World Journal of Engineering Research
ISSN: 2959-9873
DOI: https://doi.org/10.61784/wjer3030

GRAPH AUTOENCODERS: A SURVEY

LiNing Yuan
School of Information Technology, Guangxi Police College, Nanning 530028, Guangxi, China.
Corresponding Email: yuanlining@gcjcxy.edu.cn

Abstract: Graph analysis serves as a robust approach for the in-depth exploration of the inherent characteristics of graph data. Nonetheless, due to the non-Euclidean nature of such data, conventional data analysis techniques often incur significant computational expenses and spatial overhead. Graph autoencoders present a viable solution to the challenges associated with graph analysis by converting the original graph data into a low-dimensional representation while maintaining essential information. This transformation subsequently improves the efficacy of various downstream tasks, including node classification, link prediction, and node clustering. This paper offers a thorough review of the existing literature on graph autoencoders, encapsulating the fundamental strategies employed by these models and their applications in downstream tasks. Additionally, the paper suggests prospective avenues for future research in the domain of graph autoencoders.
Keywords: Graph autoencoders; Graph representation learning; Graph neural networks; Graph analysis tasks

1 INTRODUCTION

Graphs serve as prevalent information carriers within complex systems, adept at encapsulating a multitude of intricate relationships found in various domains, including social networks [1], criminal networks [2], and transportation networks [3]. As a representation of non-Euclidean data, graph structures present significant challenges when directly applied to deep learning methodologies such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). To facilitate feature representation in graph data mining, graph encoders are employed to map nodes into a low-dimensional space, thereby producing low-dimensional vectors that preserve critical information from the original graph. Presently, these methodologies have not only demonstrated efficacy in machine learning tasks associated with complex networks, such as node classification [4], link prediction [5], node clustering [6], and visualization [7], but have also found extensive application in practical scenarios, including social influence modeling [8] and content recommendation [9].

Initial iterations of graph autoencoders primarily focused on data dimensionality reduction, constructing similarity graphs based on neighborhood relationships and embedding nodes into low-dimensional vector spaces while ensuring the preservation of similarity among connected node vectors. However, these methods often exhibit high time complexity, which poses challenges for scalability in large graphs. In recent years, there has been a notable shift in graph autoencoder algorithms towards more scalable solutions. Although numerous reviews have been conducted to summarize and categorize these methodologies, they predominantly emphasize traditional approaches, thereby neglecting many emerging models.

This paper aims to provide a thorough and systematic review of graph autoencoder methodologies, contributing in the following ways: (1) a systematic analysis of existing models that offers novel insights into the understanding of current techniques; and (2) the identification of potential research directions for the advancement of graph autoencoders.

2 METHODS

The autoencoders [10] are specific type of artificial neural networks that comprises two components: an encoder and a decoder, which are employed to create vector representations of input data in an unsupervised fashion. By capturing the nonlinear relationships inherent in the data, the autoencoder enables the representations derived from the hidden layer to possess a lower dimensionality than the original input data, thereby facilitating dimensionality reduction. Graph embedding techniques that leverage autoencoders utilize these networks to model the nonlinear structures of graphs, resulting in the generation of low-dimensional embedding representations. These techniques have their origins in GraphEncoder, which employs sparse autoencoders. The fundamental concept involves inputting a normalized graph similarity matrix as the original feature set for the nodes into the sparse autoencoder for hierarchical pre-training. This process allows the resulting low-dimensional nonlinear embeddings to approximate the reconstruction of the input matrix while maintaining its sparse characteristics. GraphEncoder [11] effectively compresses the information contained in the input matrix $\boldsymbol{X}$ into a low-dimensional embedding $\boldsymbol{Y}$, which is subsequently optimized using L2 reconstruction loss. The use of sparse autoencoders not only reduces computational complexity but also provides a more flexible and efficient alternative compared to traditional spectral clustering methods.

SDNE [12] employs deep autoencoders in conjunction with first-order and second-order similarities of the graph to effectively model complex nonlinear network structures. The framework incorporates both supervised and unsupervised elements (illustrated in Figure 1) to preserve the first-order and second-order similarities among nodes. The supervised component utilizes Laplacian feature mapping as the objective function for first-order similarity, facilitating the generation of embeddings that encapsulate local structural characteristics. Conversely, the unsupervised component

adapts the L2 reconstruction loss function as the objective for second-order similarity, which allows the embeddings to capture global structural attributes. The joint optimization of both first-order and second-order similarities significantly enhances the model's resilience in the context of sparse graphs, ensuring that the resulting embeddings effectively retain both global and local structural information.

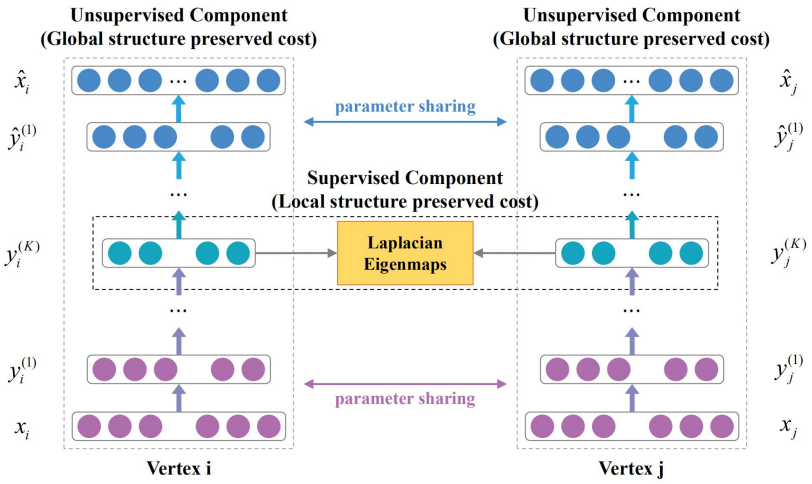

Figure 1 The Framework of SDNE.

The process of generating low-dimensional embeddings using DNGR [13] is primarily comprised of three distinct steps: (1) the application of a random walk model to capture the structural characteristics of the graph, resulting in the creation of a co-occurrence probability matrix; (2) the computation of the Positive Pointwise Mutual Information (PPMI) matrix derived from the co-occurrence probability matrix; and (3) the utilization of the PPMI matrix as input for a Stacked Denoising Autoencoder (SDAE) to produce low-dimensional embedding representations. In contrast to random walks, random surfing directly extracts the structural information of the graph, thereby addressing the limitations inherent in the original sampling methodology. The PPMI matrix effectively preserves the high-order similarity information of the graph, while the stacked architecture facilitates a gradual reduction in the dimensionality of the hidden layers, thereby enhancing the capacity of deep learning models to capture complex features. Furthermore, the incorporation of a denoising strategy contributes to the overall robustness of the model.

DNE-APP [14] employs a semi-supervised stacked autoencoder (SAE) to produce low-dimensional embeddings that preserve k-order information, which is achieved through a two-step process: (1) the generation of a similarity aggregation matrix that encapsulates k-order information using the PPMI metric and a k-step transition probability matrix; and (2) the application of the SAE to reconstruct this similarity aggregation matrix, thereby facilitating the learning of low-dimensional nonlinear embedding representations. In contrast to SDNE, which is limited to first-order and second-order similarities, the DNE-APP model is capable of maintaining various k-order similarities. Furthermore, unlike DNGR, which focuses solely on the reconstruction of high-order similarities, DNE-APP incorporates pairwise constraints during the reconstruction phase, thereby ensuring that similar nodes are positioned closer together within the embedding space.

Variational Autoencoders (VAE) [15] serve as generative models that facilitate dimensionality reduction, offering the benefits of noise tolerance and the ability to learn smooth representations. The Variational Graph Autoencoder (VGAE) [16], as illustrated in Figure 2, is the first application of VAE for the purpose of acquiring interpretable undirected graph embedding representations. In this model, the encoder component employs Graph Convolutional Networks (GCN) [17], while the decoder component utilizes the inner product of the embeddings. The optimization of the VGAE model is achieved through the minimization of both the reconstruction loss and the variational lower bound. In contrast, the Linear-VGAE [18], as proposed by Salha et al., substitutes the GCN encoder in VGAE with a straightforward linear model that is based on the normalized adjacency matrix and does not incorporate an activation function, thereby simplifying the encoder's complexity. Comparative performance evaluations indicate that this basic linear node encoding scheme is equally effective as the more complex VGAE model.

VAGE emerged as powerful graph representation learning methods with promising performance on graph analysis tasks. However, existing methods typically rely on GCN to encode the attributes and topology of the original graph. This strategy makes it difficult to fully learn high-order neighborhood information, which weakens the capacity to learn higher-quality representations. Yuan et al. propose the MoVGAE (illustrated in Figure 3) [19] with co-learning of first-order and high-order neighborhoods. GCN and Multi-order Graph Convolutional Networks (MoGCN) are utilized to generate the mean and variance for the variational autoencoders. Then, MoVGAE uses the mean and variance to calculate node representations. Specifically, this approach comprehensively encodes first-order and high-order information in the graph data.

Graph representation learning models rely on specific task to preserve features, and the generalization of node representations are limited. Aiming at the above problems, a model Self-VGAE [20] introducing self-supervised information was proposed in this paper. Firstly, two-layer graph convolutional encoder and node representation inner product decoder were used to construct a variational graph autoencoder, and the features of the original graph were extracted. Then, topology and attributes were used to generate self-supervised information, and constrain the generation of node representations during training.

In contrast to conventional asymmetric models, GALA [21] employs a fully symmetric graph convolutional autoencoder framework to produce low-dimensional embedding representations of graphs. During the reconstruction of the input matrix, the Laplacian smoothing executed by the encoder is symmetrically aligned with the Laplacian sharpening conducted by the decoder. Distinct from existing VGAE methodologies, GALA incorporates a Laplacian sharpening representation characterized by a spectral radius of 1, which facilitates the decoder's direct reconstruction of the nodes' feature matrix. In comparison to models that solely utilize Graph Convolutional Network (GCN) encoders, GALA's symmetric architecture allows for the concurrent utilization of both structural information and node features throughout the encoding and decoding phases.

On the other hand, ANE [22] employs adversarial autoencoders to generate low-dimensional embeddings that effectively capture highly nonlinear structural information. Specifically, ANE leverages first-order and second-order similarities to encapsulate both local and global structures of the graph, thereby ensuring that the generated embeddings retain a high degree of nonlinearity. The training regimen of the adversarial autoencoder adheres to two primary criteria: the first is an autoencoder training criterion predicated on reconstruction error, while the second is an adversarial training criterion aimed at aligning the aggregated posterior distribution of the embedding representation with a specified prior distribution. Through the implementation of adversarial regularization, ANE addresses the manifold rupture issue prevalent in the embedding generation process, thereby augmenting the representational capacity of the low-dimensional embeddings.

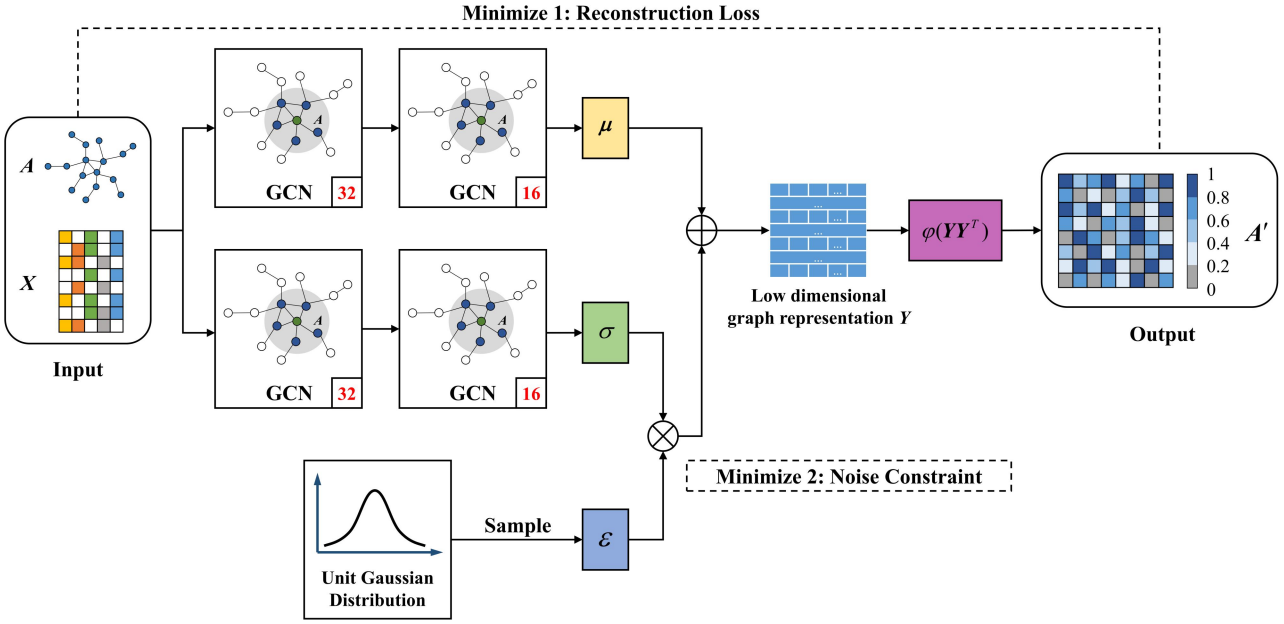

Figure 2 The Framework of VGAE.

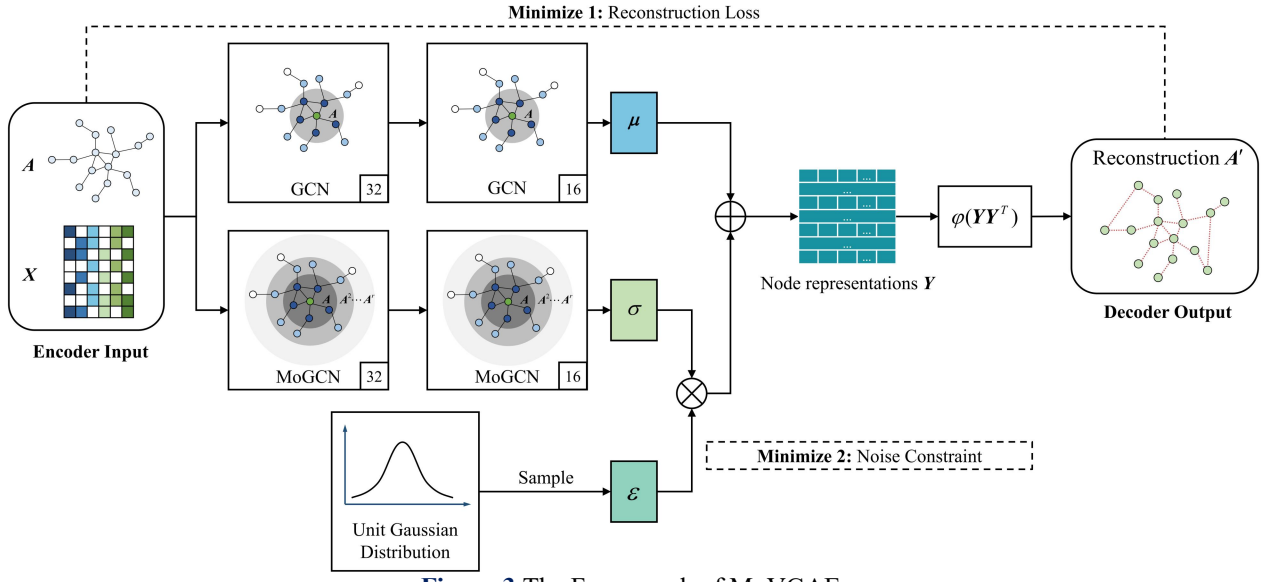

Figure 3 The Framework of MoVGAE.

3 APPLICATIONS

3.1 Network Reconstruction

Network reconstruction entails utilizing learned low-dimensional vector representations of nodes to recreate the topological structure of the original graph, thereby assessing the capacity of the generated embeddings to preserve structural information. This process involves predicting the existence of links between nodes based on the inner product or similarity of their embeddings, and evaluating the model's reconstruction efficacy by calculating the proportion of true links among the top k pairs of vertices in the original graph. The network reconstruction task is generally segmented into three phases: (1) generating embedding representations through a graph autoencoder model; (2) determining the reconstruction proximity of corresponding nodes and ranking them accordingly; and (3) calculating the proportion of true links among the top k pairs of nodes.

3.2 Node Classification

The objective of node classification is to ascertain the category to which each node belongs, utilizing both the topological structure of the graph and the features associated with the nodes. In practical graph datasets, complete labeling may not be achievable; consequently, the labels for a majority of nodes may remain unknown due to various factors. The node classification task can capitalize on the available labeled nodes and their interconnections to infer the missing labels. Furthermore, node classification tasks can be categorized into two types: multi-label classification, where each node is assigned a single category label, and multi-class classification, where nodes may possess multiple category labels.

The node classification task is typically divided into three steps: (1) generating embedding representations using a graph autoencoder model; (2) partitioning the labeled dataset into training and testing subsets; and (3) training a classifier on the training subset and validating the model's performance on the testing subset. Evaluation metrics commonly employed in node classification tasks include micro-F1 and macro-F1. For multi-class tasks, accuracy aligns with the micro-F1 value. The prediction of node labels through network structure and node features has extensive applications in network analysis, allowing for the comparison of the effectiveness of various embedding methods in this context.

3.3 Link Prediction

The link prediction task aims to ascertain whether an edge exists between two nodes or to predict unobserved links within the graph, thereby evaluating the performance of the generated embeddings in maintaining topological structure. This task is typically divided into three steps: (1) generating embedding representations using a graph autoencoder model; (2) labeling the edge information between any two nodes in the dataset and subsequently partitioning the dataset into training and testing subsets; and (3) training a classifier on the training subset and conducting link prediction experiments on the testing subset. Evaluation metrics commonly utilized in link prediction tasks include AUC (Area Under the Curve) and AP (Average Precision). AUC involves setting the threshold just below each positive example, calculating the recall of the negative class, and averaging the results. Conversely, AP sets the threshold just below each positive example, calculates the precision of the positive class, and averages the outcomes. Graph autoencoders can capture the inherent structure of the network, either explicitly or implicitly, to predict potential connections that have not yet been observed.

3.4 Node Clustering

The clustering task employs an unsupervised methodology to partition the graph into multiple communities, wherein nodes within the same community exhibit greater similarity. Following the generation of embeddings using the model, classical techniques such as spectral clustering and k-means are typically applied to cluster the node embeddings. Clustering tasks generally utilize Normalized Mutual Information (NMI) as an evaluation metric, aiming to cluster the generated embedding representations such that nodes with similar characteristics are positioned as closely as possible within the same community.

3.5 Anomaly Detection

The anomaly detection task is designed to identify "abnormal" structures within the graph, which typically encompasses anomaly node detection, anomaly edge detection, and anomaly change detection. Common methodologies for anomaly detection include two primary approaches: one involves compressing the original graph and identifying anomalies within the compressed graph through clustering and outlier detection; the other entails generating node embeddings using the model and grouping them into k communities, thereby detecting new nodes or edges that do not conform to existing communities. Anomaly detection tasks typically employ AUC as an evaluation metric. The primary focus of anomaly detection in graph data is to identify outliers (anomalous points) that significantly deviate from the normal dataset. Effective embedding representations can delineate normal points from anomalous points through the establishment of decision boundaries.

3.6 Visualization

The visualization task encompasses dimensionality reduction and the visualization of embeddings to facilitate an intuitive observation of specific features of the original graph. Visualization tasks are generally conducted on labeled datasets, wherein nodes with differing labels are represented in distinct colors within a two-dimensional space. Given that the embeddings retain certain information from the original graph, the visualization outcomes directly reflect that nodes within the same community in the two-dimensional space exhibit greater similarity. For visualization tasks, robust embedding representations ensure that similar or proximate nodes are positioned closely together in the two-dimensional representation, while dissimilar nodes are effectively separated.

4 FUTURE RESEARCH DIRECTIONS

The examination and evaluation of both traditional and innovative graph autoencoder methodologies indicate that the primary objectives at this juncture involve enhancing the scalability of models to accommodate large-scale and intricate graph data, innovating modeling techniques, and augmenting the efficacy of downstream tasks.

4.1 Autoencoders for Large-Scale Graph Data

In the context of graph embedding tasks, it is imperative to enhance the computational efficiency of models through the utilization of distributed computing or unsupervised learning methodologies. However, existing dynamic graph models frequently fall short in executing graph representation learning tasks when applied to large dynamic graphs characterized by complex evolutionary information. Dynamic graphs are typically represented as a series of snapshots or continuous networks with associated timestamps; consequently, an increase in the number of snapshots or timestamps correlates with heightened complexity in the evolutionary information of the dynamic graph. Thus, two critical aspects in addressing the challenges posed by large-scale graph autoencoders are the reduction of network evolution complexity and the enhancement of embedding model performance.

4.2 Task-Specific Embedding Models

The outputs generated by graph autoencoder models are often employed across a variety of tasks, including node classification, link prediction, and visualization. In contrast to the previously mentioned modeling approaches, task-specific embedding models concentrate exclusively on a singular task, leveraging information pertinent to that task to optimize model training. Generally, task-specific embedding models exhibit superior effectiveness for their designated tasks compared to general embedding models. Consequently, the design of high-performance models tailored for specific tasks represents a significant avenue for future research.

4.3 Application of Large Model Techniques in Graph Autoencoders

Large models (LLMs) have exhibited formidable capabilities in representation learning and generation within domains such as natural language processing, and the methodologies derived from these models offer valuable insights for the advancement of graph autoencoders. Firstly, the exploration of graph-text fusion representation investigates the integration of LLMs to comprehend textual attribute information, amalgamating it with graph structures to create multimodal graph autoencoders that enhance the informational richness and interpretability of node representations. Secondly, research on prompt learning and adaptation centers on the design of graph-related prompts to direct pre-trained graph models or LLMs in adapting to downstream graph tasks, thereby minimizing fine-tuning expenses and bolstering few-shot learning capabilities. Thirdly, the domain of graph generation and inference capitalizes on the robust generative abilities of large models, in conjunction with the structural encoding provided by graph autoencoders, to formulate more controllable and high-quality graph generation models that satisfy complex constraints, including the investigation of intricate graph inference tasks supported by large models. Lastly, parameter-efficient fine-tuning (PEFT) employs techniques such as LoRA and Adapter to large-scale graph models or graph-text fusion models, thereby diminishing the resource requirements for training and deployment.

5 CONCLUSION

This article offers an extensive review of the existing literature on graph autoencoders, delineating pertinent definitions associated with this topic and systematically examining the fundamental strategies and theoretical frameworks of current models. In the section dedicated to applications, it discusses prevalent machine learning tasks, including node classification and link prediction, while evaluating the performance of various models. Ultimately, the article suggests three potential research avenues within the domain of graph autoencoders, focusing on aspects of graph data, modeling strategies, and application contexts.

COMPETING INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

FUNDING

This work was supported by the Project for Enhancing Young and Middle-aged Teacher's Research Basis Ability in Colleges of Guangxi under Grant 2024KY0904.

REFERENCES

[1] Balasubramaniam K, Vidhya S, Jayapandian N, et al. Social network user profiling with multilayer semantic modeling using ego network. International Journal of Information Technology and Web Engineering (IJITWE), 2022, 17(1): 1-14.

[2] Troncoso F, Weber R. A novel approach to detect associations in criminal networks. Decision Support Systems, 2020, 128: 113159.

[3] Guo S N, Lin Y F, Feng N, et al. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. Proceedings of the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, Jan 27-Feb 1, 2019. Menlo Park: AAAI, 2019: 922-929.
[4] Bhagat S, Cormode G, Muthukrishnan S. Node classification in social networks. Aggarwal C C .Social Network Data Analytics. Berlin, Heidelberg: Springer, 2011: 115-148.
[5] Liben-Nowell D, Kleinberg J. The link-prediction problem for social networks. Journal of the American Society for Information Science and Technology, 2007, 58(7): 1019-1031.
[6] Ding C H Q, He X F, Zha H Y, et al. A min-max cut algorithm for graph partitioning and data clustering. Proceedings of the 2001 IEEE International Conference on Data Mining, San Jose, Nov 29-Dec 2, 2001. Washington: IEEE Computer Society, 2001: 107-114.
[7] Vander M L, Hinton G. Visualizing data using t-SNE. Journal of Machine Learning Research, 2008, 9(11): 2579-2605.
[8] Qiu J Z, Tang J, Ma H, et al. DeepInf: social influence prediction with deep learning. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, Aug 19-23, 2018. New York: ACM , 2018: 2110-2119.
[9] Silveira T, Zhang M, Lin X, et al. How good your recommender system is? A survey on evaluations in recommendation. International Journal of Machine Learning and Cybernetics, 2019, 10(5): 813-831.
[10] Bourlard H, Kamp Y. Auto-association by multilayer perceptrons and singular value decomposition. Biological Cybernetics, 1988, 59(4): 291-294.
[11] Tian F, Gao B, Cui Q, et al. Learning deep representations for graph clustering. Proceedings of the 28th AAAI Conference on Artificial Intelligence, Québec City, Jul 27 -31, 2014. Menlo Park: AAAI, 2014: 1293-1299.
[12] Wang D X, Cui P, Zhu W W. Structural deep network embedding. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, Aug 13-17, 2016. New York: ACM, 2016: 1225-1234.
[13] Cao S S, Lu W, Xu Q K. Deep neural networks for learning graph representations. Proceedings of the 30th AAAI Conference on Artificial Intelligence, Phoenix, Feb 12-17, 2016. Menlo Park: AAAI, 2016: 1145-1152.
[14] Shen X, Chung F L. Deep network embedding with aggregated proximity preserving. Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, Sydney, Jul 31 – Aug 3, 2017. New York: ACM, 2017: 40-43.
[15] Kingma D P, Welling M. Auto-encoding variational Bayes. arXiv:1312.6114, 2013.
[16] Kipf T N, Welling M. Variational graph auto-encoders. arXiv:1611.07308, 2016.
[17] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks. arXiv:1609.02907, 2016.
[18] Salha G, Hennequin R, Vazirgiannis M. Keep it simple: graph autoencoders without graph convolutional networks. arXiv:1910.00942, 2019.
[19] Yuan L, Jiang P, Wen Z, et al. Improving Variational Graph Autoencoders With Multi-Order Graph Convolutions. IEEE Access, 2024, 12: 46919-46929. DOI:10.1109/ACCESS.2024.3380012.
[20] Yuan L, Wen Z, Feng W, et al. Graph Representation Learning Enhanced by Self-supervised Information. Guangxi Sciences, 2024, 31(2): 323-334.
[21] Park J, Lee M, Chang H J, et al. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision, Seoul, Oct 27 – Nov 2, 2019. Piscataway: IEEE, 2019: 6518-6527.
[22] Xiao Y, Xiao D, Hu B B, et al. ANE: network embedding via adversarial autoencoders. Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing, Shanghai, Jan 15-17, 2018. Washington: IEEE Computer Society, 2018: 66-73.