Neural Computing and Applications (2022) 34:9087–9101
https://doi.org/10.1007/s00521-022-06930-1

ORIGINAL ARTICLE

Multi-level disentanglement graph neural network

Lirong Wu[1,2] · Haitao Lin[2] · Jun Xia[2] · Cheng Tan[2] · Stan Z. Li[2]

Received: 10 August 2021 / Accepted: 4 January 2022 / Published online: 25 January 2022

Abstract
Real-world graphs are generally generated from highly entangled latent factors. However, existing deep learning methods for graph-structured data often ignore such entanglement and simply denote the heterogeneous relations between entities as binary edges. In this paper, we propose a novel *Multi-level Disentanglement Graph Neural Network* (MD-GNN), a unified framework that simultaneously implements edge-level, attribute-level, and node-level disentanglement in an end-to-end manner. MD-GNN takes the original graph structure and node attributes as input and outputs multiple disentangled relation graphs and disentangled node representations. Specifically, MD-GNN first disentangles the original graph structure into multiple relation graphs, each of which corresponds to a latent and disentangled relation among entities. The input node attributes are then propagated in the corresponding relation graph through a multi-hop diffusion mechanism to capture long-range dependencies between entities, and finally the disentangled node representations are obtained through information aggregation and merging. Extensive experiments on synthetic and real-world datasets have shown qualitatively and quantitatively that MD-GNN yields truly encouraging results in terms of disentanglement and also serves well as a general GNN framework for downstream tasks. Code has been made available at: https://github.com/LirongWu/MD-GNN.

Keywords Graph neural networks · Disentanglement · Relation learning · Semi-supervised learning

1 Introduction

The purpose of disentanglement is to decompose an entity, such as a feature vector, into several interpretable components to better understand and explain the behavior of a learned model. Recently, there have been many methods proposed to solve the disentanglement problem with promising results. For example, Variational AutoEncoder (VAE) [13] constrains the distribution of latent features to be Gaussian and generates disentangled representations. However, most previous efforts focused on the disentanglement for Convolutional Neural Networks (CNNs) [2, 7], and few endeavors have been made on the disentanglement of irregular non-Euclidean domains, such as structural graph data.

In many real-world applications, including chemical molecules, social networks, and citation networks, data can be naturally modeled as graphs. Recently, Graph Neural Networks (GNNs), especially Graph Convolutional Networks (GCN) [14], have demonstrated their powerful potential to solve graph-related problems, such as community detection [20, 32] and anomaly detection [25]. DisenGCN [24], a pioneering work in graph disentanglement, focuses on producing independent latent features for *node-level disentanglement*, but without treatment of the underlying relations between entities. These relations are in many cases heterogeneous, but entangled together and denoted merely as a single bare binary-valued edge. However, edges often contain rich relation information, not just binary indicators of structural connectivity, which motivates us to implicitly uncover latent relations between entities via *edge-level disentanglement*. More importantly, the attributes of each entity are also highly entangled and usually aggregated and transformed as a whole. However, the attributes are generally associated with different relations to indicate what type of proximity exists between entities, which helps us to explain the interactions between entities for better *attribute-level disentanglement*.

✉ Lirong Wu
wulirong@westlake.edu.cn

[1] College of Computer Science and Technology, Zhejiang University, Hangzhou 310058, Zhejiang, China

[2] AI Lab, School of Engineering, Westlake University, Hangzhou 310024, Zhejiang, China

Figure 1 shows a motivating example to illustrate the concept of disentanglement. It can be seen that all four users share common interests in different aspects of Artificial Intelligence (AI), but their interactions have distinctly different focuses. For example, the interaction between John and Ally is mainly about "*Detection, Tracking, ImageNet*", reflecting their common interest in the topic of "*Computer Vision*", while John and Ben interact more on "*Reinforcement Learning*". Without such fine-grained relations and attributes, it is difficult to accurately characterize nodes solely from binary graph structure, and this hinders both model interpretability and performance on downstream tasks.

In this paper, we propose a novel *Multi-level Disentanglement Graph Neural Network* (MD-GNN) framework to simultaneously achieve edge-level, attribute-level, and node-level disentanglement in an end-to-end manner. MD-GNN decomposes each binary-valued edge into a feature vector, with *each dimension corresponding to a disentangled relation subspace*. Meanwhile, the attention mechanisms and orthogonal constraints are applied to help locate the attributes associated with each relation space in the input. Moreover, to capture long-range dependencies between nodes, we introduce a relation diffusion mechanism to expand the receptive field to multi-hop neighbors in each relation space *within a single layer*. Finally, features are aggregated and transformed on individual relation space to produce new features for each node, and all derived features from each relation space are concatenated to produce block-wise disentangled node features.

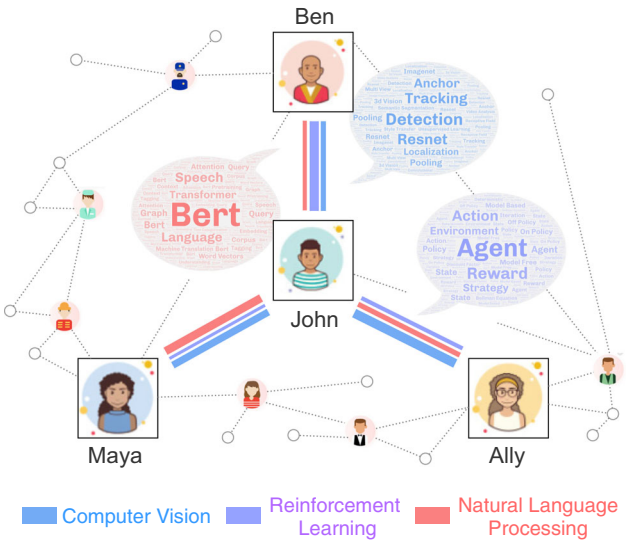


Fig. 1 A motivating example of our work. We take common interests between users as relations (marked by lines with different colors and widths) and the interaction frequency of textual content as attributes (visualized as point clouds)

Our main contributions are summarized as follows: (1) **Multi-level Disentanglement.** We are the first to provide explicit mathematical definitions for graph disentanglement, including edge-level, attribute-level, and node-level disentanglement, and propose a unified framework to implement all three levels of disentanglement. (2) **Relation-based Diffusion.** Most previous efforts [16, 34] on multi-hop message-passing are based on a given binary structure or attention mechanism. In contrast, MD-GNN incorporates the multi-hop diffusion mechanism for message passing *based on the disentangled relation spaces*, which provides a novel perspective for message-passing. (3) **Comparative Evaluation.** The proposed MD-GNN is evaluated on both synthetic and real-world datasets and numerous visualizations are provided to demonstrate the excellent disentanglement performance of MD-GNN.

2 Related work

2.1 Graph neural networks

Graph neural networks (GNN) are a family of neural networks that are widely used for graph representation learning. A general GNN framework involves two key computations for each node v_i at every layer: (1) AGGREGATE operation: aggregating messages from neighborhood $\mathcal{N}_i$; (2) UPDATE operation: updating node representation from its representation in the previous layer and aggregated messages. Considering a L-layer GNN, the formulation of the l-th layer is as follows:

$$\begin{aligned}\mathbf{m}_i^{(l)} &= \text{AGGREGATE}^{(l)}\left(\{\mathbf{h}_j^{(l-1)} : v_j \in \mathcal{N}_i\}\right)\\ \mathbf{h}_i^{(l)} &= \text{UPDATE}^{(l)}\left(\mathbf{h}_i^{(l-1)}, \mathbf{m}_i^{(l)}\right)\end{aligned} \tag{1}$$

where $1 \le l \le L$, $\mathbf{h}_i^{(l)}$ is the embedding of node v_i in the l-th layer, and $\mathbf{h}_i^{(0)} = \mathbf{x}_i$ is the input feature. For node-level tasks, the node representation $\mathbf{h}_i^{(L)}$ can be directly used for downstream tasks. However, for graph-level tasks, an extra READOUT function is usually required to aggregate node features to obtain a graph-level representation $\mathbf{h}_g$, as follows:

$$\mathbf{h}_g = \text{READOUT}\left(\{\mathbf{h}_i^{(L)} \mid v_i \in \mathcal{V}\}\right) = \frac{1}{|\mathcal{V}|}\sum_{i\in\mathbb{V}} \mathbf{h}_i^{(L)} \tag{2}$$

Recent years have witnessed a surge of interest in handling graph-related tasks with Graph Neural Networks (GNNs). There are two categories of GNNs: Spectral GNNs and Spatial GNNs. The spectral GNNs define convolution kernels in the spectral domain based on the graph signal processing theory [31]. For example, GCN-Cheby [5] uses

the polynomial of the Laplacian matrix as the convolution kernel, and Graph Convolutional Networks (GCN) [14] can be considered as a first-order approximation of GCN-Cheby with a self-loop mechanism. Besides, GraphHeat [38] designs a more powerful low-pass filter through heat kernel. Moreover, GWNN [39] replaces the eigenvectors with wavelet bases so as to further improve the efficiency of the model. Generally, spectral methods have good interpretability for graph signal processing, but lack generalization [9].

The spatial GNNs focus on the design of aggregation functions. For example, GraphSAGE [9] employs a generalized induction framework to efficiently generate node embeddings for previously unseen data by using known node feature information. Graph Attention Networks (GAT) [14] extends the idea of GCN by introducing the attention mechanism. Unlike APPNP [16], which incorporates personalized PageRank in the aggregation function, GPRGNN [4] proposes a new Generalized PageRank GNN that adaptively learns edge weights to jointly optimize the feature extraction and topological information regardless of the level of graph heterophily. Moreover, MAGNA [35] proposes a principled way to incorporate multi-hop context information into every layer of GNN attention computation by diffusing the attention scores across the network. Some other classical GNN variants include CensNet [11], AdaLNet [18] and KrylovNetc [23]. For more detailed reviews on GNNs, please refer to the survey [44].

Recently, many neural networks have been proposed with expressive power beyond the 1-WL test [3, 17, 26, 42]. However, these papers introduce extra and domain-specific components beyond standard message-passing GNNs. For example, the learned embeddings of P-GNN [42] are tied with random anchor-sets, and thus are not applicable to node/graph level tasks that require deterministic node embeddings. To address this problem, ID-GNNs [41] develops a class of message passing GNNs and show that GNNs, after incorporating inductive identity information, can surpass the expressive power of the 1-WL test while maintaining benefits of efficiency, simplicity, and broad applicability.

2.2 Disentanglement learning

The task of disentanglement learning has recently been an important research topic toward interpretable AI. The purpose of disentanglement is to decompose an entity, such as a feature vector, into several interpretable components to better understand and explain the behavior of a learned model. In recent years, many approaches have been proposed for learning disentangled representations based on deep neural networks and have achieved promising results with better robustness and interpretability [22]. In contrast to earlier attempts that relied on hand-crafted variables [36, 37], most recent works are based on the autoencoder [7, 10, 13, 29] or generative model [2]. The autoencoder-based methods generally constrain the latent feature generated from the encoder to make it independent in each dimension. For example, Variational AutoEncoder (VAE) [13] constrains the distribution of latent features to be Gaussian. In contrast, the work of [29] disentangles latent features by ensuring that each block of latent features cannot be predicted from the rest. Besides, DSD [7] swaps some of the latent features twice to achieve semi-supervised disentanglement. For the generative model, extra information is introduced during the generation. For example, InfoGAN [2] adds the class code to the model and maximizes the mutual information between the generated data and the class code. Despite many previous efforts in CNN-based disentanglement, disentanglement learning poses great challenges and there are still many unexplored problems in the GNN domain. More importantly, it's not easy to apply existing strategies directly to GNN due to its non-Euclidean property.

2.3 Graph disentanglement learning

Numerous methods have been proposed to deal with the heterogeneous relations between nodes. For example, the works of DisenGCN [24] and IPGDN [21], as pioneering attempts, achieve node-level disentanglement through neighbor routines that divide the neighbors of a node into several mutually exclusive parts. Following the idea of neighbor routines, ADGCN [43] further proposes an adversarial regularizer that improves the separability between different neighbor components, thus restricting interdependence among components. In particular, GAT applies a multi-head attention mechanism to prune irrelevant neighbors and discover intrinsic relations in the graph, which can also be considered as a special kind of edge-level disentanglement. The closest work to us is FactorGCN [40], which performs edge-level disentanglement by taking into account global-level topological semantics, such as higher-order relations. However, our advantages over FctorGNN are: (1) attribute-level disentanglement; (2) a unified end-to-end framework for all three levels of disentanglement; (3) explicit mathematical definitions; (4) replacing multi-layer disentanglement with one-layer relation diffusion. Another work similar to ours is GCAT [19], which proposes a channel-aware attention mechanism enabled by edge textual contents when aggregating information and implements this mechanism in a graph autoencoder framework. However, the textual content between nodes is usually hardly accessible in practice, i.e., the relational topological semantics of the graph is underlying, which limits the application of GCAT, which is

Table 1 Functional capability of different methods

	VAE	DisenGCN/ADGCN	GAT	FactorGCN	MDGNN (ours)
Ability to process node features	Yes	Yes	Yes	Yes	Yes
Ability to process graph structure	No	Yes	Yes	Yes	Yes
Node-level disentanglement	Yes	Yes	No	Yes	Yes
Graph-level disentanglement	No	No	Yes	Yes	Yes
Attribute-level disentanglement	No	No	No	No	Yes
Three-level disentanglements in a unified framework	No	No	No	No	Yes

exactly the problem that graph disentanglement aims to address. From a practical point of view, relation learning *with and without* textual contents are two completely different research directions. While similar in motivation, the experimental setup, datasets, and evaluation protocols of this paper are completely different from those of GCAT.

Despite the promising results of the previous works, there are still many problems left to be solved. **First**, the current relation disentanglement is performed with all input attributes without locating the attributes corresponding to specific relations, i.e., *attribute-level disentanglement has not yet been fully explored*. **Second**, the three closely related tasks of edge-level, attribute-level, and node-level disentanglement have never been tackled in a unified framework. **Last**, despite being repeatedly mentioned, *the mathematical definitions of graph disentanglement are still ambiguous and have not been clarified so far*. The distinctive features of MD-GNN in comparison with related methods are summarized in Table 1, where VAE can be seen as a special version of node-level disentanglement (without graph structure). MD-GNN enables all three levels of disentanglement, for which no other methods can achieve.

3 Methodology

3.1 Preliminary

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ be a given attribute graph, where $\mathcal{V}$ is the set of N nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathcal{X}$ is the set of N node attributes. Each node $v \in \mathcal{V}$ is associated with an attribute vector $\mathbf{x}_v \in \mathcal{X}$, where $\mathbf{x}_v \in \mathbb{R}^{d_{in}}$ with d_{in} being the dimension of input attributes. Besides, each edge $e_{u,v} \in \mathcal{E}$ denotes a connection between node u and node v. Next, we will state the mathematical definition on all three levels of graph disentanglement and then devise a well-thought-out instantiation for it. Unless particularly specified, the notations used in this paper are illustrated in Table 2.

The assumptions basis for this paper is that the disentangled relations between entities should be node-independent without hierarchical properties, which is more compatible with real-world graph data. For example, if two people are "friends", they should be disentangled as "friends" at different hierarchical levels of the relation graph. A complete graph disentanglement includes three levels: edge-level, attribute-level, and node-level disentanglement. We first define three mapping functions: $\psi : \mathcal{E} \to \mathcal{R}$, $\Omega : \mathcal{X} \to \mathcal{M}$ and $\phi : \mathcal{V} \to \mathcal{H}$ for them. (1) The edge-level disentanglement can be defined as a mapping $\psi : \mathcal{E} \to \mathcal{R}$, where $\mathbf{R}_{u,v} = \psi(e_{u,v}) \in \mathbb{R}^K$ is the disentangled relation vector for edge $e_{u,v}$, i.e., there are K latent relations to be disentangled. (2) The attribute-level disentanglement can be defined as a mapping $\Omega : \mathcal{X} \to \mathcal{M}$, where $\mathbf{M}_u = \Omega(\mathbf{x}_u) = \left[\mathbf{m}_{u,1}; \mathbf{m}_{u,2}; \ldots; \mathbf{m}_{u,K}\right] \in \mathbb{R}^{d_{in} \times K}$ denotes the disentanglement for attribute $\mathbf{x}_u$, and $(\mathbf{M}_u)_{i,k}$ describes the correlation of i-th element on the attribute vector $\mathbf{x}_u$ with relation k $(1 \le k \le K)$. In practice, it is the global consistency (rather than local node-specific correlations) between relations and attributes that we want to reveal[1], which means there exists a global attribute mask matrix $\mathbf{M} = [\mathbf{m}_1; \mathbf{m}_2; \ldots; \mathbf{m}_K]$ holding for all nodes. (3) The node-level disentanglement can be defined as $\phi : \mathcal{V} \to \mathcal{H}$, where $\mathbf{h}_u = \phi(u) \in \mathbb{R}^{d_{out}}$ is the disentangled features for node u, with d_{out} being the dimension of $\mathbf{h}_u$. Specifically, we would like $\mathbf{h}_u$ to be composed of K independent components, i.e., $\mathbf{h}_u = \left[\mathbf{h}_{u,1}, \mathbf{h}_{u,2}, \ldots, \mathbf{h}_{u,K}\right]$. The k^{th} component $\mathbf{h}_{u,k} \in \mathbb{R}^{\frac{d_{out}}{K}}$ describes the aspect of node u about relation k.

In this paper, we propose a novel instantiation, *Multi-level Disentanglement Graph Neural Network*, to achieve all three levels of graph disentanglement in a unified framework in an end-to-end manner. As shown in Fig. 2, MD-GNN guides the information aggregation and merging among nodes via latent relations revealed by relation learning and diffusion steps.

[1] Different from image data, node attributes in graphs are usually characterized as vectors with each dimension representing a specific meaning. This suggests that the correlations between attributes and relations should be node-independent.

Table 2 Notations used in this paper

Notations	Descriptions
$\mathbb{R}^m$	m-dimensional Euclidean space
$a, \mathbf{a}, \mathbf{A}$	Scalar, vector, matrix
$\mathcal{G}$	A graph
$\mathcal{V}$	The set of nodes in graph $\mathcal{G}$
$\mathcal{E}$	The set of edges in graph $\mathcal{G}$
K	Relation number
S	Iteration step
L	Layer number
$\mathbf{I}_N$	Identity matrix of dimension N
$\mathcal{N}_i$	1-hop Neighborhood set of node v_i
d_{in}	Dimension of input node attribute
d_m	Dimension of hidden node features
F	Dimension of relation subspace
$\mathbf{R}_k \in \mathbb{R}^{N\times N}$	A relation graph w.r.t relation k
$\mathbf{A}_k \in \mathbb{R}^{N\times N}$	A diffused relation graph w.r.t relation k
$\mathbf{x}_i \in \mathbb{R}^{d_0}$	Feature vector of node v_i
$\mathbf{h}_{i,k}^{(l)} \in \mathbb{R}^F$	Hidden node features of node v_i w.r.t relation k in the l-th layer
$\mathbf{h}_i^{(l)} \in \mathbb{R}^F$	Disentangled node feature of node v_i in the l-th layer
$\mathbf{m}_i \in \mathbb{R}^{d_{in}}$	Attribute mask of node v_i
$\mathbf{M}$	Global attribute mask matrix
$\lvert\cdot\rvert$	The length of a set
$\odot$	Element-wise multiplication operation
$\Vert$	Concatenation

3.2 Relation learning step

Given an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, we aim to learn a relation vector $\mathbf{R}_{u,v} = g(\mathbf{x}_u, \mathbf{x}_v) \in \mathbb{R}^K$ between nodes u and v if there exists an edge $e_{u,v} \in \mathcal{E}$. To solve this problem, we propose to conduct relation learning in a low-dimensional subspace, done by multiplying the input attributes of nodes with linear transformation matrices $\{\mathbf{W}_k\}_{k=1}^K$ where $\mathbf{W}_k \in \mathbb{R}^{d_{in}\times F}$ and F is the dimension of the subspace. In addition, in order to locate the relation-specific attributes in the input, we design a set of *learnable* attribute maskers parameterized by $\{\mathbf{m}_k\}_{k=1}^K$, where $\mathbf{m}_k \in \mathbb{R}^{d_{in}}$ masks out the node attributes irrelevant to relation k, as follows:

$$\mathbf{h}'_{k,i} = (\mathbf{m}_k \odot \mathbf{x}_i)\mathbf{W}_k \tag{3}$$

Then, the learning of relation graphs is formulated as

$$\mathbf{R}_{k,i,j} = \frac{1}{1 + e^{-\mathbf{a}_k^T[\mathbf{h}'_{k,i} \Vert \mathbf{h}'_{k,j}]}} \tag{4}$$

where $1 \le i, j \le N$, $1 \le k \le K$, and $\Vert$ denotes the concatenation operation. $\mathbf{a}_k \in \mathbb{R}^{1\times 2F}$ is the attention coefficient with respect to relation k, similar to that of GAT [33]. However, different from most previous forms of attention-based GNNs that normalize the attention coefficients among all the neighbors of nodes, our proposed model directly normalizes the attention score to [0, 1] with activation function $sigmoid(\cdot)$. The motivation behind this is twofold: (1) it endows the relation learning process with more flexibility, making it free from the constraints of neighborhood connectivity and helping to learn more essential and intrinsic relation graphs; (2) normalization prevents too large or small attention scores from

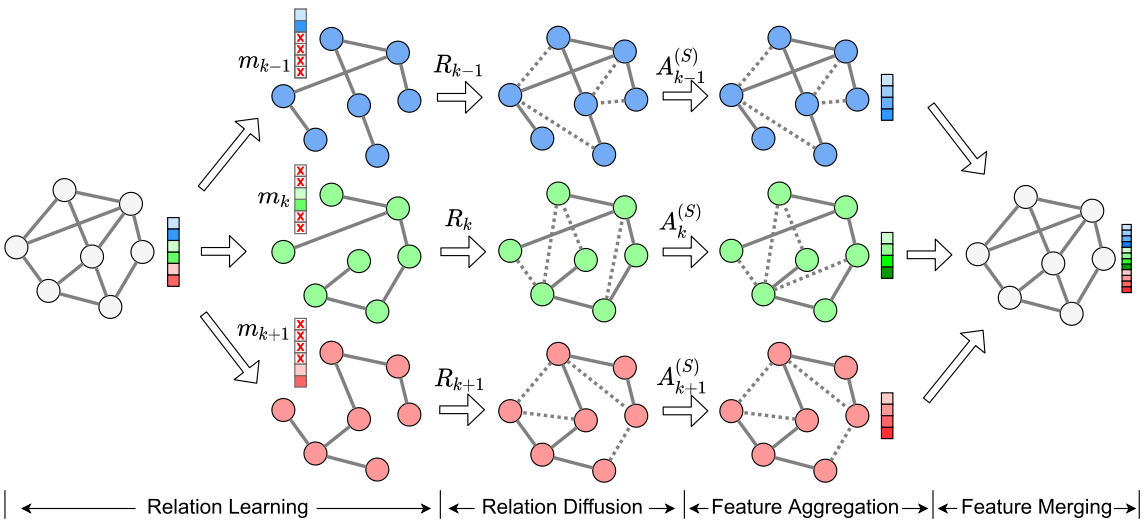


Fig. 2 Illustration of the architecture of the proposed MD-GNN. The blue, green and red in the figure represent different relation graphs and their corresponding node attributes and hidden features

dominating the optimization process, which may lead to gradient explosion or vanishing.

In our framwork, $\{\mathbf{m}_k\}_{k=1}^K$ are defined as a set of *learnable* parameters, with each $\mathbf{m}_k$ corresponding to one specific relation k. However, without other constraints, the learned $\{\mathbf{m}_k\}_{k=1}^K$ may fail to locate relation-specific attributes due to non-sufficient distinguishability. Therefore, we impose an orthogonality constraint [1] to guide them to focus on different aspects:

$$\mathcal{L}_o = \left\|\mathbf{M}^\top\mathbf{M} \odot (\mathbb{1} - \mathbf{I}_N)\right\|_{\mathrm{F}}^2 \tag{5}$$

where $\mathbf{M} = [\mathbf{m}_1; \mathbf{m}_2; \ldots; \mathbf{m}_K]$, $\mathbf{I}_N$ is a identity matrix, and $\mathbb{1}$ is a matrix composed of values 1. The underlying motivation behind this constraint is that, first, a certain relation is generally related to only some specific attributes; for example, words such as "*vocabulary*" and "*language*" may be more relevant to natural language processing and less relevant to computer vision. Second, the difference between the attributes corresponding to different relations is greater than the similarity, and the definition of word "*attributes*" inherently implies the intention to distinguish different things.

Once all the relation graphs are computed, one relation k now can be represented by a specific relation graph $\mathbf{R}_k$. However, without any other constraints, some generated relation graphs may contain similar structures, degrading the disentanglement performance and capacity of the model. More importantly, it is not easy to directly maximize the gap between various relation graphs due to the non-Euclidean property of graph structure. Therefore, we first derive a graph descriptor $\mathbf{D}_k$ for each relation graph $\mathbf{R}_k$ and *ensure that the descriptor is related only to the graph structure and not to the node features*, and then we impose constraints by maximizing the gap between graph descriptors. First, we obtain the graph descriptors by the following:

$$\mathbf{D}_k = f\left(\mathcal{P}\left(\mathcal{A}(\mathbf{R}_k, \mathbf{Z})\right)\right) \tag{6}$$

where $\mathbf{Z} = (\mathbf{z}_1; \mathbf{z}_2; \cdots; \mathbf{z}_N) \in \mathbb{R}^{N\times KF}$ and $\mathbf{z}_i = \|_{k=1}^K h'_{i,k}$ $(1 \le i \le N)$ with $\|$ being the concatenation operation. $\mathcal{A}$ is a two-layer graph autoencoder [15] which takes $\mathbf{Z}$ and relation graph $\mathbf{R}_k$ as inputs, and generates new features for each node, $\mathcal{P}(\cdot)$ is a READOUT function defined in Eq.2 that performs global average pooling for all nodes, and $f(\cdot)$ is a fully connected layer used to generate graph-specific descriptor $\mathbf{D}_k$. Note that all the relation graphs $\{\mathbf{R}_k\}_{k=1}^K$ share the same input node features $\mathbf{Z}$, thus the distinguishability of descriptors depends only on the distinction of relation graph structures. The following discriminative loss is then designed to maximize the differences between different descriptors, given by

$$\mathcal{L}_d = -\sum_{i=1}^{K-1}\sum_{j=i+1}^{K} ||\mathbf{D}_i - \mathbf{D}_j||_2^2 \tag{7}$$

3.3 Relation diffusion step

Different from FactorGCN [40], which stacks multiple disentanglement layers to enlarge the receptive field to multi-hop neighbors, we propose a relation diffusion mechanism that expands the receptive field to multi-hop neighbors in each relation space *within a single layer* to capture the long-range dependencies between nodes. First, we extend the learned one-hop relation matrix $\mathbf{R}_k$ into a multi-hop relation matrix $\mathbf{H}_k$ by:

$$\mathbf{H}_k = \sum_{i=0}^{\infty} \theta_i \mathbf{R}_k^i \tag{8}$$

where $\theta_i = \alpha(1-\alpha)^i$ with teleport probability $\alpha \in (0, 1]$ satisfies $\sum_{i=0}^{\infty}\theta_i = 1$, $\theta_i > 0$, and $\theta_i > \theta_{i+1}$. The powers of relation matrix, $\mathbf{R}_k^i$, give us the number of relation paths from node u to node v of length up to i in the relation graph $\mathbf{R}_k$. In practice, the computation of Eq. 8 is not trivial, as it involves the powers of the matrix. To solve this problem, we approximate $\mathbf{H}_k = \sum_{i=0}^{\infty}\theta_i\mathbf{R}_k^i$ by a sequence of iterations, as follows:

$$\mathbf{A}_k^{(S+1)} = (1-\alpha)\mathbf{R}_k\mathbf{A}_k^{(S)} + \alpha\mathbf{A}_k^{(0)}, \mathbf{A}_k^{(0)} = \mathbf{I}_N \tag{9}$$

Theorem 1 $\lim_{S\to\infty}\mathbf{A}_k^{(S)} = \mathbf{H}_k$.

Proof Let $S > 0$ be the total iteration steps, and the result of the S-th iteration is as follows:

$$\begin{aligned}
\mathbf{A}_k^{(S)} &= (1-\alpha)\mathbf{R}_k\mathbf{A}_k^{(S-1)} + \alpha\mathbf{A}_k^{(0)} \\
&= (1-\alpha)^2\mathbf{R}_k^2\mathbf{A}_k^{(S-2)} + (1-\alpha)\alpha\mathbf{R}_k\mathbf{A}_k^{(0)} + \alpha\mathbf{A}_k^{(0)} \\
&= \cdots \\
&= (1-\alpha)^S\mathbf{R}_k^S\mathbf{A}_k^{(0)} + \alpha\sum_{i=0}^{S-1}(1-\alpha)^i\mathbf{R}_k^i\mathbf{A}_k^{(0)} \\
&= \left((1-\alpha)^S\mathbf{R}_k^S + \alpha\sum_{i=0}^{S-1}(1-\alpha)^i\mathbf{R}_k^i\right)\mathbf{A}_k^{(0)} \\
&= (1-\alpha)^S\mathbf{R}_k^S + \alpha\sum_{i=0}^{S-1}(1-\alpha)^i\mathbf{R}_k^i
\end{aligned}$$

Since $\alpha \in (0, 1]$ and $\mathbf{R}_{k,i,j} \in (0, 1)$, we are therefore able to prove that $(1-\alpha)^S\mathbf{R}_k^S$ converges to 0 when $S \to \infty$. Finally, we can get

$$\lim_{S\to\infty}\mathbf{A}_k^{(S)} = \lim_{S\to\infty}(1-\alpha)^S\mathbf{R}_k^S + \alpha\sum_{i=0}^{S-1}(1-\alpha)^i\mathbf{R}_k^i$$
$$= \lim_{S\to\infty}\alpha\sum_{i=0}^{S-1}(1-\alpha)^i\mathbf{R}_k^i = \sum_{i=0}^{\infty}\alpha(1-\alpha)^i\mathbf{R}_k^i$$

where $\alpha(1-\alpha)^i = \theta_i$, thus

$$\lim_{S\to\infty}\mathbf{A}_k^{(S)} = \sum_{i=0}^{\infty}\alpha(1-\alpha)^i\mathbf{R}_k^i = \sum_{i=0}^{\infty}\theta_i\mathbf{R}_k^i = \mathbf{H}_k$$

We have proved theoretically that $\mathbf{A}_k^{(S)}$ converges to the value of $\mathbf{H}_k = \sum_{i=0}^{\infty}\theta_i\mathbf{R}_k^i$ as the total iteration step $S \to \infty$. The strategy for taking values of S is given in Sect. 4.4. Note that while the idea of personalized PageRank has been adopted in graph neural networks by APPNP [16], we would like to emphasize our differences: (1) we perform *relation-based* diffusion rather than based on binary graph structure like APPNP; (2) three associated tasks of relation learning, relation diffusion and transformation are completely separate in our framework, rather than entangled together like APPNP; (3) we exploit the multi-hop mechanism for better exploration on the inherent relations embedded in the graph, not just to help with feature extraction. □

3.4 Feature aggregation and merging step

Once the relation diffusion is completed, we can obtain a number of diffused relation graphs $\{\mathbf{A}_k^{(S)}\}_{k=1}^K$. Then, in the feature aggregation step, we aggregate the features in each diffused relation graph $\mathbf{A}_k^{(S)}$ accordingly to learn relation-specific representations. Specifically, the new node representations are generated by taking the weighted sum of its neighbors, formulated as:

$$\mathbf{h}_{i,k}^{(l+1)} = \sigma\left(\frac{\sum_{j\in\mathcal{N}_{i,k}}\mathbf{A}_{k,i,j}^{(S)}\mathbf{h}_j^{(l)}}{\sqrt{|\mathcal{N}_{i,k}||\mathcal{N}_{j,k}|}}\mathbf{W}^{(l,k)}\right) \tag{10}$$

where $\mathbf{h}_{i,k}^{(l+1)}$ represents the representation of node i that are pertinent to relation k in $(l+1)$ layer. In the diffused relation graph $\mathbf{A}_k^{(S)}$, $\mathcal{N}_{i,k}$ is the S-hop neighbors of node i, $\mathbf{A}_{k,i,j}^{(S)}$ is the weighting coefficient from node i to j, and $\mathbf{W}^{(l,k)}$ is a linear transformation matrix.

The learned features from different relation space is merged to produce block-wise disentangled features:

$$\mathbf{h}_i^{(l+1)} = \|_{k=1}^K\mathbf{h}_{i,k}^{(l+1)} \tag{11}$$

where $\mathbf{h}_i^{(l+1)}$ is the disentangled feature of node i in $(l+1)$ layer. In Sect. 4.2, we demonstrate the effectiveness of node-level disentanglement through the correlation analysis of the learned disentangled features.

Algorithm 1 Algorithm for the MD-GNN framework

Input: Feature Matrix: $\mathbf{X}$; Edge Set: $\mathcal{E}$; Relation Number: K; Iteration Steps: S; Layer Number: L; Teleport Probability: α; Loss Weights: β and λ; Epoch: E.
Output: Disentangled Node Representations: $\{\mathbf{h}_i\}_{i=1}^N$; Relation-specific Attribute Maskers: $\{\mathbf{m}_k\}_{k=1}^K$.

Initialize parameters $\{\mathbf{m}_k, \mathbf{a}_k, \mathbf{W}_k, \{\mathbf{W}^{(l,k)}\}_{l=1}^L\}_{k=1}^K$.
for *epoch* $\in \{0, 1, \cdots, E\text{-}1\}$ **do**
  for *relation* $k \in \{0, 1, \cdots, K\text{-}1\}$ **do**
    # Relation Learning Step
    Calculate relation graph $\mathbf{G}_k$ by Eq. 3 and Eq. 4.
    Initialize $\mathbf{A}_k^{(0)} = I_N$.
    for $s \in \{0, 1, \cdots, S\text{-}1\}$ **do**
      # Relation Diffusion Step
      Diffuse relations $\mathbf{A}_k^{(s+1)} = (1-\alpha)\mathbf{R}_k\mathbf{A}_k^{(s)} + \alpha\mathbf{A}_k^{(0)}$ by Eq. 9.
    end for
    for *layer* $l \in \{0, 1, \cdots, L\text{-}1\}$ **do**
      # Feature Aggregation and Merging Step
      Aggregate features in the graph $\mathbf{A}_k^{(S)}$ at $(l+1)$ layer by Eq. 10;
      Merge features $\mathbf{h}_i^{(l+1)} = \|_{k=1}^K\mathbf{h}_{i,k}^{(l+1)}$ at $(l+1)$ layer by Eq. 11.
    end for
    Output disentangled features $\{\mathbf{h}_i^{(L)}\}_{i=1}^N$.
  end for
  # Calculate Losses
  Calculate orthogonal loss $\mathcal{L}_o$ by Eq. 5;
  Calculate discriminate loss $\mathcal{L}_d$ by Eq. 6 and Eq. 7;
  Calculate task loss $\mathcal{L}_t$ and total loss $\mathcal{L}$ by Eq. 12;
  Update paramters $\{\mathbf{m}_k, \mathbf{a}_k, \mathbf{W}_k, \{\mathbf{W}^{(l,k)}\}_{l=1}^L\}_{k=1}^K$.
end for
return disentangled node features $\{\mathbf{h}_i^{(L)}\}_{i=1}^N$ and learnable relation-specific attribute maskers $\{\mathbf{m}_k\}_{k=1}^K$.

3.5 Architecture

FactorGCN stacks multiple disentanglement layers and sets the number of relation graphs at different layers as hyperparameters to achieve hierarchical disentanglement. However, this may lead to three potential problems: (1) Excessive hyperparameters affect the generality and scalability, and how to set the different number of relation graphs for each layer requires further exploration. (2) Learning the relation graph for each layer separately brings an excessive computational burden. (3) The assumptions basis for hierarchical disentanglement does not always hold true in the real world, because the relations between entities are *constant without hierarchical properties*. For example, if two people are "friends," they should be disentangled as "friends" in different levels of the relation graph. To solve these problems, we perform relation learning and diffusion only *once*, and then perform L-layer information aggregation and merging based on the diffused

relation graph $\{\mathbf{A}_k^{(S)}\}_{k=1}^K$. The total loss of MD-GNN is defined as

$$\mathcal{L} = \mathcal{L}_t + \beta * \mathcal{L}_o + \lambda * \mathcal{L}_d \tag{12}$$

where $\mathcal{L}_o$ and $\mathcal{L}_d$ is the orthogonal loss and discriminate loss proposed in Eqs. 5 and 7. β and λ are the weights to balance these two losses. $\mathcal{L}_t$ is the task-specific loss, which may be taken to be binary cross-entropy for the multi-label graph classification task, Mean Absolute Error (MAE) for the graph regression task, and cross-entropy for the multi-class node classification. The pseudo-code of the proposed MD-GNN is summarized in Algorithm 1. Omitting the dimensionality to simplify the notation, the computational burden of the model mainly comes from three parts: (1) relation learning ($\mathcal{O}(KN)$); (2) relation diffusion ($\mathcal{O}(KSE)$); and (3) information aggregation ($\mathcal{O}(KE)$), with a total complexity of $\mathcal{O}(K(N+E+SE))$. Since K and S is usually <10 in practice, the complexity is linearly related to the number of nodes $|\mathcal{V}|$ and edges $|\mathcal{V}|$, in the same order as other GCN and GAT variants.

4 Experiments

4.1 Experimental setups

In this section, we show the effectiveness of the proposed MD-GNN qualitatively and quantitatively on both synthetic and real-world datasets, and provide an ablation study on its various components.

4.1.1 Datasets

The effectiveness of the proposed MD-GNN is evaluated on five datasets. The first one is a synthetic dataset containing a fixed number of predefined graphs as ground-truth relation graphs. The second one is ZINC dataset [12] built from molecular graphs. The other three are widely used citation datasets including Cora [30], Citeseer [8], and PubMed [27]. A brief description of datasets is given in Table 3. Next, we introduce these datasets in detail, especially how to generate the synthetic dataset.

Synthetic dataset The synthetic dataset contains 10000 graphs, with 7000 for training, 1000 for validation, and 2000 for testing. The task for this dataset is *multi-label graph classification*. To generate this synthetic dataset, we first generate $K = 6$ predefined graphs that are well-known graphs like grid-2d graph, balanced-tree graph, and hypercube graph, from which we select 3 ground-truth relation graphs and merge them as one sample (mixed graph) as shown in Fig. 3. The type of the ground-truth graph that the mixed graph generates from is taken as the graph label. Each node in the mixed graph is associated with an D-dimensional attribute vector $C = [c_1, c_2, \ldots, c_D]$, with the attribute c_i ($\frac{D}{K} * (k-1) < i \leq \frac{D}{K} * k$) corresponding to a specific relation k ($1 \leq k \leq K$). For each attribute c_i ($1 \leq i \leq D$), if c_i is associated with relation k, its value will be sampled from the Gaussian distribution $N(k, \sigma^2)$. In the paper, we have $D = 30$ and $\sigma = 5.0$.

ZINC dataset The ZINC dataset contains 12000 graphs, with 10000 for training, 1000 for validation, and 1000 for testing. The task of this dataset is *graph regression*, where we regress the constrained solubility properties of molecular graphs. The types of bonds (edges) between atoms (nodes) are provided as ground truths to evaluate the disentanglement performance.

Cora, Citeseet, and PubMed datasets These three real-world datasets are commonly used for node classification. We use them to demonstrate that MN-GNN may well-serve as a general GNN framework, even putting aside its excellent disentanglement capability. In addition to the dataset splitting consistent with the [14] (denoted as *pub*), we follow the splitting strategy in [18] to experiment with fewer and harder label rates. We evaluate with 3%, 1% and 0.5% labeled data in training set on Cora, 1%, 0.5% and 0.3% labeled data on Citeseer, and 0.1%, 0.05% and 0.03% labeled data on PubMed. For these three datasets, we randomly select 50% samples for validation and the rest for testing.

4.1.2 Baselines

To demonstrate the powerful disentanglement capability of MD-GNN, we compare it with three state-of-the-art

Table 3 Dataset statistic information

Dataset	#Node per graph	#Relation	#Attribute per node	train/val/test	Task
Synthetic	30	6	30	7000/1000/2000	multi-label graph classification
ZINC	9-37	3	28	10000/1000/1000	graph regression
Cora (*public*)	2708	-	1433	140/500/1000	multi-class node classification
Citeseer (*public*)	3327	-	3703	120/500/1000	multi-class node classification
PubMed (*public*)	19717	-	500	60/500/1000	multi-class node classification

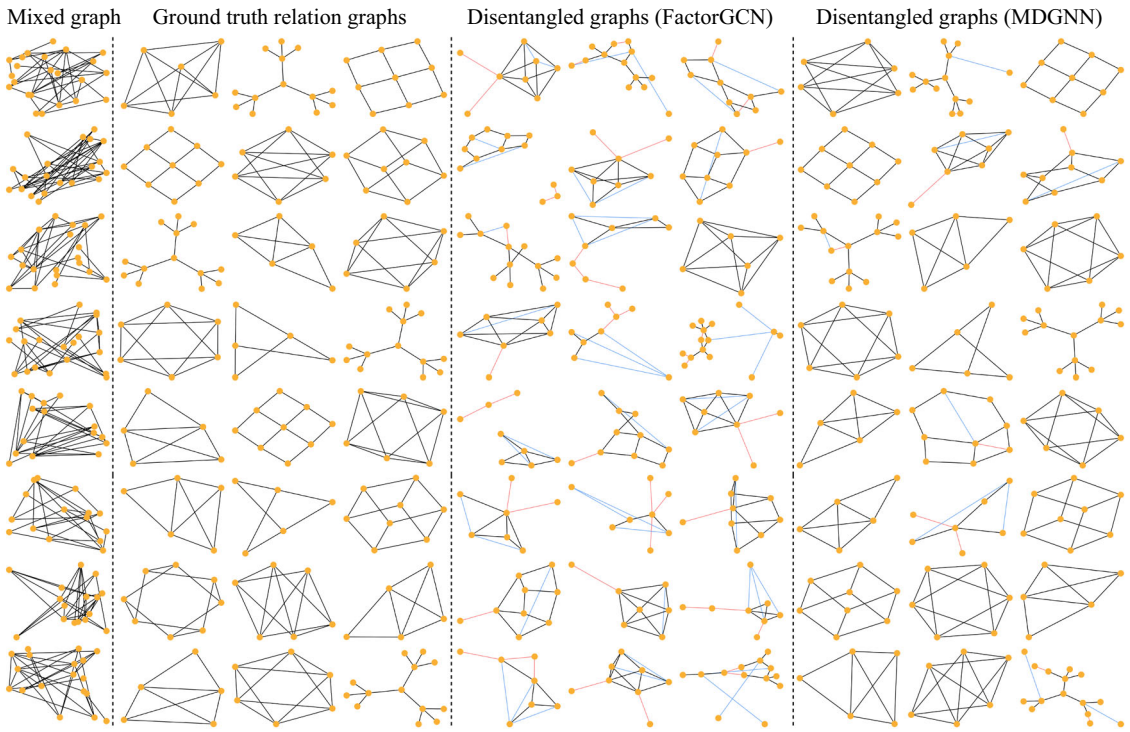


Fig. 3 Examples of the ground-truth and disentangled relation graphs on the synthetic dataset

disentangled GNNs, including DisenGCN, FactorGCN, and ADGCN on the synthetic. Besides, MLP, GCN, and GAT, as three representative methods, are also included for comparison. For the ZINC dataset, we add MoNet [28] and GatedGCN [6] as baselines. On the Cora, Citeseer, and PubMed datasets, we compare MD-GNN with GraphSAGE, AdaLNet, APPNP, GPRGNN, CensNet, and KrylovNet to demonstrate that MD-GNN serves well as a general GNN framework even putting side its excellent disentanglement capability.

4.1.3 Hyperparameters

The following hyperparameters are set for the synthetic dataset: Adam optimizer with learning rate $lr = 0.005$ and weight decay $decay$ = 5e-4; Epoch $E = 200$; Layer number $L = 2$ with middle dimension $d_m = 18$; Relation number $K = 6$, subspace dimension $F = 18$; Iteration steps $S = 3$, teleport probability $\alpha = 0.2$; Loss weights $\beta = \lambda = 1.0$. The experimental settings of the other four datasets are the same as above, but with several dataset-specific hyperparameters determined by a toolkit - NNI, including relation number K, iteration steps S, teleport probability α, and loss weights β, λ.

4.1.4 Evaluation protocol

For the downstream tasks, we adopt micro-F1 for the *multi-label graph classification* on the synthetic dataset, MAE for the *graph regression* on the ZINC dataset, and classification accuracy for the *multi-class node classification* on the Cora, Citeseer, and PubMed datasets. Furthermore, we use two metrics—GED_E and C-Score—proposed by [40] to evaluate the disentanglement performance. The first one is Graph Edit Distance on Edge (GED_E), which restricts the traditional GED by only allowing adding and removing the edges, and thus obtains a score by Hungarian match between the generated relation graphs and the ground truths. Besides GED_E, we also care about the consistency of generated relation graphs. In other words, the best-matched pairs between the generated factor graphs and the ground truths, optimally, should be identical across all samples. We, therefore, use the second metric named consistency score (C-Score), which is computed as the average percentage of the most frequently matched relation graphs. In this paper, each set of experiments is run five times with different random seeds, and the mean and standard deviation are reported as the final metric.

4.2 Qualitative evaluation

We first provide the qualitative evaluation results on the performance of the edge-level, attribute-level, and node-

level disentanglement, corresponding to the visualization of the disentangled relation graphs, relation-related attributes, and correlation analysis of the learned disentangled features in the first layer.

4.2.1 Disentangled relation graphs

Some disentangled relation graphs are provided in Fig. 3 to give an intuitive understanding of the edge-level disentanglement. We visualize the best-matched relation graphs with ground truths. In particular, the *redundant* and *missing* edges in the relation graphs are marked in red and blue, respectively. It is found that only MD-GNN yields highly consistent disentangled graphs with ground truths, while the disentangled graphs of FactorGNN show lots of misidentified edges.

4.2.2 Correlation of disentangled features

Figure 4 shows the correlation analysis of 108-dimensional latent features with $K = 6$ relations on the synthetic dataset. We can see that only the correlation map of MD-GNN exhibits six *clear diagonal blocks*, indicating that it can extract highly independent hidden features with excellent node-level disentanglement performance. None of the compared methods, except FactorGCN and ADGCN, can capture the mutual exclusion information. Nevertheless, FactorGCN and ADGCN still lag far behind the proposed MD-GNN in terms of node-level disentanglement performance.

4.2.3 Relation-related attributes

An essential criterion for relation graphs is that they must be interpretable, *i.e.*, locating the attributes associated with each relation graph and enabling attribute-level disentanglement. We show the ground truth and disentangled attributes corresponding to each relation graph in Fig. 5. Specifically, for each relation graph, we show only the highest scoring attributes to make the number of corresponding attributes equal to the true ones. It is clear that MD-GNN is able to locate the attributes associated with each relation and accomplish the task of attribute-level disentanglement, whereas no other compared methods possess this capability.

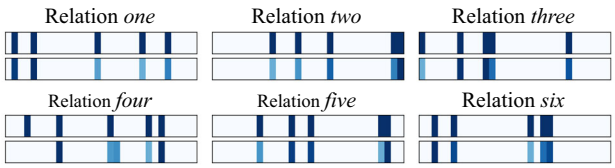


Fig. 5 Ground-truth (upper row) and disentangled (lower row) attributes corresponding to each relation (a total of six relations) in the input on the synthetic dataset

4.3 Quantitative evaluation

4.3.1 Evaluation on the synthetic dataset

The graph classification and disentanglement performance on the synthetic dataset is reported in Table 4, where we mark the disentanglement metrics of MLP and GCN as "-" since they are not capable of graph disentanglement. In terms of classification performance evaluated by Micro-F1, MD-GNN performs much better than other baselines, which demonstrates that despite the powerful disentanglement capability, it does not prevent MD-GNN from being a general-purpose GNN that still outperforms conventional models such as GCN and GAT on downstream tasks. For example, the performance of MD-GNN is 4.4% and 2.3% higher than GCN and GAT, respectively. Moreover, MD-GNN achieves the best performance with respect to the disentanglement performance evaluated by GED_E and C-Score. Compared with the advanced method ADGCN, MD-GNN has a 5.513 reduction in the GED_E metric and improves the C-Score metric by 0.294.

4.3.2 Evaluation on the ZINC dataset

For the ZINC dataset, the type information of edges is hidden during the training process and served as the ground truth to evaluate the disentanglement performance. As shown in Table 5, MD-GNN achieves the best performance on both the disentanglement and downstream tasks. We also show the performance of GatedGCN on this dataset as a baseline for comparison, which utilizes the type information of edges during the training process. In

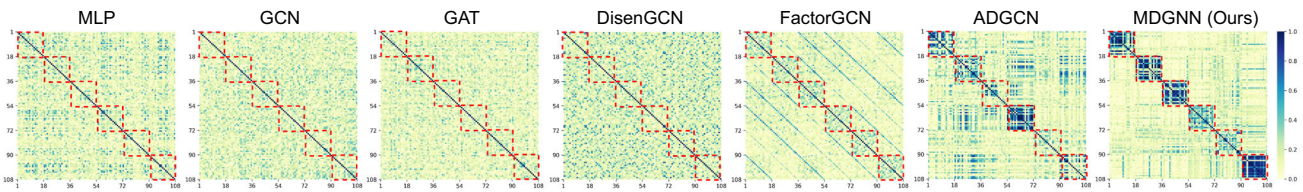


Fig. 4 Feature correlation analysis of 108-dimensional latent features on the synthetic dataset

Table 4 Graph classification and disentanglement performance on the synthetic dataset

	MLP	GCN	GAT	DisenGCN	FactorGCN	ADGCN	MD-GNN (Ours)	Random
Micro-F1 ↑	0.898±0.003	0.916±0.004	0.937±0.006	0.904±0.007	0.945±0.003	0.951±0.006	**0.960±0.004**	0.196±0.002
GED_E ↓	-	-	19.311±3.282	16.283±3.466	16.435±3.572	15.392±3.454	**9.879±3.326**	39.424±5.56
C-Score ↑	-	-	0.288±0.065	0.344±0.031	0.478±0.047	0.428±0.067	**0.722±0.037**	0.286±0.008

terms of graph regression performance evaluated by MAE, MD-GNN performs much better than other baselines, for example, its MAE performance is 0.031 and 0.010 lower than ADGCN and FactorGCN, respectively. Even when compared with the state-of-the-art method GatedGCN, MD-GNN still shows advantages. In terms of disentanglement performance, MD-GNN is the best in the metric C-score, and second only to FactorGNN in the metric GED_E.

4.3.3 Evaluation on three citation datasets

The focus of this paper is to explore disentanglement on heterogeneous graphs rather than to design more powerful GNNs to achieve state-of-the-art performance for all tasks or efficiently deal with large-scale graphs. Therefore, we also evaluate MD-GNN on three widely used node classification datasets with four different data splits to see the performance of MD-GNN as a general GCN framework. Since there is no ground truth relation between nodes, we only report the classification accuracy in Table 6. It can be observed MD-GNN achieves the best overall performance, showing the potential of MD-GNN as a general GNN framework, even putting aside its excellent disentanglement capability. More importantly, although DisenGCN and FactorGCN can be considered as the disentangled versions of GCN, their performance is not always better than that of GCN, and in some settings, even worse than GCN. For example, on the Cora dataset with 3% labels, the classification accuracy of DisenGCN and FactorGNN are 1.8% and 0.7% lower than that of GCN, respectively. Furthermore, while ADGCN shows better performance than the classical GNN model for all datasets and settings, it still lags behind MD-GNN, especially when labeled data is severely limited. For example, the performance of MD-GNN improves ADGCN by 6.8% and 8.4% on the Citeseer dataset with 0.3% labels and the PubMed dataset with 0.03% labels, respectively.

4.4 Ablation study & sensitivity analysis

4.4.1 Ablation study

This evaluates the effectiveness of various components in the proposed MD-GNN framework through five sets of experiments: the model without (A) Relation Learning (w/o RL); (B) Multi-hop Relation Diffusion (w/o Diffusion); (C) Discriminative Loss (w/o $\mathcal{L}_d$); (D) Orthogonal Loss (w/o $\mathcal{L}_o$), and the (E) the full model. Limited by space, only the results on synthetic, Cora (0.5%), Citeseer (0.3%), and PubMed (0.03%) datasets are provided in the main paper. After analyzing the reported results as shown in Fig. 6, we can draw the following conclusions: (1) Relation learning and diffusion contribute to achieving better performance on downstream tasks. Besides, the absence of multi-hop relation diffusion deteriorates the disentanglement performance slightly, which demonstrates the benefit and importance of long-range dependencies between nodes for disentanglement tasks. In addition, the absence of relation learning makes the model lose its ability to disentangle, so the disentanglement-related metrics (GED_E and C-score) are not reported. (2) The discriminative loss $\mathcal{L}_d$ and the orthogonal loss $\mathcal{L}_o$ help to achieve better edge-level and attribute-level disentanglement, and more importantly, the introduction of these two losses does not deteriorate (and even improve) the performance of downstream tasks thus not affecting the potential of MD-GNN as a general GNN framework.

4.4.2 Sensitivity analysis

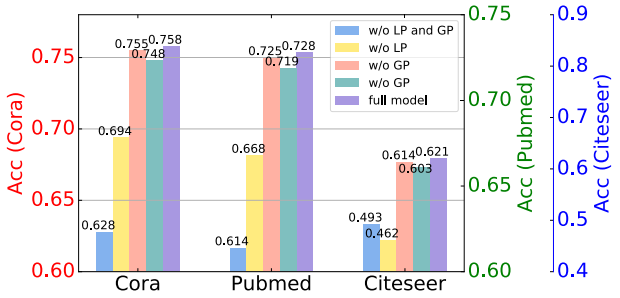
Figure 7 shows the sensitivity analysis with respect to the key hyperparameters like assumed relation number K and multi-hop iteration step S. When varying K, the iteration step S is set to a fixed value; when varying S, the assumed relation number K is fixed. In the figure, the best performance is circled. The performance gain of MD-GNN becomes larger as the number of assumed relation number K and iteration step S increase. However, when the number of K and S become too large, the disentanglement and downstream tasks become more challenging, which in turn

Table 5 Graph regression and disentanglement performance on the ZINC dataset

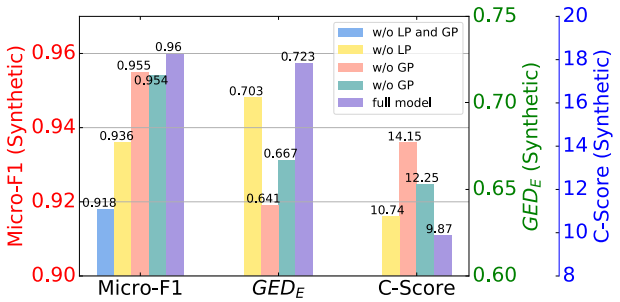
	MLP	GCN	GAT	DisenGCN	FactorGCN	ADGCN	MoNet	GatedGCN	MD-GNN (Ours)
MAE ↓	0.667±0.002	0.503±0.005	0.479±0.010	0.538±0.005	0.366±0.014	0.387±0.015	0.407±0.007	0.363±0.009	**0.356±0.010**
GED_E ↓	-	-	15.46±6.06	14.14±6.19	**12.72±5.34**	14.41±6.23	-	-	14.01±5.97
C-Score ↑	-	-	0.309±0.013	0.342±0.034	0.441±0.012	0.384±0.017	-	-	**0.510±0.023**

Table 6 Node classification accuracy (%) on the Cora, Citeseer, and PubMed datasets with four different dataset splits

Dataset	TrainPCT	GCN	GraphSAGE	GAT	AdaLNet	APPNP	GPRGNN	CensNet	KrylovNet	DisenGCN	FactorGCN	ADGCN	MD-GNN (Ours)
Cora	0.5%	52.9±7.4	37.5±5.4	41.4±6.9	60.8±9.0	58.7±2.6	59.2±4.1	57.7±3.9	74.8	57.8±5.1	59.2±3.4	60.5±4.6	**75.8±4.1**
	1%	61.0±7.2	49.0±5.8	48.6±8.0	67.5±8.7	66.7±1.5	70.5±2.8	67.1±1.3	78.0	64.8±4.3	66.1±2.1	68.7±3.1	**78.6±2.5**
	3%	74.0±2.8	64.2±4.0	56.8±7.9	77.7±2.4	74.1±1.2	77.8±1.7	79.4±1.0	82.7	72.2±2.0	73.3±1.7	76.2±2.3	**83.9±2.2**
	5.2% (*pub*)	80.5±0.8	74.5±0.8	82.6±0.7	80.4±1.1	83.3±0.8	83.9±0.6	81.2±0.9	83.2	83.3±0.9	82.4±1.1	**84.3±0.9**	**84.3±0.7**
Citeseer	0.3%	39.2±6.3	25.7±6.1	30.9±6.9	46.7±5.6	48.0±3.9	52.0±4.1	49.4±3.6	56.7	47.9±3.6	49.2±4.2	53.4±4.1	**60.2±3.7**
	0.5%	47.7±4.4	33.8±7.0	38.2±7.1	53.8±4.7	54.1±4.5	58.2±2.7	57.6±3.0	64.0	52.2±2.6	54.6±2.7	59.1±3.0	**64.2±1.5**
	1%	58.3±4.0	51.0±5.7	46.5±9.3	63.3±1.8	63.1±1.6	64.2±2.4	62.5±2.5	**68.3**	62.1±1.9	61.3±1.6	65.4±1.9	**68.3±2.0**
	3.6% (*pub*)	68.1±1.3	67.2±1.0	72.2±0.9	68.7±1.0	72.2±0.9	72.6±0.5	68.1±0.8	73.9	73.3±1.2	71.8±0.9	**74.2±1.1**	72.5±0.8
PubMed	0.03%	57.9±8.1	45.4±5.5	50.9±8.8	61.0±8.7	56.8±3.1	62.9±3.7	61.4±2.8	72.2	56.4±7.5	57.7±6.6	64.4±6.6	**72.8±3.6**
	0.05%	64.6±7.5	53.0±8.0	50.4±9.7	66.0±4.5	64.4±1.7	68.5±3.2	65.7±1.2	74.9	63.5±4.6	64.2±3.2	69.6±4.2	**75.9±2.7**
	0.1%	73.0±5.5	65.4±6.2	59.6±9.5	72.8±4.6	72.1±1.8	74.7±2.5	69.9±2.1	78.0	73.2±2.8	72.6±2.7	75.7±2.3	**79.0±2.0**
	0.3% (*pub*)	77.8±0.3	76.8±0.6	76.7±0.5	78.1±0.4	78.9±0.6	79.9±0.7	78.5±0.6	80.1	80.5±0.7	79.6±1.0	80.4±0.6	**81.2±0.3**



(a) Accuracy (%) on three real-world datasets



(b) Three evaluation metrics on the synthetic dataset

Fig. 6 Ablation study. **a** shows the accuracy on the Cora, Citeseer, and PubMed datasets. **b** shows the three evaluation metrics on the synthetic dataset

yields lower performance gains. The performance gain depends heavily on the properties of the graph data. Besides, since the synthetic dataset is a dataset specifically for evaluating disentanglement, its performance deteriorates severely when K deviates from the ground truth. In practice, empirical results show that taking $3 \leq K \leq 8$ generally yields better results. However, since how to precisely select K is a problem about estimating the intrinsic property (complexity) of graphs, we use the validation set to determine the empirically optimal K.

5 Discussion

In the proposed MD-GNN framework, the three closely related tasks of edge-level, attribute-level, and node-level disentanglement are tackled in a unified end-to-end framework. Next, we explain how this is achieved from the following three aspects: (1) edge-level disentanglement is achieved through relation learning defined in Eq. 4, which produces multiple relation graphs, each corresponding to one latent relation between nodes; (2) we define a learnable mask for each relation as in Eq. 3 and then achieve attribute-level disentanglement through the orthogonality constraint defined in Eq. 5; (3) node-level disentanglement is achieved by propagating and aggregating the input node attributes in each relation graph to obtain disentangled representations.

In terms of disentanglement performance, extensive qualitative and quantitative experiments have demonstrated that MD-GNN outperforms existing methods, including the state-of-the-art methods FactorGCN and ADGCN. Even putting aside its excellent disentanglement performance, MD-GNN still achieves much better performance than classical GNN models, such as GCN and GAT, which shows the potential of MD-GNN as a general GNN framework. Note that we have not tested the graph classification performance on any public dataset for the following three reasons: (1) The focus of this paper is to explore disentanglement on heterogeneous graphs rather than to design more powerful GNNs to achieve state-of-the-art performance for all graph-related tasks. (2) The reason we evaluated the micro-F1 metric on the synthetic dataset is to show that MD-GNN is endowed with strong disentanglement capabilities without compromising its performance as a general-purpose GNN for downstream tasks. (3) All related works on graph disentanglement, including DisenGNN, FactorGNN, and ADGCN, all mainly focus on the node classification task, and this paper just follows their experimental settings for a fair comparison.

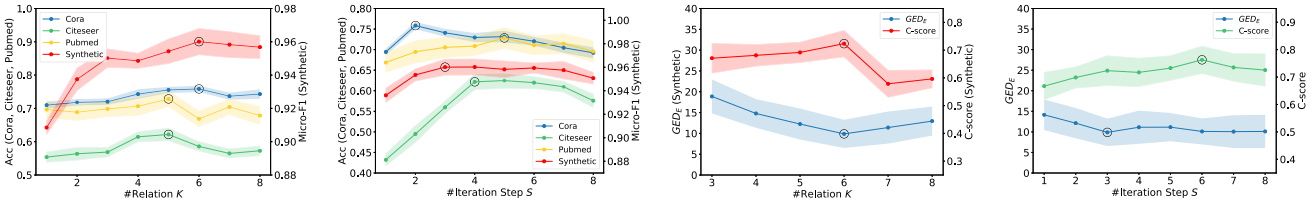


Fig. 7 Sensitivity analysis of key hyperparameters like relation number K and iteration step S. **a**, **b** is the performance of downstream task (Accuracy and Micro-F1) with different K and S on the four datasets. **c**, **d** shows the disentanglement performance (GED_E and C-Score) with different K and S on the synthetic dataset

Despite the great progress, some challenging problems on graph disentanglement are still left for future work: (1) If the relation number K is unknown, how to estimate it from the given graph data? (2) How to implement user-defined graph disentanglement, i.e., given the specific semantics, learning the corresponding representations? (3) How to combine disentanglement with graph explainability for analyzing the learned model.

6 Conclusion

The proposed MD-GNN framework implements all three levels of disentanglement simultaneously in a unified framework in an end-to-end manner. The MD-GNN model learns several interpretable relations from an input binary structure, each representing a latent and disentangled relation between entities. More importantly, we also locate the input attributes associated with each relation. The learned relations are then diffused in each relation space through a multi-hop diffusion mechanism to capture long-range dependencies and produce disentangled features through information aggregation and merging. Extensive experiments on synthetic and real-world datasets have shown that the MD-GNN outperforms other leading methods on both disentanglement and downstream tasks, indicating the proposed MD-GNN framework can *serve as a general GCN framework with the capability of multi-level graph disentanglement.*

Acknowledgments This work is supported in part by the Science and Technology Innovation 2030- Major Project (No. 2021ZD0150100) and National Natural Science Foundation of China (No. U21A20427).

References

1. Brock A, Donahue J, Simonyan K (2018) Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096
2. Chen X, Duan Y, Houthooft R, Schulman J, Sutskever I, Abbeel P (2016) Infogan: interpretable representation learning by information maximizing generative adversarial nets. Adv Neural Inf Process Syst 29:2172–2180
3. Chen Z, Villar S, Chen L, Bruna J (2019) On the equivalence between graph isomorphism testing and function approximation with gnns. arXiv preprint arXiv:1905.12560
4. Chien E, Peng J, Li P, Milenkovic O (2020) Adaptive universal generalized pagerank graph neural network. arXiv preprint arXiv:2006.07988
5. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. arXiv preprint arXiv:1606.09375
6. Dwivedi VP, Joshi CK, Laurent T, Bengio Y, Bresson X (2020) Benchmarking graph neural networks. arXiv preprint arXiv:2003.00982
7. Feng Z, Wang X, Ke C, Zeng AX, Tao D, Song M (2018) Dual swap disentangling. Adv Neural Inf Process Syst, pp. 5894–5904
8. Giles CL, Bollacker KD, Lawrence S (1998) Citeseer: An automatic citation indexing system. In: proceedings of the third ACM conference on Digital libraries, pp. 89–98
9. Hamilton W, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. Adv Neural Inf Process Syst, pp. 1024–1034
10. Higgins I, Matthey L, Pal A, Burgess C, Glorot X, Botvinick M, Mohamed S, Lerchner A (2016) beta-vae: learning basic visual concepts with a constrained variational framework
11. Jiang X, Zhu R, Li S, Ji P (2020) Co-embedding of nodes and edges with graph neural networks. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2020.3029762
12. Jin W, Barzilay R, Jaakkola T (2018) Junction tree variational autoencoder for molecular graph generation. arXiv preprint arXiv:1802.04364
13. Kingma DP, Welling M (2013) Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114
14. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907
15. Kipf TN, Welling M (2016) Variational graph auto-encoders. arXiv preprint arXiv:1611.07308
16. Klicpera J, Bojchevski A, Günnemann S (2018) Predict then propagate: graph neural networks meet personalized pagerank. arXiv preprint arXiv:1810.05997
17. Li P, Wang Y, Wang H, Leskovec J (2020) Distance encoding: design provably more powerful neural networks for graph representation learning. arXiv preprint arXiv:2009.00142
18. Liao R, Zhao Z, Urtasun R, Zemel RS (2019) Lanczosnet: Multi-scale deep graph convolutional networks. arXiv preprint arXiv:1901.01484
19. Lin L, Wang H (2020) Graph attention networks over edge content-based channels. In: proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 1819–1827
20. Liu F, Xue S, Wu J, Zhou C, Hu W, Paris C, Nepal S, Yang J, Yu PS (2020) Deep learning for community detection: progress, challenges and opportunities. arXiv preprint arXiv:2005.08225
21. Liu Y, Wang X, Wu S, Xiao Z (2020) Independence promoted graph disentangled networks. In: AAAI, pp. 4916–4923
22. Locatello F, Bauer S, Lucic M, Raetsch G, Gelly S, Schölkopf B, Bachem O (2019) Challenging common assumptions in the unsupervised learning of disentangled representations. In: international conference on machine learning, pp. 4114–4124. PMLR
23. Luan S, Zhao M, Chang XW, Precup D (2019) Break the ceiling: stronger multi-scale deep graph convolutional networks. Adv Neural Inf Process Syst 32:10945–10955
24. Ma J, Cui P, Kuang K, Wang X, Zhu W (2019) Disentangled graph convolutional networks. In: international conference on machine learning, pp. 4212–4221
25. Ma X, Wu J, Xue S, Yang J, Zhou C, Sheng QZ, Xiong H, Akoglu L (2021) A comprehensive survey on graph anomaly detection with deep learning. IEEE Trans Knowl Data Eng
26. Maron H, Ben-Hamu H, Serviansky H, Lipman Y (2019) Provably powerful graph networks. arXiv preprint arXiv:1905.11136
27. McCallum AK, Nigam K, Rennie J, Seymore K (2000) Automating the construction of internet portals with machine learning. Inf Retr 3(2):127–163
28. Monti F, Boscaini D, Masci J, Rodola E, Svoboda J, Bronstein MM (2017) Geometric deep learning on graphs and manifolds using mixture model cnns. In: proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5115–5124
29. Schmidhuber J (1992) Learning factorial codes by predictability minimization. Neural Comput 4(6):863–879
30. Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T (2008) Collective classification in network data. AI Mag 29(3):93–93

31. Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P (2013) The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process Mag 30(3):83–98
32. Su X, Xue S, Liu F, Wu J, Yang J, Zhou C, Hu W, Paris C, Nepal S, Jin Dz et al. (2021) A comprehensive survey on community detection with deep learning. arXiv preprint arXiv:2105.12584
33. Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv preprint arXiv:1710.10903
34. Wang G, Ying R, Huang J, Leskovec J (2020) Direct multi-hop attention based graph neural network. arXiv preprint arXiv:2009.14332
35. Wang G, Ying Z, Huang J, Leskovec J (2020) Multi-hop attention graph neural network
36. Wang X, Türetken E, Fleuret F, Fua P (2014) Tracking interacting objects optimally using integer programming. In: European conference on computer Vision, pp. 17–32. Springer
37. Wang X, Türetken E, Fleuret F, Fua P (2015) Tracking interacting objects using intertwined flows. IEEE Trans Pattern Anal Mach Intell 38(11):2312–2326
38. Xu B, Shen H, Cao Q, Cen K, Cheng X (2020) Graph convolutional networks using heat kernel for semi-supervised learning. arXiv preprint arXiv:2007.16002
39. Xu B, Shen H, Cao Q, Qiu Y, Cheng X (2019) Graph wavelet neural network. arXiv preprint arXiv:1904.07785
40. Yang Y, Feng Z, Song M, Wang X (2020) Factorizable graph convolutional networks. Adv Neural Inf Process Syst 33
41. You J, Gomes-Selman J, Ying R, Leskovec J (2021) Identity-aware graph neural networks. arXiv preprint arXiv:2101.10320
42. You J, Ying R, Leskovec J (2019) Position-aware graph neural networks. In: International conference on machine learning, pp. 7134–7143. PMLR
43. Zheng S, Zhu Z, Liu Z, Ji S, Zhao Y (2021) Adversarial graph disentanglement. arXiv preprint arXiv:2103.07295
44. Zhu Y, Xu W, Zhang J, Liu Q, Wu S, Wang L (2021) Deep graph structure learning for robust representations: a survey. arXiv preprint arXiv:2103.03036