Article

A Network Representation Learning Model Based on Multiple Remodeling of Node Attributes

Wei Zhang 1,2,3, Baoyang Cui 2,3, Zhonglin Ye 4 and Zhen Liu 4,*

1 School of Computer Science, Shaanxi Normal University, Xi’an 710119, China; zhang_wei@qhnu.edu.cn
2 The State Key Laboratory of Tibetan Intelligent Information Processing and Application, Xining 810008, China; cuibaoyang@stu.qhnu.edu.cn
3 School of Computer, Qinghai Normal University, Xining 810008, China
4 Graduate School of Eegineering, Nagasaki Institute of Applied Science, Nagasaki 851-0123, Japan; zhonglin_ye@snnu.edu.cn
* Correspondence: liu_zhen@campus.nias.ac.jp

Abstract: Current network representation learning models mainly use matrix factorization-based and neural network-based approaches, and most models still focus only on local neighbor features of nodes. Knowledge representation learning aims to learn low-dimensional dense representations of entities and relations from structured knowledge graphs, and most models use the triplets to capture semantic, logical, and topological features between entities and relations. In order to extend the generalization capability of the network representation learning models, this paper proposes a network representation learning algorithm based on multiple remodeling of node attributes named MRNR. The model constructs the knowledge triplets through the textual association relationships between nodes. Meanwhile, a novel co-occurrence word training method has been proposed. Multiple remodeling of node attributes can significantly improve the effectiveness of network representation learning. At the same time, MRNR introduces the attention mechanism to achieve the weight information for key co-occurrence words and triplets, which further models the semantic and topological features between entities and relations, and it makes the network embedding more accurate and has better generalization ability.

Keywords: network representation learning; knowledge modeling; structure modeling; attention; attribute contain

MSC: 68T07; 91D30; 05C82

1. Introduction

With the popularity of Internet applications such as social networks, e-commerce platforms, online news, and search engines, various types of data (e.g., text, images, audio, etc.) are constantly emerging, and people need to extract useful features from these data to help decision-making. In networks, nodes usually represent entities (e.g., people, objects, events, etc.), and the connecting edges between nodes represent relationships between entities. For network data such as social networks, knowledge graphs, etc., node’s text features and network structure features contain rich entity attributes and relationship features between entities, while the connected edges between nodes represent association relationships between entities. Therefore, it has become a hot research direction on how to use node’s text features and network structure features for network representation learning [1–5] in the field of machine learning, whose main goal is to map the nodes and network structures into a low-dimensional embedding space to facilitate learning and reasoning for various downstream tasks. Currently, network representation learning has become an important part of several fields, such as graph
embedding and knowledge graph representation learning. In terms of development history, network representation learning is mainly divided into two stages: matrix factorization-based methods and deep learning-based methods. Early network representation learning methods are mainly based on matrix factorization, i.e., these methods are based on singular value factorization, and these methods are based on probability matrix factorization. Although these methods can learn the low-dimensional representation vectors of nodes, they are usually unable to handle nonlinear relationships in the network well because of the limitations of higher sparsity and lower semantics inherent in matrix factorization. In particular, the higher sparsity of data can bring higher computational complexity and poorer parallelism to traditional machine learning algorithms, which has become a limitation to the use of statistical learning methods for solving the different network data mining tasks. With the use of deep learning (neural networks), network representation learning methods based on deep learning have begun to receive attention from researchers. Deep neural networks have become the mainstream methods in network representation learning, which mainly include methods based on self-encoder [6], methods based on convolutional neural networks, and methods based on recurrent neural networks [7]. On the other hand, there are also some methods based on Graph Convolutional Networks (GCNs) [8], whose main idea is to aggregate the feature of each node with features of its neighboring nodes through convolutional operations, learning the low-dimensional embedding vectors of the nodes.

Most of the current methods for learning network representations are based on local neighboring features of nodes, which allows embedding nodes in a low-dimensional space to obtain a compact and meaningful representation. However, this approach ignores global network structural features and the global node’s textual features, thus potentially leading to missing features or interference from noise. In order to better capture the semantic and contextual features of nodes, researchers have begun to focus on how to utilize node’s textual features and network structural features for network representation learning. In our social network, each node has rich attribute information, and text is a type of attribute. It is one of the main innovations for researchers to make full use of the text attributes of nodes.

On the other hand, knowledge representation learning aims to learn low-dimensional dense representations of entities and relations from structured knowledge bases. Most methods use triplets to capture semantic, logical, and topological features between entities and relations. However, most of the current approaches view the triplet as a constraint item of modeling. Network representation learning uses neural networks to learn low-dimensional dense representations of nodes and edges to reflect the proximity and similarity between nodes and edges. Knowledge representation learning also adopts neural networks to learn entities’ and relations’ representations, such as TransE [9,10], TransH [11], and other methods. Network representation learning also usually adopts neural network-based methods, such as DeepWalk [12], node2vec [13], and others. The integration of the two kinds of representation learning can further enhance the semantic and topological features between entity and relationship.

In order to improve the model of network representation learning, this paper introduces additional features to improve the performance of network representation learning, named MRNR, which models network structural features, text features, and knowledge triplet features in a unified learning framework, which is shown in Figure 1.

Firstly, for the text features, different weights are considered to be contributed to the model. So, the attention mechanism is introduced to give different weights to the co-occurring words in the nodes. Important co-occurring words will be identified, while unimportant co-occurring words will reduce their weights. By that, the high-quality feature inputs can be filtered from the text of the network nodes. Then, the node’s text relationship triplet is constructed to define the relationships between nodes and nodes about the text, i.e., if the text features of two nodes contain common words, it means that the two nodes have text correlation with each other. In addition, the attention mechanism
is added to the triplet, and it can make better use of the semantic features of the knowledge triplets, which can learn the contribution of different triplets to the model and improve the effect of the network representation learning models. Finally, the MRNR algorithm jointly models the structural features, textual features, and triplet features of the network and uses the attention mechanism to optimize the modeling process, which results in the final learned network representation learning vectors having more feature factors, extending the generalization ability of the representation vectors and improving the performance in downstream machine learning tasks. The method can also be applied to a wide range of fields, such as social networks, recommender systems, natural language processing, etc., which has important practical significance.

Figure 1. Learning target of MRNR algorithm model.

In summary, the main contributions of this paper are as follows:

1. An unsupervised network embedding learning model as MRNR is proposed, which is not only able to utilize network structural features and node’s text features but also adds triplet features constructed based on node’s text relationships. Finally, the representation vectors obtained from the learning procedure contain more feature factors and better reflect the multifaceted features of nodes, improving the accuracy and robustness of node representation. By adding the attention mechanism to the triplets and co-occurrence words, different weights can be assigned based on the importance and similarity, improving the expressiveness and differentiation of the node representation.

2. Compared to the existing graph attention network models, the MRNR algorithm proposed in this paper can directly calculate different weights for co-occurrence words and triples in text. This type of attention weight is fine-grained attention information. However, the existing network representation learning algorithms with attention mechanisms often take text features as the initial representation vector of nodes and then adjust the element values in the network representation vector in the neural network. Text features exhibit insufficient participation during the training process.

3. The MRNR algorithm has a reasonable framework and clear objectives, and it is an unsupervised machine-learning method that achieves the goal of modeling multiple features in a unified framework. Compared to existing graph neural networks, the MRNR algorithm proposed in this paper can be applied to unlabeled networks. At the same time, it can also be applied to labeled networks.
2. Related Work

Network representation learning methods mainly include methods based on matrix factorization, methods based on random walk, and methods based on graph convolutional neural network, etc. Random walk-based methods mainly include DeepWalk, node2vec, and LINE [14], etc. The DeepWalk algorithm uses random walk to generate node sequences in the graph and learns the representation vectors of the nodes using the Skip-Gram [12] model or CBOW model. The node2vec algorithm can enhance the learning of local features while retaining the structural features of the network by introducing the weight control of the undirected edges in the random walk. The node2vec algorithm learns the embedding vectors of nodes by introducing the weight control of undirected edges in the random walks to enhance the learning of local features. The SGCN [15] algorithm proposes a directed graph convolutional neural network based on random walks for learning node embeddings in directed graphs. The GCLA [16] algorithm proposes a graph representation learning model based on comparative learning and data augmentation, which improves the robustness and generalization ability of the model through data augmentation, such as random walks on graphs and node deletion. GraRep [17] proposes a graph representation learning method based on global structural features, which captures the global structural features and the representation vectors of different orders by singular value factorization of the adjacency matrix of different orders of the graph, and then the model combines these representation vectors to learn the low-dimensional embedding vectors of the nodes. The HAN [18] algorithm improves the learning of heterogeneous graph representations and proposes a heterogeneous graph embedding learning method based on a multi-head attention mechanism, which can effectively learn the representation vectors of different types of nodes and edges. These methods treat the sequence of nodes as a text sequence and use a Word2Vec-like model to learn the relationships between nodes.

The method based on graph convolutional neural networks utilizes graph structure features to extend convolution operations to graphs, such as GCN, GAT [19], and GraphSAGE [20]. The GCN algorithm proposes a convolutional neural network architecture that can perform convolution operations on a graph, which learns the representation vectors of the nodes in a semi-supervised manner. The GraphSAGE algorithm samples node neighbors using an aggregation operation and learns the nodes using a convolutional neural network. The GAT algorithm is a node representation learning method based on graph neural networks, which models the relationships between nodes through attention mechanisms to improve the performance of graph tasks. The GAT algorithm is a classic graph neural network algorithm that adds attention mechanisms, and it is also a type of network representation learning algorithm. However, the GAT algorithm is a semi-supervised algorithm, and the MRNR algorithm proposed in this paper is an unsupervised algorithm.

These methods can effectively employ semi-supervised mechanisms to learn node representation vectors. Graph Attention Network-based methods introduce an attention mechanism to further improve the model performance, such as AGNN [21], GATNE [22], etc. The AGNN algorithm is a node classification method based on a graph neural network algorithm, which integrates the node’s own features and neighboring features to learn the node representation through an adaptive neighbor aggregation strategy. The GATNE algorithm is a multivariate network representation learning method based on a graph neural network, which can simultaneously consider the features of different types of nodes and edges for representation learning. The DynGCN [23] algorithm proposes a representation learning method based on temporal dynamic graphs, which extends the representation learning of graphs to temporal dynamic scenarios. Dong et al. [24] propose a graph node representation learning model by maximizing the mutual information between the current node and neighbor nodes. The EIGAT [25] algorithm proposes a method of incorporating global features into local attention for knowledge representation learning, which combines global features appropriately into the GAT model family by
using scaled entity importance, which is computed by an attention-based global random walk strategy. The MEGNN [26] algorithm proposes a meta path extraction GNNs for heterogeneous graphs, which combine different bipartite graphs related to edge types into a new trainable graph structure. Sun et al. [27] propose a graph autoencoder algorithm based on a dual decoder, embedding graph topology and node attributes into a compact vector and reconstructing vertex attributes and graph structure simultaneously.

The classical approach in knowledge representation learning is TransE, which proposes an idea based on transformation strategies to achieve the modeling of multi-relationship knowledge graphs by mapping structures and relations into a low-dimensional embedding space. The DistMult [28] algorithm proposes a model based on matrix factorization, which calculates scores between entities and relations by matrix multiplication to achieve the modeling of knowledge graphs. The ConvE [29] algorithm is based on the model of a convolutional neural network, which represents entities and relationships as two-dimensional matrices and uses a convolutional neural network to model node relationships with higher prediction accuracy and interpretability. Yu et al. [30] propose a knowledge graph completeness model that integrates entity descriptions and network structures, and they study how to enhance the completeness of the knowledge graph by using the entity descriptions and the network structures. KANE [31] proposes a knowledge graph embedding model based on a knowledge graph attention network and its attributes to feature enhancement, which captures the higher-order structure and attribute features of knowledge graphs by considering both relation triplets and attribute triplets in a graph convolutional network framework.

Graph comparison learning, as an effective training scheme, can reduce popularity bias and improve noise robustness based on graph tasks. Wei et al. [32] proposed CGI to construct an optimized graph structure by adaptively deleting nodes and edges, which provides a basis for alleviating popular biases. In order to effectively discard information unrelated to downstream recommendations, CGI innovatively integrates information bottlenecks into the multi-view comparative learning process for recommendations. Cai et al. [33] proposed a lightweight and robust graph comparison learning framework named LightGCL, which can alleviate the problems caused by inaccurate comparison signals by utilizing global collaborative relationships. Compared with existing GCL-based methods, the proposed LightGCL has higher training efficiency.

Based on the above background, this paper proposes a network representation learning model that combines textual features, network structural features, and knowledge triplet features, which learns low-dimensional representations of nodes by jointly modeling textual features, network structural features, and knowledge triplet features to better support graph tasks such as node classification and link prediction. The MRNR algorithm proposed in this paper is an unsupervised machine learning algorithm that can be applied to various social network tasks, emotional computing tasks, recommendation systems, etc.

3. Algorithm Design

3.1. Definitions

This section defines some variables that are commonly used in this paper. We use $G = (V, E)$ to define a network, where $V$ denotes the set of nodes, $E$ denotes the set of edges, and $|V|$ denotes the number of nodes. $R_v \in \mathbb{R}^k$ represents the trained network representation vector, which is a $k$-dimensional matrix, and each row in the matrix denotes a node $k$-dimensional network representation vector, where $k$ is much smaller than $|V|$ and $att$ is the attention parameter. $T$ is the label of the nodes $v$. The symbol table is shown in Table 1.
### Table 1. Symbol table.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G = (V, E) )</td>
<td>inputted network</td>
</tr>
<tr>
<td>( V )</td>
<td>set of nodes</td>
</tr>
<tr>
<td>( E )</td>
<td>the set of edges</td>
</tr>
<tr>
<td>(</td>
<td>V</td>
</tr>
<tr>
<td>( R_v )</td>
<td>the trained network representation vector</td>
</tr>
<tr>
<td>( att )</td>
<td>attention parameter</td>
</tr>
<tr>
<td>( T )</td>
<td>labels of the nodes ( v )</td>
</tr>
<tr>
<td>( \log g_c(v) )</td>
<td>the loss of network node relationship modeling</td>
</tr>
<tr>
<td>( \log g_w(v) )</td>
<td>the loss of node’s text word features and attention mechanism learning</td>
</tr>
<tr>
<td>( NEG(v) )</td>
<td>negative sample set of node ( v )</td>
</tr>
<tr>
<td>( L_c(v) )</td>
<td>sampling label of node ( v )</td>
</tr>
<tr>
<td>( v_{nb} )</td>
<td>the context node of node ( v )</td>
</tr>
<tr>
<td>( D )</td>
<td>the set of nodes</td>
</tr>
<tr>
<td>( \sigma(x) )</td>
<td>Sigmoid function</td>
</tr>
<tr>
<td>( C )</td>
<td>node’s sequence corpus</td>
</tr>
<tr>
<td>( \mu )</td>
<td>the learning rate</td>
</tr>
<tr>
<td>( e_u )</td>
<td>the sum of word representation vectors</td>
</tr>
<tr>
<td>( w(u) )</td>
<td>the word representation vectors</td>
</tr>
<tr>
<td>( att(w_j) )</td>
<td>the attention weight of the word ( w_j )</td>
</tr>
<tr>
<td>( h )</td>
<td>head entity</td>
</tr>
<tr>
<td>( t )</td>
<td>tail entity</td>
</tr>
<tr>
<td>( r )</td>
<td>relation entity</td>
</tr>
<tr>
<td>( C(h, r, t) )</td>
<td>knowledge triplet</td>
</tr>
</tbody>
</table>

#### 3.2. MRNR Modeling

In this paper, text is used as the input of the relationship model, and then constructing node relationships, if it contains the same word of a node, the word is regarded as one kind of relationship between two nodes, and the more the number of the same words in the node’s text, the higher the node similarity. In order to adapt to the requirements of large-scale network-embedded learning tasks, a simple and efficient joint learning model needs to be proposed. Therefore, this paper proposes a novel joint learning framework, named MRNR, for network representation, which consists of three parts: network node relationship modeling, the node’s text relationship modeling, and the knowledge triplet relationship modeling. MRNR is improved from two levels. Firstly, an unsupervised network embedding joint learning framework is proposed, which is not only able to use network structure features and node’s text feature features but also adds the knowledge triplet features between nodes, which makes the representation vector obtained by learning contains multiple feature factors to improve the accuracy and robustness of node representation. Secondly, by adding the attention mechanism for co-occurrence words and knowledge triplets, different weights are automatically learned for different co-occurring words and knowledge triplets (text features between nodes are learned two times) to improve the expressiveness and differentiation of the node representation.

With the above improvements, the MRNR model is expected to solve the problem of joint modeling of network structural features, textual features, and knowledge triplets, which can help obtain better-quality representation vectors. The framework of the MRNR algorithm is specifically shown in Figure 2.
As shown in Figure 2, the network node relationship modeling learns the network structure feature representation, the node’s text feature relationship modeling learns the node’s text feature representation, and the triplet relationship modeling learns the node’s triplet relationship feature representation. The model ultimately integrates each node’s neighbor information and text features to obtain better-quality vectors. It should be noted that the knowledge triplet introduced in this paper is essentially a reuse of text features. Therefore, the MRNR algorithm models the node’s textual features by two different methods. At the same time, an attention mechanism is introduced to optimize the model modeling process, which provides the possibility of efficient use of text features.

Network Node Relationship Modeling: in order to have closer vector space distances between node pairs with connected edges and make node pairs with multi-hop edges or no edges have farther vector space distances between them, network node relationship modeling constantly adjusts the values in the node representation vectors.

Node’s textual relationship modeling: relationships between nodes are modeled by considering textual features between nodes. When there are common word features between nodes, the nodes containing this relationship get more learning from the model, which makes these types of nodes have a closer vector space distance between each other. In order to make the node pairs with the same important words get more learning opportunities between them, an attention mechanism is introduced in the node’s text relationship modeling, which enables the training of important words in the model to have higher weights. Ultimately, the parameters learned from the model are more reasonable.

Triplet relationship modeling: This model learns the triplet relationship features between nodes and introduces an attention mechanism to distinguish the importance and participation of different triplets. This allows the model to be given higher weights for important triplets and lower weights for unimportant triplets to affect and optimize the model training process. Triplet relationship modeling is a secondary modeling of text features, which is an efficient modeling of text features within a unified framework.

In summary, the MRNR model obtains and exchanges features between each other’s nodes through shared vectors, which enables the MRNR algorithm to obtain valuable
At the same time, modeling learning process to have a stronger generalization ability in all kinds of tasks. In this paper, we propose a learning model that combines network structural features, text features, and triplet features, and its overall objective function is

\[ L(v) = L_c(v) + L_w(v) + L_r(v) \]

\[ = \sum_{v \in C} (\log g_c(v) + \rho \log g_w(v) + \beta \log g_R(v)). \]

where \( \log g_c(v) \) is part of network node relationship modeling, \( \rho \log g_w(v) \) is part of the node's text word features and attention mechanism learning, \( \beta \log g_R(v) \) is part of the triplet features and attention mechanism learning, and \( \rho \) and \( \beta \) are the reconciliation coefficients that balance the three models. The goal of this paper is to maximize Equation (1), which uses stochastic gradient ascent to update each parameter, where the three parts have the same form in parameter updating. The first term of the above equation is the objective function like the CBOW model, the second term of the above equation is the objective function of the textual features and the attention mechanism, which is essentially a CBOW model adding attention mechanism, and the third term of the above equation is the process of modeling the triplet relationship adding attention mechanism. For the speed of modeling, the optimization method of the second part and the third part still refer to that of the first part.

(1) Network node relationship modeling

Inspired by the CBOW model, this subsection proposes a network node relationship modeling approach, which is essentially a negative sampling-based CBOW model. Where nodes \( w \) are positive samples, and other nodes are negative samples. For the current node \( v \), the context node of node \( v \) is \( v_{nb} \), where node \( v \) is a positive sample and other nodes are negative samples. The negative sample set of node \( v \) is \( NEG(v) \), \( NEG(v) \neq null \). For \( \forall u \in D \), \( D \) is the set of nodes, and the sampling result of the node \( v \) is \( L_c(v) \) and it is defined as follows:

\[ L^v(u) = \begin{cases} 1, & u = v, \\ 0, & u \neq v, \end{cases} \]

where \( L^v(u) \) is 1 for the positive sample, otherwise \( L^v(u) \) is 0.

For node \( v \) and its context node \( v_{nb} \) and its positive sample \( (v, v_{nb}) \), the goal is to maximize the probability as follows:

\[ g_c(v) = \prod_{v_i \in [v] \cup NEG(v)} p(v_i | v_{nb}). \]

And

\[ p(v_i | v_{nb}) = [\sigma(e_v^T \theta v_i)]^{L^v(v_i)} \cdot (1 - \sigma(e_v^T \theta v_i))^{1-L^v(v_i)}. \]

\( \sigma(x) \) is the Sigmoid function. \( e_v \) is the sum of the node vectors in \( v_{nb} \). \( \theta v_i \) is the node \( v_i \) corresponding to a vector to be trained.

Therefore, \( g_c(v) \) can be changed based on Equations (2) and (3):

\[ g_c(v) = \sigma(e_v^T \theta v) \prod_{v_i \in [v] \cup NEG(v)} [1 - \sigma(e_v^T \theta v_i)]. \]

Then, the overall optimization objective function is as follows:

\[ G = \prod_{v \in C} g_c(v). \]

For computational convenience, taking logarithms of \( G \), the negative sampling-based CBOW model objective function is the following logarithmic operation on Equation (5):
\[ L_c(v) = \log G = \log \prod_{v \in C} g_c(v) = \sum_{v \in C} \log g_c(v) \]
\[ = \sum_{v \in C} \sum_{v \in \text{NEG}(v)} \{(1 - L^v(v_i)) \cdot \log[1 - \sigma(e_v^\top \theta^v)] + L^v(v_i) \cdot \log[\sigma(e_v^\top \theta^v)]\}, \tag{7} \]
and
\[ L(v, v_i) = (1 - L^v(v_i)) \cdot \log[1 - \sigma(e_v^\top \theta^v)] + L^v(v_i) \cdot \log[\sigma(e_v^\top \theta^v)], \tag{8} \]
we give the gradient computation of the function \( L(v, v_i) \) with respect to \( \theta^v \) as follows:
\[ \frac{\partial L(v, v_i)}{\partial \theta^v} = \frac{\partial}{\partial \theta^v} \{(1 - L^v(v_i)) \cdot \log[1 - \sigma(e_v^\top \theta^v)] + L^v(v_i) \cdot \log[\sigma(e_v^\top \theta^v)]\}, \tag{9} \]
Utilizing the derivative functions of \( \log \sigma(x) \) and \( \log(1 - \sigma(x)) \) to optimize Equation (8), we obtain
\[ \frac{\partial L(v, v_i)}{\partial \theta^v} = L^v(v_i)[1 - \sigma(e_v^\top \theta^v)]e_v - [1 - L^v(v_i)] \sigma(e_v^\top \theta^v)e_v \]
\[ = \{L^v(v_i)[1 - \sigma(e_v^\top \theta^v)] - [1 - L^v(v_i)] \sigma(e_v^\top \theta^v)\} e_v \]
\[ = [L^v(v_i) - \sigma(e_v^\top \theta^v)]e_v. \tag{10} \]
In Equation (6), \( C \) is the node’s sequence corpus after the nodes have performed random walk, which is optimized using stochastic gradient ascent to obtain the updated equations for each parameter. The updated equation of \( \theta^v \) is
\[ \theta^v := \theta^v + \mu[L^v(v_i) - \sigma(e_v^\top \theta^v)]e_v, \tag{11} \]
where \( \mu \) denotes the learning rate.
Considering the gradient of \( e_v \) in \( L(v, v_i) \), using the \( e_v \) and \( \theta^v \) symmetry, the updated equation of the embedding vector of each node \( v(u) \) in the context can be obtained by
\[ v(u): = v(u) + \mu \sum_{v_i \in \text{NEG}(v)} \{L^v(v_i) - \sigma(e_v^\top \theta^v)\} \cdot \theta^v, \tag{12} \]
where \( u \in v_i \), \( e_v \) is the sum of vectors of each node in the context node. How do we update the representation vectors of each node in the context node? We use the method like Word2Vec, \( \sum_{v_i \in \text{NEG}(v)}[L^v(v_i) - \sigma(e_v^\top \theta^v)] \cdot \theta^v \) is directly contributed to each component to update the representation vector of each node in the context. This paper adopts the same strategy and method. Finally, Equation (6) can be simplified as
\[ L_c(v) = \sum_{v \in C} \sum_{v \in \text{NEG}(v)} \log p(v_i|v_{nb}). \tag{13} \]

(2) Node’s text Relationship Modeling
With reference to the above model, this paper constructs a node’s textual relationship modeling model that maximizes the probability of the occurrence of the target node by modeling the probability of the occurrence of words in the context node. The objective function of the node’s textual relationship modeling is as follows:
\[ L_w(v) = \rho \cdot \sum_{v \in C} \sum_{u \in \text{NEG}(v)} \log(g_u(v)) = \rho \cdot \sum_{v \in C} \sum_{u \in \text{NEG}(v)} \log[\sigma(e_u^\top \theta^v)] + (1 - L^v(u)) \log[1 - \sigma(e_u^\top \theta^v)], \tag{14} \]
and
\[ f_1 = L^v(u) \cdot \log[\sigma(e_u^\top \theta^v)] + (1 - L^v(u)) \log[1 - \sigma(e_u^\top \theta^v)]. \tag{15} \]
In Equation (13), the target node is a positive sample node, and the other nodes are negative sample nodes. Then using the $f_j$ respectively for the $\theta^{vi}$ and $e_u$ to obtain the partial derivation, and the parameter update equation is obtained as

$$\theta^{vi} := \theta^{vi} + \mu [L^{vi}(u) - \sigma(e_u \cdot \theta^{vi})] \cdot e_u.$$  

(16)

The embedding vector for each node in the context of $w(u)$ is updated with the equation

$$w(u) := w(u) + \mu \sum_{v \in \text{NEGB}(v)} [L^{vi}(u) - \sigma(e_u \cdot \theta^{vi})] \cdot \theta^{vi},$$

(17)

where $\mu$ denotes the learning rate. After obtaining the updated forms of $\theta^{vi}$ and $e_u$, the objective function can be optimized iteratively. The weights $\rho$ need to be multiplied into Equation (14) before $\mu$ to balance the network structure model and the text feature model.

When context words act as context nodes, the sum of the context vectors is $e_u$, which is calculated as follows:

$$e_u = \sum_{j=1}^{[S]} \text{att}(w_j) \cdot d_j,$$

(18)

$\text{att}(w_j)$ is the attention weight of the word $w_j$ for the target node, $d_j$ is the representation vector of the word $w_j$, and $[S]$ is the number of words in the node’s texts.

The attention function ($\text{att}$) is calculated as follows:

$$\text{att}(w_j) = \frac{\exp (d_j \cdot w(u))}{\sum_{k=1}^{[S]} \exp (d_k \cdot w(u))}.$$  

(19)

Attention weights are computed from representation vectors of co-occurrence words and representation vectors of target nodes.

(3) Triplet relationship modeling

Inspired by the translation mechanism of the TransE algorithm, we can represent the interaction between nodes as a translation operation in the space. TransE algorithm, as a classical model of relationship embedding in the knowledge graph, is based on the principle of mapping the entities and relationships to the representation vectors in the low-dimensional space, which predicts the relationships between the entities through the calculation of the distance between the vectors. Meanwhile, the TransE algorithm provides an interpretable and easy-to-understand relationship representation. In this paper, we use the co-occurring words in the text of the network nodes to construct the triplet. If the text of two nodes contains a co-occurrence word, then the word is treated as the relationship between the two nodes. For example, if the text of node 1 and node 2 both contain “Network”, then the node triplet (Node1, Network, Node2) can be constructed. The triplets constructed in this way are not independent of each other but are related to each other in the whole network. These relationships can portray the semantic features of the target network structure to a certain extent. We refer to these interrelated triplets as knowledge triplets.

Define the target word ($w_i$) of the knowledge triplet as follows:

$$C(w_i) = \{(r, t) | \forall r, t, (w_i, r, t) \in G\},$$

(20)

For example, if the text of Node 1 is “Network Representation Learning” and the text of Node 2 is “Network Embedding Learning”, we can construct two triplet relationships: (Node1, Network, Node2) and (Node1, Learning, Node2). By using a knowledge triplet containing the target word $w_i$, the likelihood of $w_i$ appearing in a given context can be predicted. As relational triplets continue to appear, the representation vectors of the head entity ($h$), relationship ($r$), and tail entity ($t$) in the triplet network are continuously
adjusted. It can be found that the knowledge triplet incorporates the co-occurring features of the text of the network nodes, which helps the model to express the semantic similarity of the text between the network nodes.

In the TransE model, the representation learning model needs to define a rating function $f(h, r, t) = \|h + r - t\|$ for users to assess the credibility of a triplet. The higher the score, the greater the probability that the triplet is a factual feature and the higher the credibility. For the triplet set $(h, r, w_i)$ constructed by the target word $w_i$, if the triplet is a factual feature, then $h + r \approx w_i$, and it shows that the corresponding vector $h + r$ should be closer to $w_i$. In this paper, knowledge triplets are introduced into the objective function of modeling, and the probability of existing factual triplets is

$$f(h, r, t) = P((h, r, t)|C(h, r, t)).$$

We factorize this equation with the conditional probability equation as follows:

$$f(h, r, t) = P(h|C(h, r, t)) \cdot P(t|C(h, r, t), h) \cdot P(r|C(h, r, t), h, t).$$

The first part of the above equation, $P(h|C(h, r, t))$, represents the occurrence conditional probability of the head entity $h$ under the given knowledge triplet $(h, r, t)$. Due to the direct correlation between the head entity and its neighbors, $P(h|C(h, r, t)) \approx P(h|C_n(h))$, inspired by reference [9], can be represented as a Softmax function:

$$P(h|C_n(h)) = \frac{\exp(\text{att} \cdot d_n(h, C_n(h)))}{\sum_{h'} \exp(\text{att} \cdot d_n(h', C_n(h)))},$$

where $d_n(., .)$ is used to measure the degree of association between $h'$ and $h$, and $d_n(h', C_n(h))$ is defined as follows:

$$d_n(h', C_n(h)) = \frac{1}{|C_n(h)|} \sum_{(r, t) \in C_n(h)} \|h' \odot r \odot t\|,$$

where “$\odot$” is the Hadamard product, $|C_n(h)|$ denotes the number of knowledge triplets, and $\|\cdot\|$ denotes first-order normal form and second-order normal form. It has been shown in literature 3 that combining operations on vectors of entities and relations is better using multiplication than addition. So in this paper, we use the Hadamard product to measure the similarity degree between them. Finally, attention mechanisms are added to all triplet relationships in $h'$ and $C_n(h)$, and the learned attention parameter is used to adjust the weight of each triplet relation.

The second part of Equation (22), $P(t|C(h, r, t), h)$, represents the conditional probability of the occurrence of tail entity $(t)$ under the conditions of having head entity $(h)$ and knowledge triplet $(h, r, t)$. This paper uses co-occurrence words between entities to measure the degree of association between head and tail entities and sets $P(t|C(h, r, t), h) = P(t|C_n(h, t), h)$, where $C_n(h, t)$ represents the knowledge triple of entity pair $h$ and $t$, and $P(t|C_n(h, t), h)$ represents a softmax function.

$$P(t|C_n(h, t), h) = \frac{\exp(\text{att} \cdot d_n(t, C_n(h, t)))}{\sum_{t' \in E, t' \neq t} \exp(\text{att} \cdot d_n(t', C_n(h, t)))},$$

where $d_n(., .)$ is used to measure the degree of association between $t'$ and $C_n(h, t)$, $d_n(t', C_n(h, t))$ is defined as follows:

$$d_n(t', C_n(h, t)) = \frac{1}{|C_n(h, t)|} \sum_{(r, t) \in C_n(h, t)} \|h \odot r \odot t'\|,$$

$|C_n(h, t)|$ denotes the number of knowledge triplets. Finally, the attention mechanism is added to all the triplet relations in the $C_n(h, t)$, and the learned attention parameters are used to adjust the weights of each triplet relation, which is regarded as the degree of association between $t'$ and $C_n(h, t)$.
P(r|C(h, r, t), h, t) represents the occurrence conditional probability of relationship (r) under the conditional of having head entity (h), tail entity (t), and knowledge triplet C(h, r, t). Since the entities h and t have been determined, the knowledge triplet has been introduced. Therefore, the knowledge triplet C(h, r, t) in P(r|C(h, r, t), h, t) can be omitted and formalized as

\[ P(r|h, t) = \frac{\exp (\text{att} \cdot d_n(h, r, t))}{\sum_{r' \in E} \exp (\text{att} \cdot d_n(h, r', t))}, \tag{27} \]

where \( d_n(\ldots) \) is used to measure the degree of association between \( r \) and entity pairs \((h, t)\), defined as follows:

\[ d_n(h, r, t) = ||h \circ r \circ t||. \tag{28} \]

The knowledge triplet features are brought into the model, Equation (21) can be approximated as

\[ f(h, r, t) \approx P(h|C_n(h)) \cdot P(t|C_n(h, t), h) \cdot P(r|h, t). \tag{29} \]

The MRNR model uses the triplet establishment probability as a triplet scoring function to maximize the joint probability of all triplets. We define the objective function as follows:

\[ g_R(v) = \prod_{(h, r, t) \in G} f(h, r, t). \tag{30} \]

The objective function is still optimized by negative sampling so that the likelihood function of the triplet model is

\[ g_R(v) = \prod_{(h, r, t) \in G} f(h, r, t) \approx \prod_{(h, r, t) \in G} [\sigma(d_n(h, C_n(h)))] \cdot \prod_{(h, r, t) \in G} [\sigma(d_n(t, C_n(h, t)))] \cdot \prod_{(h, r, t) \in G} [\sigma(d_n(h, r, t)) \cdot \prod_{r' \in R^\sim} \sigma(-d_n(h, r', t))], \tag{31} \]

where \( h', r' \) and \( t' \) are negative samples of the head entity, relation, and tail entity in the triplet, respectively. \( R^\sim, R^\sim, \) and \( T^\sim \) are the sets of these types of negative samples, respectively. \( h' \) is obtained by replacing the head entity of a triplet with any other entity. \( \sigma(x) \) is the Sigmoid function. To facilitate parameter optimization, Equation (30) is converted to negative logarithmic form.

\[ L(v) = \beta \log g_R(v) = \beta \sum_{(h, r, t) \in G} \log f(h, r, t) \approx -\beta \sum_{(h, r, t) \in G} [\log \sigma(d_n(h, C_n(h))) + \log \sigma(d_n(t, C_n(h), t)) + \log P(r|h, t)] \]

\[ + \beta \sum_{(h, r, t) \in G} [\log \sigma(d_n(h, r, t)) \cdot \prod_{r' \in R^\sim} \log \sigma(-d_n(h, r', t))]. \tag{32} \]
Equation (31) is the objective function of network node triplet relationship modeling. The calculation method for the update equation of each parameter is still similar to the modeling of network node relationships and node's text relationships, which is optimized using the random gradient rise method.

4. Experimentation

4.1. Data Sets

In order to evaluate the effectiveness of MRNR, this paper uses three citation network datasets, which include the academic network datasets Citeseer, DBLP, and SDBLP. SDBLP removes the nodes where the number of citations is less than 3 for DBLP. After performing operations such as deleting isolated nodes from the original dataset, the statistical data of the dataset is shown in Table 2.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Number of Nodes</th>
<th>Number of Sides</th>
<th>Average Degree</th>
<th>Training Set</th>
<th>Validation Set</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citeseer</td>
<td>4610</td>
<td>5923</td>
<td>2.57</td>
<td>1333</td>
<td>667</td>
<td>2610</td>
</tr>
<tr>
<td>DBLP</td>
<td>17,725</td>
<td>105,781</td>
<td>11.926</td>
<td>3333</td>
<td>1667</td>
<td>12,725</td>
</tr>
<tr>
<td>SDBLP</td>
<td>3119</td>
<td>39,516</td>
<td>25.339</td>
<td>666</td>
<td>334</td>
<td>3119</td>
</tr>
</tbody>
</table>

It can be found that the density of the network is getting higher. We use Citeseer, DBLP, and SDBLP to simulate three types of network datasets with different attributes to verify the generalization ability of the MRNR algorithm.

4.2. Introduction to the Contrast Algorithm

(1) DeepWalk

DeepWalk originates from the Word2Vec algorithm. The DeepWalk algorithm is the most classic network embedding learning algorithm based on neural networks, and many subsequent network embedding learning algorithms are mainly based on the DeepWalk algorithm.

(2) LINE

LINE is an unsupervised learning graph embedding algorithm proposed by Tang et al. in 2015, which aims to embed nodes into low-dimensional vector spaces for analysis in machine learning and data mining tasks. The LINE algorithm has two main variants: The first one is based on the neighboring relationship between nodes, which learns the similarity between neighboring nodes by maximizing their first-order similarity. The second one is based on the neighboring relationships, which learns the second-order similarity of nodes by maximizing the similarity between second-order neighboring nodes. Ultimately, the LINE algorithm combines the first-order and second-order similarities of the nodes to produce the final embedding vectors of the nodes.

(3) GraRep

GraRep is a graph embedding algorithm based on higher-order proximity. GraRep is different from traditional first-order proximity (DeepWalk, LINE, etc.), which considers higher-order proximity between nodes. The model uses the features of nodes in their neighbors with different hop counts to construct higher-order adjacency matrices and performs Singular Value Factorization (SVD) of these matrices to generate embedding vectors. The GraRep's advantage is the ability to capture complex relationships between nodes using higher-order proximity between nodes to generate richer embedding representations.

(4) MFDW
The MFDW algorithm uses the idea of DeepWalk based on matrix factorization to compute the representation vector of each node, which is a simplified algorithm of DeepWalk.

(5) Text Feature (TF)

The Text Feature algorithm is a graph embedding algorithm based on text features. The Text Feature algorithm first converts the text features of each node into a text feature matrix using a text feature extraction technique, and then it maps the text feature matrix into a low dimensional space using the SVD algorithm to obtain the representation vector of each node.

(6) TADW

TADW is a graph embedding algorithm that combines topology and node attributes. TADW is a matrix factorization algorithm that combines a matrix ($M$) with the text matrix ($T$) for joint factorization, which improves the expressiveness and generalization of the embedding vectors.

4.3. Experimental Setup

This paper evaluates the MRNR algorithm and the comparison algorithm using a network node classification task with Liblinear as the baseline classifier. In order to test the generalization ability of the algorithms, we set 9 training set scales from 0.1 to 0.9 and use the remaining nodes as the test set. We set the network embedding vectors obtained by all algorithms uniformly to 100 dimensions. We set the length of random walks as 40, the number of random walks as 10, the window size as 5, negative sampling as 5, the minimum node frequency as 5, and the learning rate of the neural network as 0.05. In the experiment, we will repeat the training 10 times and take the average of the experimental results as the final result. We fix the $\rho$ and $\beta$ as 0.5 in the node classification task.

4.4. Analysis of Experimental Results

In this paper, we use real network datasets Citeseer, DBLP, and SDBLP as evaluation datasets. We will take 10% to 90% of the data as the training set and the remaining data as the test set. Table 3 shows the accuracy of the comparison algorithm and MRNR algorithm in network node classification tasks under 3 datasets and 9 different training set ratios.

Based on the given three datasets, the experimental results show that the MRNR algorithm achieves the best performance at all proportions, and its accuracy is higher than other compared methods at different data proportions. For example, for the Citeseer dataset, MRNR achieves an accuracy of 79.93% at 90% data proportion, while the highest accuracy of other comparison methods is only 64.30% (MFDW). On the DBLP and SDBLP datasets, the accuracy of MRNR achieves 85.33% and 85.63% at 80% data proportion, which are higher than the highest accuracy of other comparison methods.

Taking the Citeseer dataset as an example, it can be found that the MRNR algorithm achieves a better performance under different proportions of the training set. Especially under the proportion of 90% of the training set, MRNR’s accuracy reaches 79.93%, which is much higher than other algorithms. In the comparison algorithms, MFDW is a matrix factorization form of DeepWalk and its accuracy reaches 64.30%; the LINE algorithm has the worst network node classification performance. The MRNR algorithm optimizes the process of modeling network structural features and adds node’s text features and triplet features, which outperforms the DeepWalk algorithm.
Table 3. Comparison results on the node classification task.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Methods of Comparison</th>
<th>Percentage of Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10%</td>
</tr>
<tr>
<td>Citeseer</td>
<td>DeepWalk</td>
<td>55.89</td>
</tr>
<tr>
<td></td>
<td>LINE</td>
<td>42.64</td>
</tr>
<tr>
<td></td>
<td>GraRep</td>
<td>39.38</td>
</tr>
<tr>
<td></td>
<td>MFDW</td>
<td>57.62</td>
</tr>
<tr>
<td></td>
<td>Text Feature</td>
<td>57.69</td>
</tr>
<tr>
<td></td>
<td>MRNR</td>
<td>75.76</td>
</tr>
</tbody>
</table>

In the DBLP dataset, the classification performance of DeepWalk is slightly inferior to LINE and GraRep, and the MFDW algorithm has a better classification performance than DeepWalk. As the proportion of the training set increases, the network node classification performance of Text Feature is increasingly superior to DeepWalk. The MRNR still performs best in high-proportion training sets, and its accuracy reaches 84.89%.

In the SDBLP dataset, the MRNR algorithm also performs well with different proportions of training sets. Especially in a high proportion of training sets, the MRNR algorithm’s accuracy reaches 84.53%. In the comparison algorithms, the classification performance of the GraRep algorithm and the MRNR algorithm both obtained better performance due to the dense networks.

Overall, the MRNR algorithm performs well in different datasets and training set ratios and even outperforms the comparison algorithms in some cases. This shows that the algorithm proposed in this paper has good potential for node classification problems. We compare representation learning methods using network structures and representation learning methods using text features. The experimental results showed that MRNR significantly improved its network node classification performance after adding text features. The MRNR algorithm proposed in this paper exhibits excellent node classification performance on sparse networks such as Citeseer. The main reason is that MRNR introduces text features of nodes and assigns different weights to different co-occurrence words and triples in the text, allowing more features to participate in learning and training during model modeling. On dense datasets such as SDBLP, the network structure features are very rich, so the performance differences between different algorithms are relatively smaller.

As shown in Figure 3, the span of network node classification accuracy values of the five comparative algorithms on the Citeseer and DBLP datasets is greater than their values on the SDBLP dataset. On the SDBLP dataset, the network node classification accuracies of the five comparison algorithms show a significant upward trend. However, on the Citeseer and DBLP datasets, the accuracy curves show a relatively slow upward trend. The main reason for this phenomenon is that on sparse networks, there is a large
variability in the features acquired between different algorithms. If a certain algorithm can acquire features that more fully reflect the network structure, its network classification performance becomes better. On such sparse networks, the network representation learning algorithms based on joint learning can compensate for the insufficient training problem due to the sparsity of the network. Regarding dense networks, different algorithms can obtain effective network structure features from sufficiently connected edges, and thus, the differences in classification performance are smaller.

Figure 3. Performance comparisons of six algorithms on three datasets.

4.5. Network Embedding Visualization

The main purpose of network representation visualization is to examine whether the trained representation vectors exhibit a significant clustering phenomenon, which can be considered whether the network representation has learned the community features of the network. If the community division based on network representations is more accurate, it has better reliability in network node classification tasks. This paper randomly selects four classes of nodes from the Citeseer dataset and 150 nodes from each class. At the same time, we use the t-SNE algorithm to degrade and visualize the network representation vectors. The specific results are shown in Figure 4.
Figure 4. Visualization results of the 3 datasets on DeepWalk and MRNR (Different colors represent different labels of nodes.)

From Figure 4, it can be found that this paper visualizes some of the network representation vectors obtained from DeepWalk and the MRNR model trained on the three datasets: Citeseer, DBLP, and SDBLP. From Table 2, it can be seen that DeepWalk performs poorly in network representation classification tasks; therefore, DeepWalk also exhibits the same results in visualization tasks. The MRNR model shows significant improvement in network representation and classification performance in three datasets. Therefore, the representation vectors of all three types of nodes exhibit obvious clustering phenomena and clustering boundaries.

4.6. Case Analysis

In order to verify the feasibility of the model, the target node is randomly selected in the Citeseer dataset. The text of this node is “Quantum Field Theory as Dynamical System”. We get the three nodes with the highest similarity to the target node. Then, we analyze the correlation between the text of the target node and the text of the three nodes. As shown in Table 4, the DeepWalk model only considers the network structure feature similarity, and it does not consider the node’s text feature similarity. Therefore, the texts of the three nodes do not all contain the same words as the target node text. The TADW model and MRNR consider both the network structure feature and the node’s text features, so the returned similar nodes all have the same co-occurring words. This case cannot explain which algorithm has better machine learning performance, and it only explains that the MRNR algorithm is able to model both network structural features and textual features.
Table 4. Case analysis (Citeseer dataset).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Vertex Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepWalk</td>
<td>Statistics Localization Regions and Modular Symmetries in Quantum Field Theory</td>
</tr>
<tr>
<td></td>
<td>Extensions of Conformal Nets and Super Selection Structures</td>
</tr>
<tr>
<td></td>
<td>Modular Covariance Pet Spin and Statistics</td>
</tr>
<tr>
<td>TADW</td>
<td>Remarks on Causality in Relativistic Quantum Field Theory</td>
</tr>
<tr>
<td></td>
<td>Modular Covariance Pet Spin and Statistics</td>
</tr>
<tr>
<td></td>
<td>An Algebraic Spin and Statistics Theorem</td>
</tr>
<tr>
<td>MRNR</td>
<td>Higher-dimensional Algebra and Topological Quantum Field Theory</td>
</tr>
<tr>
<td></td>
<td>Statistics Localization Regions and Modular Symmetries in Quantum Field Theory</td>
</tr>
<tr>
<td></td>
<td>Quantum Field Theory as Eigenvalue Problem Manuscript</td>
</tr>
</tbody>
</table>

5. Summary

In this paper, we propose a network representation learning algorithm (MRNR) that jointly models network structure features, node’s text features, and knowledge triplet features. Its goal is to import textual features as supervision information in the training process based on structural feature modeling, letting the node representation vectors obtained learn similar textual features. In the process of network modeling, if the structural relationships of the nodes are only considered, the learned representation vectors ignore the semantic relationships between the nodes. In order to consider the semantic relationships between nodes, the text features and knowledge triplet features are introduced into the proposed model. Meanwhile, we introduce an attention mechanism in order to involve valuable features, resulting in the more valuable features having higher weights. In different machine learning tasks, the machine learning performance of the MRNR algorithm based on the remodeling of node attributes is significantly better than the network representation learning algorithm using a single feature. For texts and triplets, adding attention mechanisms based on network structural features can improve the machine-learning performance of network representation. The MRNR algorithm proposed in this paper is currently unable to handle the situation of missing node’s text features. Therefore, the MRNR algorithm will mainly solve this type of problem, and at the same time, new text feature embedding algorithms will be studied in the future.

Author Contributions: Methodology, Z.Y.; Supervision, Z.L.; Writing — original draft, W.Z. and B.C. All authors have read and agreed to the published version of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This article is supported by the National Key Research and Development Program of China (No. 2020YFC1523300), the Innovation Platform Construction Project of Qinghai Province (No. 2022-ZJ-T02) and Research Fund of Qinghai Normal University: 2023QZR007.

Data Availability Statement: Data available on request.

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

References
Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.