symmetry

MDPI

Article

Hypernetwork Representation Learning with Common Constraints of the Set and Translation

Yu Zhu [1], Haixing Zhao [2,*], Jianqiang Huang [1] and Xiaoying Wang [1]

[1] Department of Computer Technology and Application, Qinghai University, Xining 810000, China
[2] State Key Laboratory of Tibetan Intelligent Information Processing and Application, Qinghai Normal University, Xining 810000, China
* Correspondence: h.x.zhao@163.com

Abstract: Different from conventional networks with only pairwise relationships among the nodes, there are also complex tuple relationships, namely the hyperedges among the nodes in the hypernetwork. However, most of the existing network representation learning methods cannot effectively capture the complex tuple relationships. Therefore, in order to resolve the above challenge, this paper proposes a hypernetwork representation learning method with common constraints of the set and translation, abbreviated as HRST, which incorporates both the hyperedge set associated with the nodes and the hyperedge regarded as the interaction relation among the nodes through the translation mechanism into the process of hypernetwork representation learning to obtain node representation vectors rich in the hypernetwork topology structure and hyperedge information. Experimental results on four hypernetwork datasets demonstrate that, for the node classification task, our method outperforms the other best baseline methods by about 1%. As for the link prediction task, our method is almost entirely superior to other baseline methods.

Keywords: hypernetwork representation; hypernetwork structure; hyperedge set; translation mechanism; common constraint

Citation: Zhu, Y.; Zhao, H.; Huang, J.; Wang, X. Hypernetwork Representation Learning with Common Constraints of the Set and Translation. *Symmetry* **2022**, *14*, 1745. https://doi.org/10.3390/sym14081745

Academic Editors: João Ruivo Paulo, Cristina P. Santos and Gabriel Pires

Received: 7 July 2022
Accepted: 18 August 2022
Published: 22 August 2022

1. Introduction

The goal of network representation learning, also regarded as network embedding, is to map each node to a low-dimensional representation vector space. The node representation vector can be applied to some popular network analysis tasks, such as node classification [1], link prediction [2], and community detection [3].

According to the type of network, network representation learning is divided into conventional network representation learning and hypernetwork representation learning. As for conventional network representation learning, most of the related studies only take the network topology structure as input to learn node representation vectors, such as DeepWalk [4], node2vec [5], LINE [6], GraRep [7], and HOPE [8]. Nevertheless, the node representation vectors learnt only from the network topology structure are not desirable vectors. Hence, some researchers have proposed some methods to incorporate other types of supplementary information, such as text, label, and community, into the process of network representation learning, such as CANE [9] and CNRL [10].

Nevertheless, the above network representation learning methods are designed for conventional networks with pairwise relationships.

As for the hypernetwork, hypernetwork representation learning [11] has been gradually widely studied by researchers. According to the characteristics of hypernetwork representation learning methods, they are divided into expanded spectral analysis and non-expanded methods. The expanded spectral analysis methods, such as star and clique extensions [12], transform the hypernetwork into a conventional network to learn the node representation vector while losing hyperedge information during the hypernetwork expansion. The non-expanded methods without the hyperedge decomposition are mainly

divided into non-expanded spectral analysis and neural-network-based methods, such as Hyper2vec [13], HPHG [14], DHNE [15], and so on.

Although the expanded spectral analysis methods are intuitive, there is a loss of hyperedge information. The non-expanded methods do not decompose the hyperedge. For example, Hyper2vec captures the pairwise relationships among the nodes on the hyperedge-based walk sequence but does not capture the tuple relationships among the nodes well. HPHG, combined with a one-dimensional convolutional layer, effectively captures tuple relationships among the nodes, and DHNE captures tuple relationships among the nodes by combining multi-layer perceptron, but both HPHG and DHNE are limited to heterogeneous hyperedges with a fixed size. However, the above methods cannot effectively capture the complex tuple relationships with an unfixed size. Therefore, in order to resolve the above challenge, this paper proposes a hypernetwork representation learning method with common constraints of the set and translation to effectively capture tuple relationships among the nodes.

The following two points are the main characteristics of this paper:

- The hypernetwork was transformed into a conventional network abstracted as a two-section graph. Based on this conventional network, a hypernetwork representation learning method with common constraints of the set and translation was proposed to learn node representation vectors rich in both the hypernetwork topology structure and hyperedges.
- The strength of our proposed method was to incorporate a hyperedge (tuple relationship) that is not limited to a fixed size into the process of hypernetwork representation learning. The weakness of our proposed study was that some hypernetwork structure information was still missing because the hypernetwork was transformed into a conventional network.

2. Related Studies

Different from the conventional network with only pairwise relationships among the nodes, there are also complex tuple relationships, namely the hyperedges among the nodes in the hypernetwork. However, most of the existing network representation learning methods cannot effectively capture the complex tuple relationships. Therefore, in order to resolve the above challenge, researchers have proposed some hypernetwork representation learning methods, which were divided into the expanded spectral analysis and non-expanded methods. As for the expanded spectral analysis methods, by transforming the hypernetwork into a conventional network, the problem of hypernetwork representation learning was simplified into the problem of conventional network representation learning, and then solved according to the spectral characteristics of the Laplace matrix. For example, star and clique extensions are two classical hypernetwork expansion methods. As for the non-expanded methods, they are mainly divided into the non-expanded spectral analysis and the neural network-based methods. The non-expanded spectral analysis methods directly model the hypernetwork, that is, the Laplacian matrix on the hypernetwork is directly built, and this modeling process ensures the integrity of hypernetwork information. For example, Zhou [16] extended the powerful method of spectral clustering [17], originally run on undirected graphs, to the hypergraph [18] and further developed the hypergraph learning algorithm on the basis of the spectral hypergraph clustering method. Hyper2vec was proposed based on the biased random walk strategy on the hypergraph to preserve the structure and inherent property of the hypernetwork. Neural-network-based methods have a strong learning ability, flexible structure design, and high generalization, which make up for the defects of spectral analysis methods. For example, for DHNE, it was theoretically proved that the linear similarity measure in the embedding space used by the existing methods could not preserve the indecomposability of the hypernetwork. Thus, a new deep model was proposed to realize the local and global proximity of the nonlinear tuple similarity function in the embedding space. HPHG designs a random walk based on the hypergraph to retain the hypernetwork topology structure information to learn

node representation vectors. Hyper-SAGNN [19] uses a self-attention mechanism [20] to aggregate hypergraph information, constructs pairwise attention coefficients between the nodes as the dynamic features of the nodes, and combines the original static features of the nodes to describe the nodes.

3. Problem Definition

Given the hypernetwork $H = (V, E)$, abstracted as the hypergraph, which was composed of the node set $V = \{v_i\}_{i=1}^{|V|}$ and the hyperedge set $E = \{e_i = (v_1, v_2, \ldots, v_m)\}_{i=1}^{|E|}$ $(m \geq 2)$, the goal of hypernetwork representation learning, with common constraints of the set and translation, was to learn a low-dimensional vector $r_n \in R^k$ for each node n in the hypernetwork, where k was expected to be much smaller than $|V|$.

4. Preliminaries

4.1. Transforming Hypergraph into Two-Section Graph

A feasible way to transform the hypergraph into a conventional graph was to carry out the research of the hypergraph, because the research for the conventional graph was relatively mature. In the literature [18], hypergraphs were transformed into three kinds of conventional graphs, namely line, incidence, and two-section graphs. In fact, two-section graphs lost less hypernetwork structure information than line and incidence graphs. Hence, a hypergraph was transformed into a two-section graph in this study. A hypergraph and its corresponding two-section graph are shown in Figure 1.

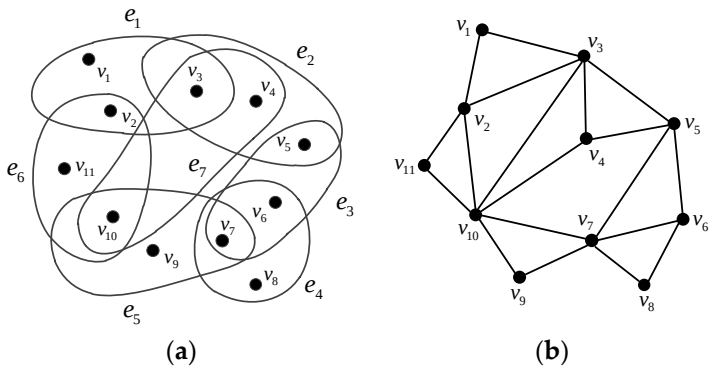

Figure 1. Hypergraph and two-section graph: (**a**) hypergraph; (**b**) two-section graph.

The two-section graph $S = (V', E')$ transformed from the hypergraph $H = (V, E)$ was a conventional graph with the following conditions:

- $V' = V$, that is, the node set of two-section graph S was equal to the node set of the hypergraph H.
- One edge was associated with any two different nodes if and only if the two nodes were simultaneously associated with at least one hyperedge.

4.2. TransE

Knowledge representation is the vectorization of the entity and relation in the knowledge graph, which specifically maps the entity or relation to a low-dimensional vector space. For simplicity, (h, r, t) denotes the triplet (head, relation, and tail), where $\boldsymbol{h}$, $\boldsymbol{r}$, and $\boldsymbol{t}$ denote the head entity, relation, and tail entity, respectively, and $\boldsymbol{h}$, $\boldsymbol{r}$, and $\boldsymbol{t}$ denote the vectors corresponding to the head entity, relation, and tail entity, respectively. In the relation extraction of the knowledge graph, as a knowledge representation learning algorithm based on the translation, TransE [21] thought that the head entity vector plus the relation vector were approximately equal to the tail entity vector, that is, $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$ when the triplet (h, r, t) held ($\boldsymbol{t}$ should be the nearest neighbor of $\boldsymbol{h} + \boldsymbol{r}$), while $\boldsymbol{h} + \boldsymbol{r}$ should otherwise be far away from $\boldsymbol{t}$. TransE is shown in Figure 2.

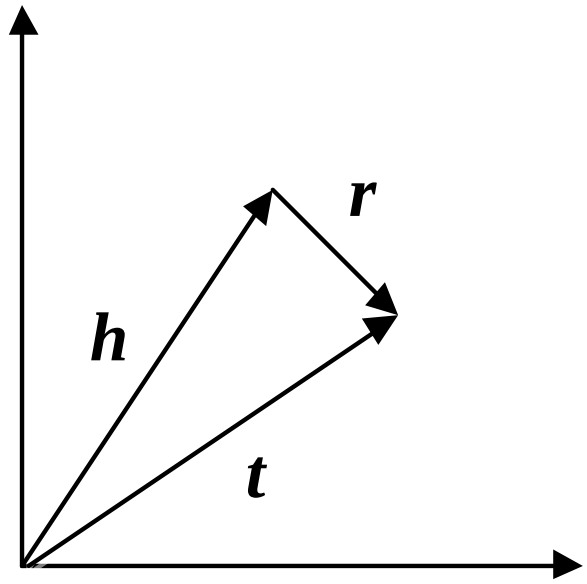

Figure 2. TransE, where $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$.

5. Our Method

Hypernetwork representation learning with common constraints of the set and translation HRST is introduced in detail in this section. Firstly, the topology-derived model is introduced in Section 5.1. Secondly, the set constraint model is introduced in Section 5.2. Thirdly, the translation constraint model is introduced in Section 5.3. Fourthly, the joint optimization of the above three models is introduced in detail in Section 5.4. Finally, the complexity analysis of HRST is introduced in Section 5.5.

5.1. Topology-Derived Model

Because the computational efficiency of CBOW [22] is greater than that of skip-gram [22], a topology-derived model [11] based on the negative sampling to be used to capture the network structure was introduced. To be specific, in the optimization procedure of this model, the center node n was the positive sample, other nodes were the negative samples, and $NEG(n)$ was the subset of negative samples with a predefined size ds. For $\forall\, u \in V$, the node labels are denoted as follows:

$$L^n(u) = \begin{cases} 1, & u \in \{n\} \\ 0, & u \in NEG(n) \end{cases} \tag{1}$$

The prediction probability of the node u is denoted as $p(u|context(n))$ under the condition of the contextual nodes $context(n)$ corresponding to n. The node sequence set is denoted as C. In view of the above conditions, we maximized the following objective function:

$$D_1 = \prod_{n \in C} \prod_{u \in \{\{n\} \cup NEG(n)\}} p(u|context(n)) \tag{2}$$

When the node n was regarded as the contextual node, the embedding vector v_n was the representation of the node n, while the parameter vector $\boldsymbol{\theta}_n$ was the representation of the node n when the node n was regarded as the center node. $p(u|context(n))$ in the formula (2) is denoted as follows:

$$p(u|context(n)) = \begin{cases} \sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u), & L^n(u) = 1 \\ 1 - \sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u), & L^n(u) = 0 \end{cases} \tag{3}$$

where $\sigma(X_n^T\boldsymbol{\theta}_u) = 1/(1+e^{-X_n^T\boldsymbol{\theta}_u})$ is a sigmoid function and X_n is the summing operation of the representation vectors corresponding to all the nodes of $context(n)$. Formula (3) can also be written as an integral expression:

$$p(u|context(n)) = [\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]^{L^n(u)} \cdot [1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]^{1-L^n(u)} \tag{4}$$

Consequently, Formula (2) can be rewritten as follows:

$$D_1 = \prod_{n\in C}\prod_{u\in\{\{n\}\cup NEG(n)\}}\left\{[\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]^{L^n(u)} \cdot [1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]^{1-L^n(u)}\right\} \tag{5}$$

Formally, by means of maximizing D_1, the network topology was encoded into the node representation vectors.

5.2. Set Constraint Model

Because the above topology-derived model only considered the network structure, a set constraint model [11] based on the negative sampling to consider both the network structure and the hyperedge was introduced. To be specific, in the optimization procedure of this model, T_n was the set of the hyperedges associated with the center node n, and also the set of the nodes associated with the center node n if the hyperedge was regarded as the node. The center node n was the positive sample, and other nodes not associated with the center node $n \in V$ were the negative samples. As for $\forall\, v \in T_n$, $NEG(v)$ was the subset of negative samples with a predefined size ds, and the node labels are denoted as follows:

$$\delta(\vartheta|v) = \begin{cases} 1, & \vartheta \in \{n\} \\ 0, & \vartheta \in NEG(v) \end{cases} \tag{6}$$

In view of the node sequences C and the set of the hyperedges, we tried to maximize the following objective function to meet the set constraint:

$$\begin{aligned} D_2 &= \prod_{n\in C}\prod_{v\in T_n} p(n|v) = \prod_{n\in C}\prod_{v\in T_n}\prod_{\vartheta\in\{\{n\}\cup NEG(v)\}}\left\{\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)^{\delta(\vartheta|v)} \cdot [1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]^{1-\delta(\vartheta|v)}\right\} \\ &= \prod_{n\in C}\prod_{v\in T_n}\left\{\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_n)\cdot\prod_{\vartheta\in NEG(v)}[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\right\} \end{aligned} \tag{7}$$

where $\boldsymbol{e}_v$ is the parameter vector corresponding to $v \in T_n$.

By means of maximizing D_2, the hyperedges were encoded into the node representation vectors.

5.3. Translation Constraint Model

Because the above set constraint model did not fully consider the hyperedges, it could not learn node representation vectors very well. Hence, we tried to incorporate the hyperedges associated with the nodes, regarded as the interaction relationships among the nodes, into the process of hypernetwork representation learning.

Inspired by the successful application of the translation mechanism in TransE, the nodes and interaction relationships were mapped into a unified representation space, where the interaction relationships among the nodes could be regarded as the translation operations in the representation space.

To be specific, for the center node n in V, if there was a node $h \in V$ and a hyperedge $r \in E$ to make $n \in r,\ h \in r$, that is, the hyperedge r was simultaneously associated with the node n and node h, a normal triplet (h, r, n) held, where h is a node with the relationship r with the node n, H_r is the set of the nodes with the relationship r with the node n, and R_n is the set of hyperedges associated with the center node n, namely the set of relationships.

Inspired by the above topology-derived model, a novel translation constraint model based on the negative sampling was proposed. To be specific, in the optimization procedure of this model, the center node n was the positive sample, other nodes were the negative samples, and $NEG(n)$ was the subset of negative samples of the center node n with a predefined size ds. For $\forall\ \xi \in V$, the node labels are denoted as follows:

$$\delta^n(\xi) = \begin{cases} 1, & \xi \in \{n\} \\ 0, & \xi \in NEG(n) \end{cases} \tag{8}$$

In view of the node sequences C and the translation constraint, we tried to maximize the following objective function to meet the translation constraint:

$$D_3 = \prod_{n \in C} \prod_{r \in R_n} \prod_{h \in H_r} \prod_{\xi \in \{\{n\} \cup NEG(n)\}} p(\xi | h + r) = \prod_{n \in C} \prod_{r \in R_n} \prod_{h \in H_r} \prod_{\xi \in \{\{n\} \cup NEG(n)\}} \left\{ \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_{\xi})^{\delta^n(\xi)} \cdot [1 - \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_{\xi})]^{1-\delta^n(\xi)} \right\} \tag{9}$$

where $\boldsymbol{e}_h$, $\boldsymbol{e}_r$, and $\boldsymbol{e}_{h+r}$ are all the parameter vectors, $\boldsymbol{e}_{h+r} = \boldsymbol{e}_h + \boldsymbol{e}_r$. By means of maximizing D_3, the interaction relations were encoded into the node representation vectors.

5.4. Joint Optimization

In this subsection, the hypernetwork representation learning method with common constraints of the set and translation HRST is proposed. HRST can jointly optimize the topology-derived, set constraint, and translation constraint models. Figure 3 shows the HRST framework.

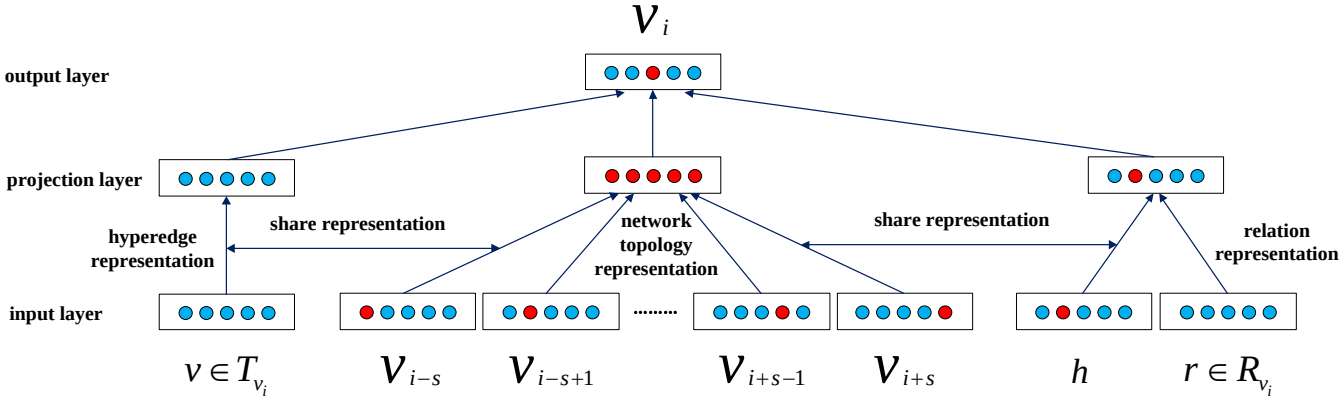

Figure 3. HRST framework, where v_i is the center node; the other nodes v_{i-s}, v_{i-s+1}, v_{i+s-1}, v_{i+s}, etc. are contextual nodes of the center node v_i, namely $context(v_i)$; T_{v_i} is the hyperedge set associated with the center node v_i; r is the interaction relation, namely the hyperedge; h is a node with the relation r with the center node v_i; and R_{v_i} is the hyperedge set associated with the center node v_i.

In Figure 3, the network topology representation, and the hyperedge and relation representations from the topology-derived model, and the set constraint and the translation constraint models, respectively, shared the same representation rich in the hyperedges.

In order to facilitate calculation, we took the logarithm of D_1, D_2, and D_3 to maximize the following joint optimization objective function to meet common constraints of the set and translation:

$$
\begin{aligned}
L &= \sum_{n\in C}\left\{\begin{array}{l}
\sum\limits_{u\in\{\{n\}\cup NEG(n)\}}\left\{L^n(u)\cdot\log[\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]+[1-L^n(u)]\cdot\log[1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\right\}+\\
\beta_1\cdot\sum\limits_{v\in T_n}\sum\limits_{\vartheta\in\{\{n\}\cup NEG(v)\}}\left\{\delta(\vartheta|v)\cdot\log[\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]+[1-\delta(\vartheta|v)]\cdot\log[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\right\}+\\
\beta_2\cdot\sum\limits_{r\in R_n}\sum\limits_{h\in H_r}\sum\limits_{\xi\in\{\{n\}\cup NEG(n)\}}\left\{\delta^n(\xi)\cdot\log[\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}}\boldsymbol{\theta}_\xi)]+[1-\delta^n(\xi)]\cdot\log[1-\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}}\boldsymbol{\theta}_\xi)]\right\}
\end{array}\right\}\\
&= \sum_{n\in C}\left\{\begin{array}{l}
\sum\limits_{u\in\{\{n\}\cup NEG(n)\}}\left\{L^n(u)\cdot\log[\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]+[1-L^n(u)]\cdot\log[1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\right\}+\\
\sum\limits_{v\in T_n}\sum\limits_{\vartheta\in\{\{n\}\cup NEG(v)\}}\beta_1\cdot\left\{\delta(\vartheta|v)\cdot\log[\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]+[1-\delta(\vartheta|v)]\cdot\log[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\right\}+\\
\sum\limits_{r\in R_n}\sum\limits_{h\in H_r}\sum\limits_{\xi\in\{\{n\}\cup NEG(n)\}}\beta_2\cdot\left\{\delta^n(\xi)\cdot\log[\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}}\boldsymbol{\theta}_\xi)]+[1-\delta^n(\xi)]\cdot\log[1-\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}}\boldsymbol{\theta}_\xi)]\right\}
\end{array}\right\}
\end{aligned}
\tag{10}
$$

where the harmonic factors β_1 and β_2 were used to counterweigh the contribution rate among the topology-derived, the set constraint, and the translation constraint models.

In order to facilitate derivation, $L(n,u,v,\vartheta,r,h,\xi)$ is denoted as follows:

$$
\begin{aligned}
L(n,u,v,\vartheta,r,h,\xi) = &\left\{L^n(u)\cdot\log[\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]+[1-L^n(u)]\cdot\log[1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\right\}+\\
&\beta_1\cdot\left\{\delta(\vartheta|v)\cdot\log[\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]+[1-\delta(\vartheta|v)]\cdot\log[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\right\}+\\
&\beta_2\cdot\left\{\delta^n(\xi)\cdot\log[\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}}\boldsymbol{\theta}_\xi)]+[1-\delta^n(\xi)]\cdot\log[1-\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}}\boldsymbol{\theta}_\xi)]\right\}
\end{aligned}
\tag{11}
$$

The objective function L was optimized by the stochastic gradient ascent method. The objective was to give six kinds of gradients of L.

Firstly, the gradient on $\boldsymbol{\theta}_u$ of $L(n,u,v,\vartheta,r,h,\xi)$ was calculated as follows:

$$
\begin{aligned}
\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial\boldsymbol{\theta}_u} &= L^n(u)\cdot[1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\cdot X_n-[1-L^n(u)]\cdot\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)\cdot X_n\\
&= \left\{L^n(u)\cdot[1-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]-[1-L^n(u)]\cdot\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)\right\}\cdot X_n\\
&= [L^n(u)-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\cdot X_n
\end{aligned}
\tag{12}
$$

Consequently, the updating formula of $\boldsymbol{\theta}_u$ is denoted as follows:

$$
\boldsymbol{\theta}_u = \boldsymbol{\theta}_u+\alpha\cdot[L^n(u)-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\cdot X_n \tag{13}
$$

where α is the learning rate.

Secondly, the gradient on X_n of $L(n,u,v,\vartheta,r,h,\xi)$ was calculated. The symmetry property between $\boldsymbol{\theta}_u$ and X_n was utilized to get the gradient of X_n:

$$
\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial X_n} = [L^n(u)-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\cdot\boldsymbol{\theta}_u \tag{14}
$$

Consequently, the updating formula of $\mathrm{v}_{v'}$ is denoted as follows, where $v'\in context(n)$:

$$
\begin{aligned}
\boldsymbol{v}_{\mathrm{v}'} &= \boldsymbol{v}_{\mathrm{v}'}+\alpha\cdot\sum_{u\in\{\{n\}\cup NEG(n)\}}\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial X_n}\\
&= \boldsymbol{v}_{\mathrm{v}'}+\alpha\cdot\sum_{u\in\{\{n\}\cup NEG(n)\}}[L^n(u)-\sigma(X_n^{\mathrm{T}}\boldsymbol{\theta}_u)]\cdot\boldsymbol{\theta}_u
\end{aligned}
\tag{15}
$$

Thirdly, the gradient on $\boldsymbol{\theta}_\vartheta$ of $L(n,u,v,\vartheta,r,h,\xi)$ was calculated as follows:

$$
\begin{aligned}
\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial\boldsymbol{\theta}_\vartheta} &= \beta_1\cdot\left\{\frac{\partial}{\partial\boldsymbol{\theta}_\vartheta}\left\{\delta(\vartheta|v)\cdot\log[\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]+[1-\delta(\vartheta|v)]\cdot\log[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\right\}\right\}\\
&= \beta_1\cdot\left\{\delta(\vartheta|v)\cdot[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\cdot\boldsymbol{e}_v-[1-\delta(\vartheta|v)]\cdot\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)\cdot\boldsymbol{e}_v\right\}\\
&= \beta_1\cdot\left\{\left\{\delta(\vartheta|v)\cdot[1-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]-[1-\delta(\vartheta|v)]\cdot\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)\right\}\cdot\boldsymbol{e}_v\right\}\\
&= \beta_1\cdot[\delta(\vartheta|v)-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\cdot\boldsymbol{e}_v
\end{aligned}
\tag{16}
$$

Consequently, the updating formula of $\boldsymbol{\theta}_\vartheta$ is denoted as follows:

$$
\boldsymbol{\theta}_\vartheta = \boldsymbol{\theta}_\vartheta+\alpha\cdot\beta_1\cdot[\delta(\vartheta\Big|v)-\sigma(\boldsymbol{e}_v^{\mathrm{T}}\boldsymbol{\theta}_\vartheta)]\cdot\boldsymbol{e}_v \tag{17}
$$

Fourthly, the gradient on $\boldsymbol{e}_v$ of $L(n,u,v,\vartheta,r,h,\xi)$ was calculated. The symmetry property between $\boldsymbol{\theta}_\vartheta$ and $\boldsymbol{e}_v$ was utilized to get the gradient of $\boldsymbol{e}_v$:

$$\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial \boldsymbol{e}_v} = \beta_1 \cdot [\delta(\vartheta \Big| v) - \sigma(\boldsymbol{e}_v^{\mathrm{T}} \boldsymbol{\theta}_\vartheta)] \cdot \boldsymbol{\theta}_\vartheta \tag{18}$$

Consequently, the updating formula of $\boldsymbol{e}_v$ is denoted as follows, where $v \in T_n$:

$$\boldsymbol{e}_v = \boldsymbol{e}_v + \alpha \cdot \beta_1 \cdot [\delta(\vartheta \Big| v) - \sigma(\boldsymbol{e}_v^{\mathrm{T}} \boldsymbol{\theta}_\vartheta)] \cdot \boldsymbol{\theta}_\vartheta \tag{19}$$

Fifthly, the gradient on $\boldsymbol{\theta}_\xi$ of $L(n,u,v,\vartheta,r,h,\xi)$ was calculated as follows:

$$\begin{aligned} \tfrac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial \boldsymbol{\theta}_\xi} &= \beta_2 \cdot \left\{ \tfrac{\partial}{\partial \boldsymbol{\theta}_\xi} \left\{ \delta^n(\xi) \cdot \log[\sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi)] + [1 - \delta^n(\xi)] \cdot \log[1 - \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi)] \right\} \right\} \\ &= \beta_2 \cdot \left\{ \delta^n(\xi) \cdot [1 - \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi)] \cdot \boldsymbol{e}_{h+r} - [1 - \delta^n(\xi)] \cdot \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi) \cdot \boldsymbol{e}_{h+r} \right\} \\ &= \beta_2 \cdot \left\{ \left\{ \delta^n(\xi) \cdot [1 - \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi)] - [1 - \delta^n(\xi)] \cdot \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi) \right\} \cdot \boldsymbol{e}_{h+r} \right\} \\ &= \beta_2 \cdot [\delta^n(\xi) - \sigma(\boldsymbol{e}_{h+r}^{\mathrm{T}} \boldsymbol{\theta}_\xi)] \cdot \boldsymbol{e}_{h+r} \end{aligned} \tag{20}$$

Consequently, the updating formula of $\boldsymbol{\theta}_\xi$ is denoted as follows:

$$\boldsymbol{\theta}_\xi = \boldsymbol{\theta}_\xi + \alpha \cdot \beta_2 \cdot [\delta^n(\xi) - \sigma(\boldsymbol{e}_{h+r}^{T} \boldsymbol{\theta}_\xi)] \cdot \boldsymbol{e}_{h+r} \tag{21}$$

Finally, the gradient on $\boldsymbol{e}_{h+r}$ of $L(n,u,v,\vartheta,r,h,\xi)$ was calculated. The symmetry property between $\boldsymbol{\theta}_\xi$ and $\boldsymbol{e}_{h+r}$ was utilized to get the gradient on $\boldsymbol{e}_{h+r}$:

$$\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial \boldsymbol{e}_{h+r}} = \beta_2 \cdot [\delta^n(\xi) - \sigma(\boldsymbol{e}_{h+r}^{T} \boldsymbol{\theta}_\xi)] \cdot \boldsymbol{\theta}_\xi \tag{22}$$

where, $\boldsymbol{e}_{h+r} = \boldsymbol{e}_h + \boldsymbol{e}_r$ and the vectors to update are $\boldsymbol{e}_h$ and $\boldsymbol{e}_r$, so the updating of the gradient $\frac{\partial L(n,u,v,\vartheta,r,h,\xi)}{\partial \boldsymbol{e}_{h+r}}$ was utilized on $\boldsymbol{e}_h$ and $\boldsymbol{e}_r$ respectively. The updating formulae of $\boldsymbol{e}_h$ and $\boldsymbol{e}_r$ are denoted as follows.

$$\boldsymbol{e}_h = \boldsymbol{e}_h + \alpha \cdot \beta_2 \cdot \sum_{\xi \in \{\{n\} \cup NEG(n)\}} [\delta^n(\xi) - \sigma(\boldsymbol{e}_{h+r}^{T} \boldsymbol{\theta}_\xi)] \cdot \boldsymbol{\theta}_\xi \tag{23}$$

$$\boldsymbol{e}_r = \boldsymbol{e}_r + \alpha \cdot \beta_2 \cdot \sum_{\xi \in \{\{n\} \cup NEG(n)\}} [\delta^n(\xi) - \sigma(\boldsymbol{e}_{h+r}^{T} \boldsymbol{\theta}_\xi)] \cdot \boldsymbol{\theta}_\xi \tag{24}$$

5.5. Complexity Analysis

The time complexity of HRST was $O(|C| \cdot (ds+1) \cdot (\beta_1 \cdot M_s + \beta_2 \cdot M_{H_r} \cdot M_R + 1))$, where the time complexities of the topology-derived, the set constraint, and the translation constraint models are $O(|C| \cdot (ds+1))$, $O(|C| \cdot (ds+1) \cdot M_s)$, and $O(|C| \cdot (ds+1) \cdot M_{H_r} \cdot M_R)$, respectively, where ds is a constant independent of the network size, and $M_s = max\left\{ \left|T_{v_1}\right|, \left|T_{v_2}\right|, \ldots, \left|T_{v_{|V|}}\right| \right\}$ are the maxima of the set of the hyperedges associated with the node v_i, M_{H_r} is the maxima of H_r, which is the set of the nodes in the triplets with the relation r with the node v_i, and $M_R = max\left\{ \left|R_{v_1}\right|, \left|R_{v_2}\right|, \ldots, \left|R_{v_{|V|}}\right| \right\}$ are the maxima of the set of the relations associated with the node v_i.

The stochastic gradient ascent method was used for optimization. More details are shown in Algorithm 1.

Algorithm 1: HRST

Input:
  Hypernetwork $H = (V, E)$
  Embedding size d
Output:
  Embedding matrix $X \in R^{|V| \times d}$
for node n in V do
  initialize embedding vector $\boldsymbol{v}_n \in R^{1 \times d}$
  initialize parameter vector $\boldsymbol{\theta}_n \in R^{1 \times d}$
  for node v in T_n do
    initialize parameter vector $\boldsymbol{e}_v \in R^{1 \times d}$
  end for
  for hyperedge r in R_n do
    for node h in H_r do
      initialize parameter vector $\boldsymbol{e}_{h+r} \in R^{1 \times d}$
    end for
  end for
end for
node sequences C = RandomWalk()
for $(n, context(n))$ in C do
  update parameter vector according to Formula (13)
  update embedding vector according to Formula (15)
  update parameter vector according to Formula (17)
  for node v in T_n do
    update parameter vector according to Formula (19)
  end for
  update parameter vector according to Formula (21)
  for hyperedge r in R_n do
    for node h in H_r do
      update parameter vector according to Formula (23)
      update parameter vector according to Formula (24)
    end for
  end for
end for
for $i = 0$; $i < |V|$; $i++$ do
  $X_i = \boldsymbol{v}_{v_i}$
end for
return X

6. Experiments

6.1. Dataset

Four hypernetwork datasets were used to evaluate the effectiveness of HRST. Detailed dataset statistics are shown in Table 1.

Table 1. Dataset statistics.

Dataset	Node Type			#(V)			#(E)
GPS	user	location	activity	146	70	5	1436
MovieLens	user	movie	tag	457	1688	1530	5965
drug	user	drug	reaction	4	132	221	1195
wordnet	head	relationship	tail	1754	7	1549	2174

Four datasets are shown as follows:

- GPS [23] described a situation where a user partook in an activity in a location. The set of three-tuple <user, location, activity> was used to construct the hypernetwork.

- MovieLens [24] described personal tag activities from MovieLens. The set of three-tuple <user, movie, tag> was used to construct the hypernetwork, where each movie had at least one genre.
- Drug (http://www.fda.gov/Drugs/, accessed on 27 January 2020) described a situation where the user took drugs and had certain reactions that led to adverse events. The set of three-tuple <user, drug, reaction> was used to construct the hypernetwork.
- wordnet [21] was composed of a set of triplets <head, relation, tail> extracted from WordNet3.0. The set of three-tuple <head, relationship, tail> was used to construct the hypernetwork.

6.2. Baseline Methods

DeepWalk. DeepWalk is a classical representation learning method to learn node representation vectors.

node2vec. node2vec preserves network neighborhoods of the nodes to learn node representation vectors.

LINE. LINE preserves both first- and second-order proximities to learn node representation vectors.

GraRep. GraRep captures global structure properties of a graph by k-step loss functions to learn node representation vectors.

HOPE. HOPE captures the higher-order proximity and asymmetric transitivity of a graph to learn node representation vectors.

SDNE. SDNE [25] utilizes first- and second-order proximities to characterize local and global network structures to learn node representation vectors.

HRSC. HRSC [11] incorporates the hyperedge sets associated with the nodes into the process of hypernetwork representation learning.

HRTC. HRTC models the interaction relationships among the nodes through the translation mechanism and incorporates the relationships among the nodes into the process of hypernetwork representation learning.

HRST. HRST incorporates the hyperedge sets associated with the nodes and interaction relationships among the nodes modeled through the translation mechanism into the process of hypernetwork representation learning.

6.3. Experimental Setting

Node classification and link prediction were used to evaluate the effectiveness of HRST. The vector dimension was set to 100, the number of the random walks to begin with every node to 10, and the length of the random walks to begin with every node to 40. Some datasets were randomly selected as the training set and the rest as the test set.

6.4. Node Classification

The multi-label classification tasks [1] were conducted on the MovieLens and wordnet datasets because labels are only on these two datasets. In addition, the nodes without labels on the two datasets were removed. An SVM [26] classifier was trained to calculate node classification accuracies.

From Tables 2 and 3, the following observations were obtained as follows:

- For the two datasets, the average value of the node classification accuracy of HRST was very close to those of HRSC and HRTC, and better than those of other baseline methods. For instance, for the average values of the node classification accuracy, HRST outperformed the other best baseline methods (e.g., DeepWalk) by about 1% on the two datasets. Meanwhile, the average values of the node classification accuracies of the remaining baseline methods were roughly weaker than those of HRST.
- The average value of the node classification accuracy of GraRep ranked only second to those of HRST, HRSC, HRTC, and DeepWalk, and was very close to those of DeepWalk, because GraRep integrated the hyperedges to a certain extent into the process of network representation learning.

Table 2. Node classification accuracies on MovieLens (%).

Methods	Training Ratios										
	10%	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**Average**	**Rank**
DeepWalk	48.01	50.35	51.41	52.60	52.59	53.47	53.57	54.23	54.09	52.26	4
node2vec	46.93	49.28	50.77	51.51	52.62	52.58	53.04	53.44	52.71	51.43	6
LINE	43.93	45.46	46.52	47.29	47.70	48.16	48.02	49.09	48.34	47.17	8
GraRep	47.75	50.11	51.16	52.01	52.10	53.15	53.34	53.43	53.24	51.81	5
HOPE	46.33	48.57	49.95	50.69	51.06	51.04	51.29	52.53	51.79	50.36	7
SDNE	41.74	41.79	42.34	42.73	43.36	43.27	43.89	43.43	42.81	42.82	9
HRSC	48.60	50.81	52.02	53.19	53.73	54.12	54.83	54.95	56.14	53.15	3
HRTC	48.73	51.26	52.71	53.62	54.38	54.57	55.03	55.82	56.30	53.60	1
HRST	48.45	50.90	52.41	53.23	53.83	54.45	54.95	55.58	55.84	53.29	2

Table 3. Node classification accuracies on wordnet (%).

Methods	Training ratios										
	10%	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**Average**	**Rank**
DeepWalk	29.91	33.44	34.53	35.05	35.70	36.80	37.93	36.71	39.00	35.45	4
node2vec	29.27	32.23	33.71	34.52	36.17	36.05	37.53	37.66	37.30	34.94	6
LINE	22.77	24.11	25.11	24.94	25.23	25.59	25.87	26.60	25.44	25.07	8
GraRep	32.59	34.74	34.63	35.21	35.38	36.05	35.10	36.63	37.79	35.35	5
HOPE	30.53	33.61	35.02	35.97	34.90	35.11	36.21	36.20	34.84	34.71	7
SDNE	21.96	21.57	22.05	22.37	23.26	22.59	23.63	23.60	25.31	22.93	9
HRSC	31.54	33.94	34.98	36.84	37.35	38.02	38.78	40.22	41.10	36.97	2
HRTC	31.30	33.79	35.59	36.18	36.95	37.94	38.14	38.63	40.07	36.51	3
HRST	30.81	33.92	36.03	36.85	37.32	38.73	39.28	40.08	41.43	37.16	1

In a word, it was found that the quality of the node representation vectors learnt from HRST was better.

6.5. Link Prediction

In this subsection, the link prediction task was evaluated by the measure AUC [27]. From Tables 4–7, the following observations were obtained as follows:

Table 4. AUC values on GPS.

Methods	Training Ratios								
	60%	**65%**	**70%**	**75%**	**80%**	**85%**	**90%**	**Average**	**Rank**
DeepWalk	0.4308	0.4278	0.4205	0.4583	0.4418	0.4914	0.4831	0.4505	5
node2vec	0.3660	0.3614	0.3808	0.3939	0.3834	0.3958	0.3649	0.3780	8
LINE	0.4575	0.4829	0.4761	0.4562	0.4429	0.4663	0.4574	0.4628	4
GraRep	0.3873	0.3805	0.3882	0.3765	0.3820	0.3857	0.3874	0.3839	7
HOPE	0.3805	0.3676	0.3416	0.2971	0.2794	0.2518	0.2334	0.3073	9
SDNE	0.3262	0.4371	0.4319	0.3157	0.4379	0.3527	0.4540	0.3936	6
HRSC	0.7516	0.7562	0.7488	0.7449	0.7236	0.7325	0.7279	0.7408	1
HRTC	0.6845	0.6428	0.6483	0.6403	0.6216	0.6005	0.5856	0.6319	3
HRST	0.7200	0.7220	0.7241	0.7276	0.6929	0.7071	0.7006	0.7135	2

Table 5. AUC values on MovieLens.

Methods	Training Ratios								
	60%	65%	70%	75%	80%	85%	90%	Average	Rank
DeepWalk	0.7845	0.8129	0.8301	0.8440	0.8729	0.8800	0.9025	0.8467	2
node2vec	0.7078	0.7390	0.7418	0.7696	0.7939	0.8036	0.8296	0.7693	7
LINE	0.8282	0.8242	0.8253	0.8320	0.8365	0.8172	0.8231	0.8266	5
GraRep	0.7290	0.7833	0.7907	0.8121	0.8277	0.8481	0.8544	0.8065	6
HOPE	0.6895	0.7333	0.7203	0.7522	0.7787	0.7986	0.8049	0.7539	8
SDNE	0.4004	0.3511	0.3494	0.3406	0.3433	0.3598	0.4171	0.3660	9
HRSC	0.8714	0.8706	0.8681	0.8644	0.8706	0.8651	0.8528	0.8661	1
HRTC	0.8495	0.8497	0.8351	0.8325	0.8387	0.8281	0.8320	0.8379	3
HRST	0.8377	0.8434	0.8351	0.8268	0.8341	0.8339	0.8297	0.8344	4

Table 6. AUC values on drug.

Methods	Training Ratios								
	60%	65%	70%	75%	80%	85%	90%	Average	Rank
DeepWalk	0.4852	0.4954	0.4934	0.4580	0.4901	0.4638	0.4713	0.4796	6
node2vec	0.4500	0.4525	0.4490	0.4525	0.4329	0.4712	0.4345	0.4489	8
LINE	0.4750	0.4672	0.4636	0.4625	0.4741	0.4523	0.4768	0.4674	7
GraRep	0.5025	0.5089	0.4867	0.5051	0.5557	0.5835	0.5362	0.5255	4
HOPE	0.5055	0.5269	0.4933	0.4690	0.4941	0.4668	0.4271	0.4832	5
SDNE	0.2948	0.4310	0.4454	0.5050	0.5196	0.3536	0.3836	0.4190	9
HRSC	0.7871	0.7774	0.7822	0.7920	0.7747	0.7983	0.8056	0.7882	1
HRTC	0.7153	0.7071	0.7134	0.7134	0.6868	0.7108	0.7240	0.7101	3
HRST	0.7914	0.7697	0.7685	0.7977	0.7586	0.7771	0.8105	0.7819	2

Table 7. AUC values on wordnet.

Methods	Training Ratios								
	60%	65%	70%	75%	80%	85%	90%	Average	Rank
DeepWalk	0.7780	0.8181	0.8305	0.8341	0.8708	0.8765	0.8880	0.8423	4
node2vec	0.7807	0.8242	0.8309	0.8285	0.8519	0.8503	0.8595	0.8323	5
LINE	0.8063	0.8184	0.8056	0.8091	0.8000	0.7938	0.7926	0.8037	6
GraRep	0.7685	0.7742	0.7888	0.7806	0.7958	0.7972	0.7756	0.7830	7
HOPE	0.6902	0.7314	0.7417	0.7403	0.7649	0.7763	0.7700	0.7450	8
SDNE	0.3712	0.5348	0.4784	0.4824	0.4254	0.6159	0.4850	0.4847	9
HRSC	0.8953	0.9079	0.9086	0.9036	0.9093	0.9045	0.9034	0.9047	1
HRTC	0.9030	0.9115	0.9050	0.9016	0.9098	0.9027	0.8912	0.9035	2
HRST	0.8938	0.9014	0.8910	0.8896	0.9026	0.8961	0.8945	0.8956	3

- On the GPS and drug datasets, the average AUC value of HRST was very close to that of HRSC and superior to that of HRTC. On the wordnet dataset, the average AUC value of HRST was almost the same as those of HRSC and HRTC. On the MovieLens dataset, the average AUC values of HRST and HRTC were weaker than those of HRSC and DeepWalk. On the whole, HRST performed better than most baseline methods, which indicated the effectiveness of HRST.
- HRST performed consistently at different training ratios compared with with other baseline methods, which demonstrated its feasibility and robustness.
- HRST almost performed better than other baseline methods without incorporating hyperedges, which verified the assumption that it was good for link prediction to incorporate the hyperedges into the process of hypernetwork representation learning.

In a word, the above observations demonstrated that HRST can obtain high-quality node representation vectors.

6.6. Parameter Sensitivity

The harmonic factors β_1 and β_2 were used to counterweigh the contribution rate among the topology-derived, the set constraint, and the translation constraint models. The training ratio and β_2 were fixed to 50% and 0.5, respectively, and calculated node classification accuracies with a different β_1, assuming that β_1 ranged from 0.1 to 0.9 on MovieLens and wordnet datasets. Figure 4 shows the comparisons of node classification accuracies with a different β_1.

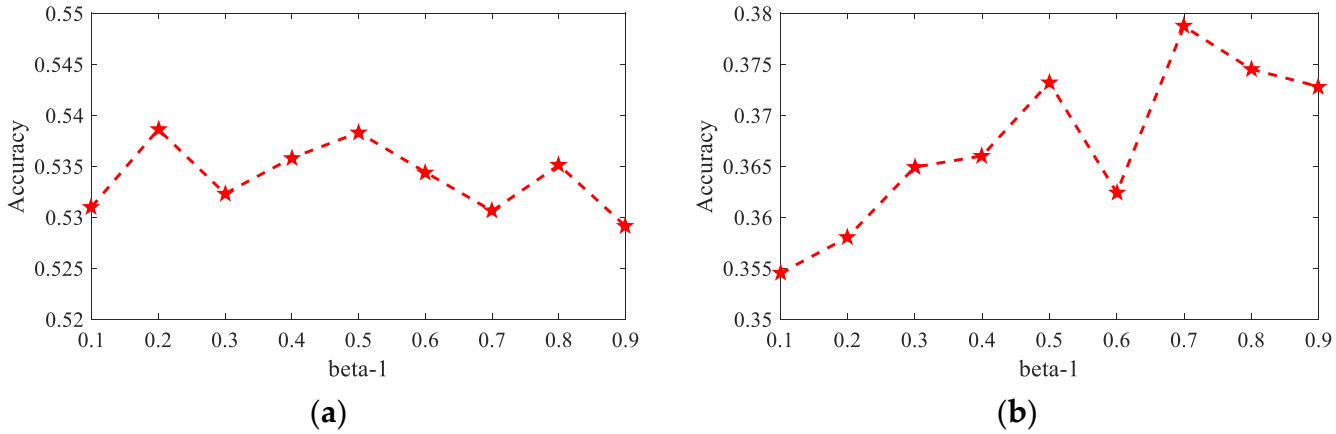

Figure 4. Parameter sensitivity: (**a**) sensitivity on MovieLens; (**b**) sensitivity on wordnet.

As shown in Figure 4, the node classification performance of HRST was not sensitive to the parameter β_1 and demonstrated the robustness of HRST, because the variation ranges of node classification accuracies with a different β_1 were all within 2.5%.

As for MovieLens and wordnet datasets, the best evaluated results in terms of node classifications were achieved at $\beta_1 = 0.2$ and $\beta_1 = 0.7$, respectively.

7. Conclusions

Hypernetwork representation learning can explore the relationships among the nodes and find a universal method to solve practical problems, and it has a wide range of application scenarios, such as trend prediction, personalized recommendation, and other online applications. Therefore, we proposed a hypernetwork representation learning method with common constraints of the set and translation to effectively incorporate the hyperedges into the process of hypernetwork representation learning and regard the learning process of node representation vectors as a joint optimization problem, which was solved by means of the stochastic gradient ascend method. The experimental results demonstrated that our proposed method was almost entirely superior to other baseline methods. Although we carried out the research of the hypernetwork representation learning by means of a transformation strategy from the hypergraph to the graph and tried to incorporate the hyperedges into the process of the network representation learning, some hypernetwork structure information was still lost. Therefore, future research can be carried out regarding two aspects: firstly, continue to try to incorporate the hyperedges into network representation learning methods; secondly, the hypernetwork should no longer be transformed into the conventional network, so that the hyperedges are no longer decomposed, but regarded as a whole to study the hypernetwork representation learning.

Author Contributions: Conceptualization, Y.Z. and H.Z.; methodology, Y.Z. and H.Z.; software, Y.Z.; validation, Y.Z.; formal analysis, Y.Z.; investigation, Y.Z. and H.Z.; resources, Y.Z.; data curation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, Y.Z. and H.Z.; visualization, Y.Z.; supervision, Y.Z. and H.Z.; project administration, Y.Z. and H.Z.; funding acquisition, Y.Z., X.W., and J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant numbers 62166032, 62162053, and 62062059; by the Natural Science Foundation of Qinghai Province, grant numbers 2022-ZJ-961Q and 2022-ZJ-701; by the Project from Tsinghua University, grant number SKL-IOW-2020TC2004-01; and by the Open Project of State Key Laboratory of Plateau Ecology and Agriculture, Qinghai University, grant number 2020-ZZ-03.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ruan, Q.S.; Zhang, Y.R.; Zheng, Y.H.; Wang, Y.D.; Wu, Q.F.; Ma, T.Q.; Liu, X.L. Recommendation model based on a heterogeneous personalized spacey embedding method. *Symmetry* **2021**, *13*, 290. [CrossRef]
2. Wang, M.H.; Qiu, L.L.; Wang, X.L. A survey on knowledge graph embeddings for link prediction. *Symmetry* **2021**, *13*, 485. [CrossRef]
3. Li, Y.H.; Wang, J.Q.; Wang, X.J.; Zhao, Y.L.; Lu, X.H.; Liu, D.L. Community detection based on differential evolution using social spider optimization. *Symmetry* **2017**, *9*, 183. [CrossRef]
4. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD Internatonal Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
5. Grover, A.; Leskovec, J. Node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
6. Tang, J.; Qu, M.; Wang, M.Z.; Zhang, M.; Yan, J.; Mei, Q.Z. Line: Large-scale information network embedding. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 1067–1077.
7. Cao, S.S.; Lu, W.; Xu, Q.K. Grarep: Learning graph representations with global structural information. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management, Melbourne, Australia, 19–23 October 2015; pp. 891–900.
8. Ou, M.D.; Cui, P.; Pei, J.; Zhang, Z.W.; Zhu, W.W. Asymmetric transitivity preserving graph embedding. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1105–1114.
9. Tu, C.C.; Liu, H.; Liu, Z.Y.; Sun, M.S. CANE: Context-aware network embedding for relation modeling. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1722–1731.
10. Tu, C.C.; Zeng, X.K.; Wang, H.; Zhang, Z.Y.; Liu, Z.Y.; Sun, M.S.; Zhang, B.; Lin, L.Y. A unified framework for community detection and network representation learning. *IEEE Trans. Knowl. Data Eng.* **2019**, *31*, 1051–1065. [CrossRef]
11. Zhu, Y.; Zhao, H.X. Hypernetwork representation learning with the set constraint. *Appl. Sci.* **2022**, *12*, 2650. [CrossRef]
12. Agarwal, S.; Branson, K.; Belongie, S. Higher order learning with graphs. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, USA, 25 June 2006; pp. 17–24.
13. Huang, J.; Chen, C.; Ye, F.H.; Wu, J.J.; Zheng, Z.B.; Ling, G.H. Hyper2vec: Biased random walk for hyper-network embedding. In Proceedings of the 24th International Conference on Database Systems for Advanced Applications, Chiang Mai, Thailand, 23–25 April 2019; pp. 273–277.
14. Huang, J.; Liu, X.; Song, Y.Q. Hyper-path-based representation learning for hyper-networks. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 449–458.
15. Tu, K.; Cui, P.; Wang, X.; Wang, F.; Zhu, W.W. Structural deep embedding for hyper-networks. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 426–433.
16. Zhou, D.Y.; Huang, J.Y.; Schölkopf, B. Learning with hypergraphs: Clustering, classification and embedding. In Proceedings of the 19th International Conference on Neural Information Processing Systems, Vancouver, Canada, 4–7 December 2006; pp. 1601–1608.
17. Sharma, K.K.; Seal, A.; Herrera-Viedma, E.; Krejcar, O. An enhanced spectral clustering algorithm with s-distance. *Symmetry* **2021**, *13*, 596. [CrossRef]
18. Bretto, A. *Hypergraph Theory: An Introduction*; Springer Press: Berlin, Germany, 2013; pp. 24–27.
19. Zhang, R.C.; Zou, Y.S.; Ma, J. Hyper-SAGNN: A self-attention based graph neural network for hypergraphs. *arXiv* **2019**, arXiv:1911.02613.
20. Song, G.; Li, J.W.; Wang, Z. Occluded offline handwritten chinese character inpainting via generative adversarial network and self-attention mechanism. *Neurocomputing* **2020**, *415*, 146–156. [CrossRef]
21. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. In Proceedings of the 26th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 2787–2795.

22. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the 26th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 3111–3119.
23. Zheng, V.W.; Cao, B.; Zheng, Y.; Xie, X.; Yang, Q. Collaborative filtering meets mobile recommendation: A user-centered approach. In Proceedings of the 24th AAAI Conference on Artificial Intelligence, Atlanta, GA, USA, 11–15 July 2010; pp. 236–241.
24. Harper, F.M.; Konstan, J.A. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 19. [CrossRef]
25. Wang, D.X.; Cui, P.; Zhu, W.W. Structural deep network embedding. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1225–1234.
26. Xu, J.L.; Han, J.W.; Nie, F.P.; Li, X.L. Multi-view scaling support vector machines for classification and feature selection. *IEEE Trans. Knowl. Data Eng.* **2020**, *32*, 1419–1430. [CrossRef]
27. Wang, Y.G.; Huang, G.N.; Yang, J.J.; Lai, H.D.; Liu, S.; Chen, C.R.; Xu, W.C. Change point detection with mean shift based on AUC from symmetric sliding windows. *Symmetry* **2020**, *12*, 599. [CrossRef]