200; 300\}$ on three datasets. As shown in Fig. 7, it can find that the metric Hits@10 changes consistently. The best value of d is about 200, which achieves a better performance on three datasets. Both larger and smaller feature dimensions will lead to worse performance. Furthermore, it is advised to assess the impact of feature dimension on a per-dataset basis.

4.6. *Case study*

As previously stated, the value α_r of relation types can be acquired throughout the training process. Fig. 8 shows multiple relation types and accompanying weight values. A particular relation type as well as its weight value have a positive connection. Higher weight values are assigned to the relation types *nation* and *capital_of*, indicating that SHGNet believes these two relations to be more essential. The findings suggest that the relation weight mechanism could detect and learn the differences between these relation types. The proposed model can aggregate useful information selectively based on the weight value of each relation type.

5. Conclusion

To simplify the feature propagation and model the multi-relational graph data, we propose a novel GNN-based method named SHGNet for KGRL. SHGNet discards the two standard operations including transformation matrix and nonlinear activation, which makes our model simpler to train and more efficient. In addition, we propose a hierarchical aggregation architecture for capturing complicated structures and rich semantics in heterogeneous KGs. SHGNet first aggregates each relation type-based node feature. Then, using relation weighting, features with different semantic information are fused. Massive experiments on the link prediction problem are carried out to demonstrate the effectiveness of the proposed model. SHGNet provides better scalability while maintaining prediction quality according to the results. SHGNet obtains around 6.8% and 3.6% relative increase in MRR with TransE and DistMult objective respectively compared to the best performing baseline, and achieves nearly 3x the relative improvement in running time. The limitation of proposed model and future research in this paper mainly consists of two parts. Firstly, a critical goal

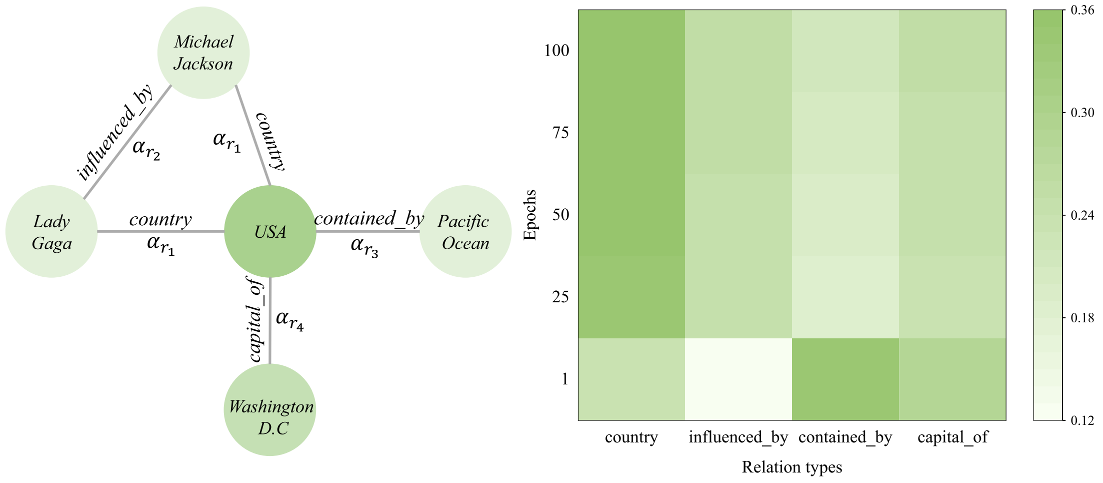

Fig. 8. Case study of various relation types and their corresponding weight value α_r.

is to obtain useful incorrect training triplets. We plan to examine whether the performance of SHGNet can be enhanced further by sampling valuable wrong triplets. Secondly, each entity in a KG may also contain additional background information, expanding SHGNet to leverage logical rules alongside words, pictures, and so forth.

CRediT authorship contribution statement

Zhifei Li: Conceptualization, Methodology, Writing – review & editing. **Qi Zhang:** Writing – review & editing. **Fangfang Zhu:** Writing – review & editing. **Duantengchuan Li:** Funding acquisition, Writing – original draft. **Chao Zheng:** Software, Writing – original draft. **Yan Zhang:** Funding acquisition, Writing – review & editing.

Data availability

Data will be made available on request.

Acknowledgment

The authors sincerely thank anonymous reviewers for their constructive comments, which helped improve this paper. This work was supported in part by the National Natural Science Foundation of China under Grant 62207011, 61977021, and Key Project of Technology Innovation in Hubei Province under Grant 2019ACA144, and School Project of Hubei University under Grant 202111903000001.

References

Balazevic, I., Allen, C., & Hospedales, T. M. (2019). TuckER: Tensor factorization for knowledge graph completion. In *Proc. conf. empirical methods natural language process* (pp. 5188–5197).

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proc. ACM SIGMOD int. conf. manage. data* (pp. 1247–1250).

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Adv. neural inf. process. syst.* (pp. 2787–2795).

Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E. R., & Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. In *Proc. 24th AAAI conf. artif. intell.* (pp. 1306–1313).

Chao, L., He, J., Wang, T., & Chu, W. (2021). PairRE: Knowledge graph embeddings via paired relation vectors. In *Proc. 59th annu. meeting assoc. comput. linguistics* (pp. 4360–4369).

Chen, X., Jia, S., & Xiang, Y. (2020). A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*, *141*, Article 112948.

Chen, J., Ma, T., & Xiao, C. (2018). FastGCN: Fast learning with graph convolutional networks via importance sampling. In *Proc. 2nd int. conf. learn. representations*.

Dettmers, T., Minervini, P., Stenetorp, P., & Riedel, S. (2018). Convolutional 2D knowledge graph embeddings. In *Proc. 32nd AAAI conf. artif. intell.* (pp. 1811–1818).

Ebisu, T., & Ichise, R. (2020). Generalized translation-based embedding of knowledge graph. *IEEE Transactions on Knowledge and Data Engineering*, *32*(5), 941–951.

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Adv. neural inf. process. syst.* (pp. 1024–1034).

Hao, Y., Liu, H., He, S., Liu, K., & Zhao, J. (2018). Pattern-revising enhanced simple question answering over knowledge bases. In *Proc. 27th int. conf. comput. linguistics* (pp. 3272–3282).

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020). LightGCN: Simplifying and powering graph convolution network for recommendation. In J. X. Huang, Y. Chang, X. Cheng, J. Kamps, V. Murdock, J. Wen, & Y. Liu (Eds.), *Proc. 43rd int. ACM SIGIR conf. res. dev. inf. retr.* (pp. 639–648).

Hu, S., Zou, L., Yu, J. X., Wang, H., & Zhao, D. (2018). Answering natural language questions by subgraph matching over knowledge graphs. *IEEE Transactions on Knowledge and Data Engineering*, *30*(5), 824–837.

Huang, X., Zhang, J., Li, D., & Li, P. (2019). Knowledge graph embedding based question answering. In *Proc. 12th ACM int. conf. web search data mining* (pp. 105–113).

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. 32nd int. conf. mach. learn.* (pp. 448–456).

Ji, G., He, S., Xu, L., Liu, K., & Zhao, J. (2015). Knowledge graph embedding via dynamic mapping matrix. In *Proc. 53rd annu. meeting assoc. comput. linguistics 7th int. joint conf. natural language processing* (pp. 687–696).

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Yu, P. S. (2022). A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(2), 494–514.

Jiang, X., Wang, Q., & Wang, B. (2019). Adaptive convolution for multi-relational learning. In *Proc. conf. North Amer. assoc. comput. linguistics: human language technol.* (pp. 978–987).

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proc. 3rd int. conf. learn. representations*.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Proc. 5th int. conf. learn. representations*.

Klicpera, J., Bojchevski, A., & Günnemann, S. (2019). Predict then propagate: Graph neural networks meet personalized PageRank. In *Proc. 7th int. conf. learn. representations*.

Li, Z., Liu, H., Zhang, Z., Liu, T., & Xiong, N. N. (2022a). Learning knowledge graph embedding with heterogeneous relation attention networks. *IEEE Transactions on Neural Networks and Learning Systems*, (8), 3961–3973.

Li, Z., Zhao, Y., Zhang, Y., & Zhang, Z. (2022b). Multi-relational graph attention networks for knowledge graph completion. *Knowledge-Based Systems*, Article 109262.

Lin, Y., Liu, Z., Zhu, X., Zhu, X., & Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proc. 29th AAAI conf. artif. intell.* (pp. 2181–2187).

Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM*, *38*(11), 39–41.

Nguyen, D. Q., Nguyen, T. D., Nguyen, D. Q., & Phung, D. (2018). A novel embedding model for knowledge base completion based on convolutional neural network. In *Proc. conf. North Amer. assoc. comput. linguistics: human language technol.* (pp. 327–333).

Nickel, M., Rosasco, L., & Poggio, T. A. (2016). Holographic embeddings of knowledge graphs. In *Proc. 30th AAAI conf. artif. intell.* (pp. 1955–1961).

Nickel, M., Tresp, V., & Kriegel, H. P. (2011). A three-way model for collective learning on multi-relational data. In *Proc. int. conf. mach. learn.* (pp. 809–816).

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in pytorch. In *Adv. neural inf. process. syst.*.

Rossi, A., Barbosa, D., Firmani, D., Matinata, A., & Merialdo, P. (2021). Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data*, *15*(2), 1–49.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., & Welling, M. (2018). Modeling relational data with graph convolutional networks. In *Proc. 15th extended semantic web conf.* (pp. 593–607).

Shang, C., Tang, Y., Huang, J., Bi, J., He, X., & Zhou, B. (2019). End-to-end structure-aware convolutional networks for knowledge base completion. In *Proc. 33rd AAAI conf. artif. intell.* (pp. 3060–3067).

Shi, C., Li, Y., Zhang, J., Sun, Y., & Yu, P. S. (2017). A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, *29*(1), 17–37.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(1), 1929–1958.

Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019). RotatE: Knowledge graph embedding by relational rotation in complex space. In *Proc. 7th int. conf. learn. representations*.

Sun, Z., Vashishth, S., Sanyal, S., Talukdar, P. P., & Yang, Y. (2020). A re-evaluation of knowledge graph completion methods. In *Proc. 58th annu. meeting assoc. comput. linguistics* (pp. 5516–5522).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proc. IEEE conf. comput. vis. pattern recog.* (pp. 2818–2826).

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., & Bouchard, G. (2016). Complex embeddings for simple link prediction. In *Proc. 33rd int. conf. mach. learn.* (pp. 2071–2080).

Vashishth, S., Sanyal, S., Nitin, V., Agrawal, N., & Talukdar, P. P. (2020a). InteractE: Improving convolution-based knowledge graph embeddings by increasing feature interactions. In *Proc. 34th AAAI conf. artif. intell.* (pp. 3009–3016).

Vashishth, S., Sanyal, S., Nitin, V., & Talukdar, P. P. (2020b). Composition-based multi-relational graph convolutional networks. In *Proc. 8th int. conf. learn. representations*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In *Adv. neural inf. process. syst.* (pp. 5998–6008).

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In *Proc. 6th int. conf. learn. representations*.

Wang, H., Dai, S., Su, W., Zhong, H., Fang, Z., Huang, Z., Feng, S., Chen, Z., Sun, Y., & Yu, D. (2022). Simple and effective relation-based embedding propagation for knowledge representation learning. In *Proc. 31st int. joint conf. artif. intell.* (pp. 2755–2761).

Wang, X., He, X., Cao, Y., Liu, M., & Chua, T.-S. (2019). KGAT: Knowledge graph attention network for recommendation. In *Proc. 25th ACM SIGKDD int. conf. knowl. discovery data mining* (pp. 950–958).

Wang, Q., Mao, Z., Wang, B., & Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, *29*(12), 2724–2743.

Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proc. 28th AAAI conf. artif. intell.* (pp. 1112–1119).

Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., & Guo, M. (2019). Exploring high-order user preference on the knowledge graph for recommender systems. *ACM Transactions on Information Systems*, *37*(3), 1–26.

Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., & Wang, Z. (2019). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proc. 25th ACM SIGKDD int. conf. knowl. discovery data mining* (pp. 968–977).

Wu, C., Liu, S., Zeng, Z., Chen, M., Alhudhaif, A., Tang, X., Alenezi, F., Alnaim, N., & Peng, X. (2022). Knowledge graph-based multi-context-aware recommendation algorithm. *Information Sciences*, *595*, 179–194.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24.

Wu, F., Souza, J., Zhang, T., Fifty, C., Yu, T., & Weinberger, K. Q. (2019). Simplifying graph convolutional networks. In *Proc. 36th int. conf. mach. learn.* (pp. 6861–6871).

Xiong, W., Hoang, T., & Wang, W. Y. (2017). DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proc. conf. empirical methods natural language process* (pp. 564–573).

Yang, B., Yih, W.-t., He, X., Gao, J., & Deng, L. (2014). Embedding entities and relations for learning and inference in knowledge bases. In *Proc. 3rd int. conf. learn. representations*.

Zhang, Z., Cui, P., & Zhu, W. (2022). Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *34*(1), 249–270.

Zhang, Z., Li, Z., Liu, H., & Xiong, N. N. (2022). Multi-scale dynamic convolutional network for knowledge graph embedding. *IEEE Transactions on Knowledge and Data Engineering*, *34*(5), 2335–2347.

Zhang, Z., Wang, J., Ye, J., & Wu, F. (2022). Rethinking graph convolutional networks in knowledge graph completion. In *Proc. 2022 world wide web conf.* (pp. 798–807).

Zhang, Q., Weng, X., Zhou, G., Zhang, Y., & Huang, J. X. (2022). ARL: an adaptive reinforcement learning framework for complex question answering over knowledge base. *Information Processing and Management*, *59*(3), Article 102933.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, *1*, 57–81.