IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 April 2023, accepted 15 May 2023, date of publication 22 May 2023, date of current version 8 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3278271

RESEARCH ARTICLE

Random Walk Graph Auto-Encoders With Ensemble Networks in Graph Embedding

CHENGXIN XIE[1], XIUMEI WEN[1,2], FANXING MENG[1], AND HUI PANG[1]
[1]College of Information Engineering, Hebei University of Architecture, Zhangjiakou, Hebei 075000, China
[2]Big Data Technology Innovation Center of Zhangjiakou, Zhangjiakou, Hebei 075000, China

Corresponding authors: Fanxing Meng (sir363@163.com) and Hui Pang(23283162@qq.com)

This work was supported in part by the Basic Scientific Research Business Expenses of Provincial Colleges and Universities in Hebei Province under Project XY2023020.

ABSTRACT Recently graph auto-encoders have received increasingly widespread attention as one of the important models in the field of deep learning. Existing graph auto-encoder models only use graph convolutional neural networks (GCNs) as encoders to learn the embedding representation of nodes. However, GCNs are only suitable for transductive learning, have poor scalability and shallow models with a poor perceptual field, and have limitations in node feature extraction. To alleviate these problems, we propose to use an adaptive weight integration graph attention network (GAT) and GCN's random walk graph auto-encoder (EGRWR-GAE) to better learn the embedding representation of nodes. There is a large amount of noise in the graph data, which interferes with feature extraction and the GAT model is sensitive to noisy data, we propose a random walk graph auto-encoder (EGSRWR-GAE) that integrates GAT, GCN, and self-supervised graph attention networks (SuperGAT) using adaptive weights. The effectiveness of our model is well demonstrated by three publicly available datasets (Cora, Citeseer, and Pubmed) with optimizations of up to 2.2% on the link prediction task and up to 12.9% on the node clustering task.

INDEX TERMS Graph auto-encoder, adaptive weight, ensemble, EGRWR-GAE, EGSRWR-GAE.

I. INTRODUCTION

Usability has been demonstrated in low-dimensional vector representations of nodes in graphs for a wide range of machine-learning tasks [1], [6]. These tasks mainly include node classification, recommender systems, social networks, and link prediction. A large amount of graph data exists in real life, especially in structures such as social networks, traffic forecasting, and proteins [1], [26], [27]. Graph embedding, link prediction, and node clustering are important tasks in graph mining [26]. Graph auto-encoders are an important object of research in graph neural networks, and the earliest graph auto-encoder model was proposed by Kipf and Welling [15]. The model is better able to obtain embedding representations of nodes, learn low-dimensional vector representations of nodes, and better apply them to tasks such as graph embedding, link prediction, and node clustering [1], [26], [33]. As a result, a great deal of research has been carried out by researchers to better learn node feature representations.

The associate editor coordinating the review of this manuscript and approving it for publication was Shadi Alawneh.

Graph embedding is a process of converting high-dimensional dense graph data into low-dimensional vectors through some mapping [5], [18], [19]. To obtain better graph embeddings, researchers have continuously proposed new models. Pan et al. [25] proposed a graph auto-encoder based on adversarial regularization, which solved the problem of lower embeddings due to the traditional graph auto-encoder model ignoring the underlying data distribution in the graph, but the model is not an optimal solution to alleviate the low embedding problem. For the optimization scheme proposed by Pan et al. [25], Huang and Frederking [26] argued that the method is not optimal and therefore proposed a graph auto-encoder based on the random walk regularization technique. The random walk graph auto-encoder can alleviate the problem of ignoring node features in the reconstruction matrix and can better alleviate the low embedding problem [26], but the model can only alleviate the low embedding problem and does not solve it. Although several graph auto-encoder models have been proposed in recent works, none of them consider graph structure and node features. To address this problem, Salehi et al. [6] proposed

a graph attention encoder that can be used on induction tasks but the model is only applicable to shallow structures. Dou et al. [27] merged the ideas from [25] and [26] and proposed a graph auto-encoder based on adversarial and random walk strategies to solve the low embedding problem. The existing graph auto-encoder models all use GCNs as encoders [1], [7], [8], [10], [11]. GCNs are only applicable to transductive and cannot be applied to inductive learning [9]. At the same time, GCN as an encoder only uses a shallow structure [18], which makes the convergence of the model poor and will lose some node features causing some limitations in node feature extraction [2], [10], [11], [33], [38]. To solve these problems, we proposed the EGRWR-GAE model. However, there is also a large amount of noisy data in the graph data [23], and to solve the interference caused by noisy data for feature extraction, we proposed the EGSRWR-GAE model. The specific contributions of this paper are as follows:

- We use adaptive weights to integrate GAT and GCN as new encoders, which can solve the limitations and poor scalability problems of traditional graph auto-encoder feature extraction.
- We use adaptive weights to integrate GAT, GCN, and SuperGAT as new encoders that can resolve the interference caused by noisy data to better obtain the embedding representation of the nodes.
- We change the disadvantage of the traditional graph auto-encoder model that only applies to transductive, in this paper the model can be applied to both transductive and inductive learning, enhancing the generalization ability of the model. At the same time, our model can also be used for directed graphs, solving the drawback that the traditional model is only applicable to undirected graphs.

II. RELATED WORK

The essence of graph embedding is a compression technique that compresses higher-dimensional vectors into lower-dimensional vectors. Perozzi et al. [12] proposed a DeepWalk model based on a depth-first search, which computes similarity ranking to obtain nearest-neighbor recommendations. Tang et al. [13] proposed a LINE model based on breadth-first search, which is a model that uses neighborhood similarity to compute first-order and second-order similarity to obtain the embedding representation of nodes. Grover and Leskovec [14] propose a Node2Vec model with structure and homogeneity, which is essentially an extension of DeepWalk. Ribeiro et al. [16] propose the Struc2Vec model, which captures structural similarity through the structure of the graph.

With the continuous research in deep learning, graph embedding research nowadays is more likely to use graph auto-encoder models, which are simple in the overall architecture, easy to use, and efficient [26]. Tran [17] proposed the MTGAE model, which can obtain the latent representation of a node using the graph structure. Most graph auto-encoder models only consider undirected graphs, ignoring the more complex directed graphs [8], [19]. Salha et al. [19] applied graph auto-encoder to the task of link prediction of directed graphs, solving the situation where most models only consider undirected graphs. However, real-life graph data often exists with hundreds of thousands of nodes, and the structure of the graph is very complex. For the processing of large graph data, traditional models are difficult to handle and poorly scalable [1]. To obtain better node embedding methods, Salha et al. [1] proposed the FastGAE model, which uses random subgraphs to scale millions of nodes and complex graph structures and can rapidly accelerate the convergence speed and performance of the model. Although several models have been proposed recently, many of them have shortcomings in reconstructing the original graph, model robustness, and error metrics. For this reason, Hou et al. [34] proposed a self-supervised training GraphMAE model to alleviate these problems associated with the graph training process. To make better use of node relationships and graph structure, Chen et al. [36] proposed the LGCN-FF model, which learns feature representations of nodes from heterogeneous graphs and then uses DSA functions to achieve graph fusion. GCN uses graph structure to obtain node information, but the extant models are shallow models that have limitations in solving multi-view tasks. Wu et al. [37] proposed the IMvGCN model, which uses reconstruction error and Laplace matrix learning tasks to better establish the connection between GCN and multi-view learning from a feature and structural perspective, and to better learn node features. Existing graph auto-encoder models all use only shallow structures [18], and there are limitations of non-Euclidean data on the link prediction task. The DGAE model proposed by Wu et al. [18] for this problem can effectively solve the problem. The graph auto-encoder of the shallow model has poor perceptual field and convergence and will lose some node features causing some limitations in node feature extraction [11]. Instead, we use the integration idea to integrate different networks to learn node features and use adaptive weights to combine them. Experiments have shown that the encoder integrating multiple networks is much better than the encoder of a single network.

III. METHODOLOGY

This section focuses on the relevant technologies used.

A. GAT

The GCN model assigns identical weights to neighbors of the same order of neighborhood [3], [20], which limits the model's ability to capture spatial information relevance, while how GCN combines features of neighboring nodes is closely related to the graph structure, limiting the model's ability to generalize [21]. Velikov et al. [21] proposed a GAT model that uses an attention mechanism to weigh the summation of neighboring node features. Essentially, GAT simply replaces the original normalization function of GCN with an aggregation function of neighboring node features using attention

weights. both GAT and GCN are essentially feature extractors that predict new node features for N nodes according to their input node features [21].

GAT is an optimized version of GCN, which addresses the shortcomings of GCN in terms of its inability to be applied to inductive learning and directed graphs [3]. the essence of GAT is that each vertex j computes attention to all vertices on the graph, which allows the model to be well suited to inductive learning without being limited by the structure of the graph. Secondly, the GAT model requires weights to be calculated with neighboring nodes, so it can be applied even in directed graphs.

B. SUPERGAT

GAT can better extract node features using the attention mechanism, but the presence of a large amount of noisy data in the graph data [23], [24] can cause the GAT model to be ineffective. To solve this problem, Kim et al. [22] proposed the SuperGAT model. SuperGAT is essentially an optimization of the GAT model, which is not always effective in the face of noisy data. The SuperGAT model improves this problem by directing attention through the presence or absence of edges between pairs of nodes. The model can reduce the interference of noisy data and is essentially a variant form of GAT. SuperGAT proposes four kinds of attention: GO, DP, SD, and MX [22]. The Go approach is the original GAT model, and DP refers to the dot product approach where two vectors do the inner product. the SD is scaling on top of DP and MX is a combination of Go and DP using Sigmoid scales DP between 0 and 1 and then multiplies it with the Go result.

C. RANDOM WALK GRAPH AUTO-ENCODERS

The Graph Auto Encoder (GAE) enables the task of link prediction. Similar to the Auto Encoder, the GAE can reconstruct the adjacency relationships between nodes. The graph auto-encoder model was first proposed by Kipf and Welling [15] and the model is divided into two main parts encoder and decoder. The graph G = (A, X) where A denotes the adjacency matrix and X denotes the feature matrix of the node [26], [27]. The GCN is used as the encoder [1], [7], [8], [10], [11] and the embedding representation Z of the nodes is obtained through a non-linear activation function, Z is defined as shown in equation (1):

$$Z = GCN(X, A)[15] \tag{1}$$

For the decoder, the original graph can be reconstructed by making an inner product of the obtained node embedding matrices Z and ZT. The decoder is defined as shown in equation (2):

$$A' = \sigma\left(ZZ^T\right)[15] \tag{2}$$

where A' denotes the reconstructed adjacency matrix and σ denotes the activation function.

A walk strategy was proposed by the random walk graph auto-encoder to solve the low embedding problem. A random walk is used on the graph to obtain a sequence of nodes [4]. The sequence of nodes that are walked to is fed into a SkipGram model [26]. The SkipGram model takes the feature vectors of these nodes as input and then outputs the representation vectors of the nodes after a hidden layer, using down-sampling for optimization. In this way, contextual nodes can be predicted, generating a low-dimensional representation for each node. The exact structure can be seen in the lower half of the model in Figure 1.

D. OUR MODEL

Unlike traditional graph auto-encoder models that only use GCN as an encoder, we propose random walk graph auto-encoders using an integration approach: EGRWR-GAE and EGSRWR-GAE. The specific structure of the models can be seen in Figure 1. The EGRWR-GAE model integrates GAT and GCN as encoders using adaptive weights. The EGSRWR-GAE model integrates GAT and GCN as encoders using the EGSWR-GAE model integrates GAT, GCN, and SuperGAT as encoders using adaptive weights. Both models take the adjacency matrix A and the node feature matrix X as input, obtain the node features in the hidden layer using the GAT, GCN, and SuperGAT network layers respectively, then integrate the node features learned by each of the three networks through adaptive weights to form the new node features, and finally pass through the fully connected layer to obtain the node embedding matrix Z. Where the adaptive weights W are hyperparameters. The optimal value is obtained through continuous training, and finally, the original graph is reconstructed using the decoder. Our proposed EGRWR-GAE and EGSRWR-GAE models are essentially reconstructing new node features to obtain a better representation of node features. Our model, therefore, uses three different networks to learn node features to obtain better node features. Firstly, we take the neighborhood matrix A and the node feature matrix X, which are composed of the graph, as input to each of the three different networks, and then pass them through the non-linear activation layer to obtain the hidden layer representation of the nodes. We integrate the node features learned by the three networks with adaptive weights to generate new node features, and by reconstructing them we can obtain an embedding representation of the node for downstream tasks.

It is found that our proposed models EGRWR-GAE and EGSRWR-GAE can solve the limitations of traditional graph auto-encoder feature extraction and effectively address the shortcomings of using a single network as an encoder. The specific model structure is shown in Figure 1.

IV. EXPERIMENT

Our work revolves around two unsupervised tasks: link prediction and node clustering. Table 1 shows the details of the three datasets we used.

A. LINK PREDICTION

1) BASELINE

In this paper, we compare several algorithmic models commonly used for link prediction.

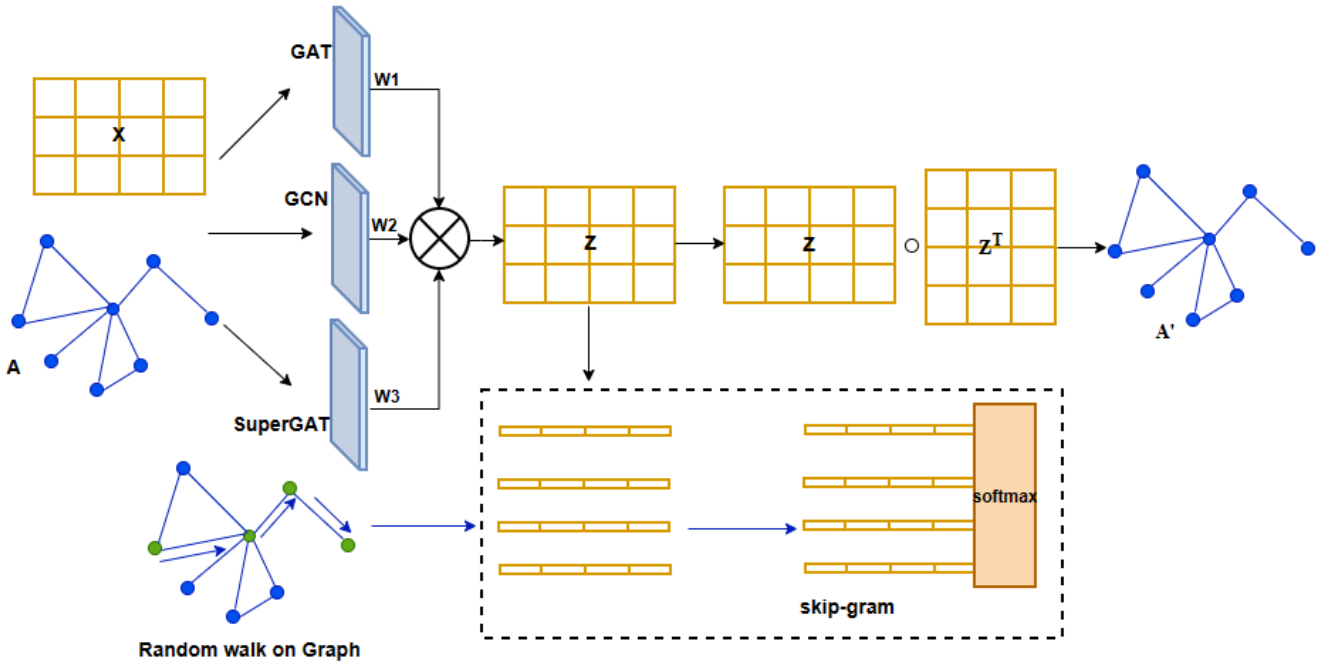

FIGURE 1. The overall architecture of the EGSRWR-GAE model. The EGRWR-GAE model integrates GCN and GAT networks as new encoders via adaptive weights.

TABLE 1. Datasets used in this paper.

Datasets	Nodes	Edges	Content Words	Features	Labels
Cora	2708	5429	3880584	1433	7
Citeseer	3327	4732	12274336	3703	6
Pubmed	19717	44338	9858500	500	3

Deepwalk: Deepwalk is a ranking that computes similarity to get nearest neighbor recommendations, a sequence of nodes generated by a random walk on the graph, and then an embedding of these nodes generated by the Word2Vec method.

Spectral Clustering: Spectral Clustering is an efficient embedding algorithm commonly used for social networks.

GAE: The topology of the graph and the node features are used as input to obtain an embedding representation of the nodes. Finally, the original graph is reconstructed using the inner product.

ARGA [25]: ARGA is based on a graph auto-encoder using an adversarial regularization technique to better learn the embedding representation of nodes.

RWR-GAE [26]: RWR-GAE is to input the sequence of nodes obtained by restarting random walk from the graph to the SkipGram model, thus predicting contextual nodes and obtaining a better embedding representation.

NPGNN [35]: The model is not only applicable to inductive learning tasks but also generalizes well in the training of subgraphs and can transfer subgraph information to the full graph.

EGRWR-GAE: Our proposed model uses adaptive weights to integrate GAT and GCN as new encoders for better learning of node embeddings.

EGSRWR-GAE: Our proposed model uses adaptive weights to integrate GAT, GCN, and SuperGAT as new encoders to learn the embedding representation of nodes.

2) METRICS

To make our work more comparable, we use the AUC and AP metrics [1], [25], [26]. AUC is the area under the ROC curve, and its higher value indicates the probability of predicting positive cases ahead of negative cases, AP is the average accuracy of the PR curve, and its higher value indicates the accuracy of the model in predicting positive cases [25], [27]. In this paper, the data set is divided into a training set, a validation set, and a test set, where the validation set contains 5% of the data and the remaining 10% of the test set is used as the training set [25], [26], [27].

3) EXPERIMENT SETTINGS

On the Cora and Citeseer datasets the total number of training epochs = 200, the initial learning rate is lr = 0.01, the hidden layer units are 32 and 16, the step size is l = 30, the window size is w = 30, and the number of random walks t = 50, We are in agreement with these parameters from Kipf and Welling [1]. where the dropout rate dropout = 0.5 and the number of heads head = 3 for GAT on the Cora dataset, dropout = 0.6 for SuperGAT using attention as MX, the sampling rate of 0.8 for edges and the number of heads head = 12. On the Citeseer dataset, the dropout rate is

dropout = 0.6 and the number of heads head for GAT = 3, SuperGAT has a dropout rate of 0.4, uses SD as attention, has a sampling rate of 0.8 edges, and has a head count of 12. On the Pubmed dataset, the total training epoch = 2000, the initial learning rate lr = 0.01, the hidden layer cells are 128, the step size l = 70, the window size w = 80, and the number of random walks t = 30. t = 30. GAT dropout rate dropout = 0.6, number of heads head = 3, SuperGAT dropout rate dropout = 0.6, using attention as MX, the sampling rate of edges 0.8, number of heads head = 12.

TABLE 2. Parameter settings for different datasets.

Parameters	Cora	Citeseer	Pubmed
window size w	30	30	80
walk length l	30	30	70
walk per node r	50	50	30
number of neurons	32-16	32-16	128-64
learning rate	0.01	0.01	0.01
number of iterations	200	200	2000

4) EXPERIMENT RESULTS

Our results in the linked predictions are detailed in Table 3. We compared the AUC and AP values of the different algorithmic models on the Cora, Citeseer, and Pubmed datasets and experimentally demonstrated that our models EGRWR-GAE and EGSRWR-GAE performed better: all of them had AUC and AP values above 93.5%. Compared to the NPGNN model, our proposed model EGRWR-GAE improves the AUC and AP in the Cora dataset by 1.1% and 1% respectively. the EGSRWR-GAE model improves the AUC and AP in the Cora dataset by 0.6% and 0.9% respectively. Compare to the RWR-GAE model, the EGRWR-GAE model improves the AUC and AP in the Citeseer dataset by 1.5% and 3% respectively. The AUC and AP of the EGRWR-GAE model in the Citeseer dataset improved by 1.4% and 2.9%, respectively. Compared to the NPGNN model, the AUC and AP of the EGRWR-GAE model in the Pubmed dataset improved by 1.8% and 2.2%, respectively. The AUC and AP of the EGSRWR-GAE model in the Pubmed dataset were improved by 1.7% and 2.1%, respectively.

The comparison with different models shows that our model achieves good results in connection prediction. The main reason for this is that we use an integrated approach to learn node features using different networks, which is widely used in machine learning. The advantages of integration are also very obvious. We use separate networks to learn the features of the nodes. The integration method is then used to combine the different networks using adaptive weights to form new node features. Through several experiments, our model achieves good results on the link prediction task.

B. NODE CLUSTERING

1) BASELINES

For the node clustering task, we use the traditional k-means algorithm for the clustering task, in addition to the models for link prediction comparison and other comparison models:

TABLE 3. Algorithm comparison results.

Model	Cora		Citeseer		Pubmed	
	AUC	AP	AUC	AP	AUC	AP
SC[25]	84.6	88.5	80.5	85.0	84.2	87.8
DeepWalk[25]	83.1	85.0	80.5	83.6	84.4	84.1
GAE[25]	91.0	92.0	89.5	89.9	96.4	96.5
ARGA[27]	92.3	93.1	91.7	93.0	96.6	97.0
RWR-GAE(our)	92.6	92.2	92.1	91.5	96.2	96.2
NPGNN[35]	93.1	93.9	94.0	95.0	95.2	95.2
EGRWR-GAE	94.2	94.9	93.6	94.5	97.0	97.4
EGSRWR-GAE	93.7	94.8	93.5	94.4	96.9	97.3

k-means [25]: k-means is the most classical representative of unsupervised learning clustering algorithms

GraphEncoder [28]: GraphEncoder is an unsupervised algorithm that uses the obtained node embedding representations for clustering tasks.

DNGR [29]: DNGR is a graph embedding model that uses the graph structure to generate a vector representation of the nodes.

RTM [30]: RTM is a relational model of network structure and node hierarchy

RMSC [31]: RMSC is a multi-view clustering algorithm

TADW [32]: TADW proves that DeepWalk is an equivalent of the matrix decomposition algorithm

ARWR-GE [27]: The essence of this algorithm is a fusion of the ARGA [25] model and the RWR-GAE [26] model to better solve the embedding problem.

EGAE [39]: The model uses a relaxed k-means inner product distance space to obtain the optimal partitioning, allowing for a good interpretative representation of the nodes, which can also be applied to other tasks.

DGAE [18]: The model uses deep structure, using skip connections to incorporate node feature representations and graph structure to better extract potential information from the graph.

2) METRICS

For the clustering task, our model uses the same evaluation metrics as the other models, which can make the comparison of the models more illustrative. The metrics used are the accuracy Acc response rate of the model to accurately identify true positives and false negatives [25], [33]. Normalized Mutual Information NMI, is used to measure how similar the results of two clusters are [33]. Adjusted Rand index ARI, which responds to the degree of overlap between the two

divisions [33]. F1, also known as the balanced F-score, is defined as the summed average of the precision and recall rates [33]. precision indicates the proportion of correctly predicted positive classes out of all predicted positive classes [33].

3) EXPERIMENT RESULTS

The results of our experiments on the node clustering task are detailed in Table 4, Table 5, and Table 6. We compared Acc, NMI, ARI, F1, and Precision of different algorithmic models on the Cora, Citeseer, and Pubmed datasets. on the Cora dataset, our model EGRWR-GAE compared to ARWR-GE showed an 8% improvement in ACC, a 6% improvement in NMI, a 10.6% improvement in F1, a 12.7% improvement in ARI and a 10.9% improvement in Precision. NMI improved by 6%, F1 by 10.6%, Precision by 12.7%, and ARI by 10.9%. The model EGSRWR-GAE improved ACC by 8.7%, NMI by 6.7%, F1 by 11.3%, Precision by 12.1%, and ARI by 12.9% compared to ARWR-GE. On the Citeseer dataset our model EGRWR-GAE has a 5% improvement in ACC, 3.3% improvement in NMI, 3% improvement in F1, 6.4% improvement in Precision, and 5.4% improvement in ARI compared to RWR-GAE. The model EGSWR-GAE showed a 4.1% improvement in ACC, 4.8% improvement in F1, 5.4% improvement in Precision, and 1.7% improvement in ARI compared to ARWR-GE. On the Pubmed dataset our model EGRWR-GAE has a 7% improvement in ACC, 7.3% improvement in NMI, 3% improvement in F1, 18% improvement in Precision, and 8.3% improvement in ARI compared to GAE. The model EGSRWR-GAE showed a 6.8% improvement in ACC, a 9.1% improvement in F1, an 18.2% improvement in Precision, and an 8% improvement in ARI compared to GAE.

TABLE 4. Comparison of clustering results of different algorithmic models on the Cora dataset.

Model	Acc	NMI	F1	Precision	ARI
DeepWalk[25]	0.484	0.327	0.392	0.361	0.243
k-means[25]	0.492	0.321	0.368	0.369	0.230
Spectral[25]	0.367	0.127	0.318	0.193	0.031
GraphEncoder[25]	0.325	0.109	0.298	0.182	0.006
DNGR[25]	0.419	0.318	0.340	0.266	0.142
RTM[25]	0.440	0.230	0.307	0.332	0.169
RMSC[25]	0.407	0.255	0.331	0.227	0.090
TADW[25]	0.560	0.441	0.481	0.396	0.332
GAE[27]	0.594	0.426	0.591	0.593	0.343
ARGA[27]	0.640	0.447	0.617	0.644	0.353
RWR-GAE(our)	0.655	0.469	0.618	0.629	0.417
ARWR-GE[27]	0.667	0.482	0.620	0.631	0.417
EGAE[39]	0.693	0.511	-	-	0.448
EGRWR-GAE	0.747	0.542	0.726	0.758	0.526
EGSRWR-GAE	0.754	0.549	0.733	0.752	0.546

TABLE 5. Comparison of clustering results of different algorithmic models on the Citeseer dataset.

Model	Acc	NMI	F1	Precision	ARI
DeepWalk[25]	0.337	0.088	0.270	0.248	0.092
k-means[25]	0.540	0.305	0.409	0.405	0.279
Spectral[25]	0.239	0.056	0.299	0.179	0.010
GraphEncoder[25]	0.225	0.033	0.301	0.179	0.010
DNGR[25]	0.326	0.180	0.300	0.200	0.044
RTM[25]	0.451	0.239	0.342	0.349	0.203
RMSC[25]	0.295	0.139	0.320	0.204	0.049
TADW[25]	0.455	0.291	0.414	0.312	0.228
GAE[25]	0.408	0.176	0.372	0.418	0.124
ARGA[27]	0.572	0.348	0.544	0.572	0.340
RWR-GAE(our)	0.559	0.312	0.527	0.545	0.290
EGAE[39]	0.583	0.334	-	-	0.308
EGRWR-GAE	0.609	0.345	0.557	0.609	0.344
EGSRWR-GAE	0.600	0.308	0.575	0.599	0.307

TABLE 6. Comparison of the clustering results of different algorithmic models on the Pubmed dataset.

Model	Acc	NMI	F1	ARI
DeepWalk[40]	0.543	0.102	0.530	0.088
k-means[40]	0.580	0.278	0.544	0.246
Spectral[40]	0.496	0.147	0.299	0.010
GraphEncoder[40]	0.531	0.210	0.506	0.184
DNGR[40]	0.468	0.153	0.445	0.059
M-NMF[40]	0.470	0.084	0.443	0.058
RMSC[40]	0.629	0.273	0.521	0.247
TADW[40]	0.565	0.224	0.481	0.228
GAE[40]	0.632	0.249	0.511	0.246
DGAE[18]	0.684	0.290	-	0.291
EGRWR-GAE	0.702	0.322	0.691	0.329
EGSRWR-GAE	0.700	0.340	0.693	0.326

We compare the models mentioned in the baseline on each of the three datasets and our model achieves better results on the clustering task compared to more models. The experiments show that our model outperforms the other models in all five metrics. The clustering results are found to be significantly better than other models, further demonstrating the usefulness of our model for graph embedding. Through the experimental data above we can demonstrate that our proposed model has a better clustering effect.

V. GRAPH VISUALIZATION

We demonstrate the effectiveness of our proposed model on the Cora dataset using TSNE in two dimensions for the clustering task, which can be seen in Figure 2. We compare the clustering effect of the original RWR-GAE model with that of our proposed model for the clustering task, and we can see that our proposed model is superior.

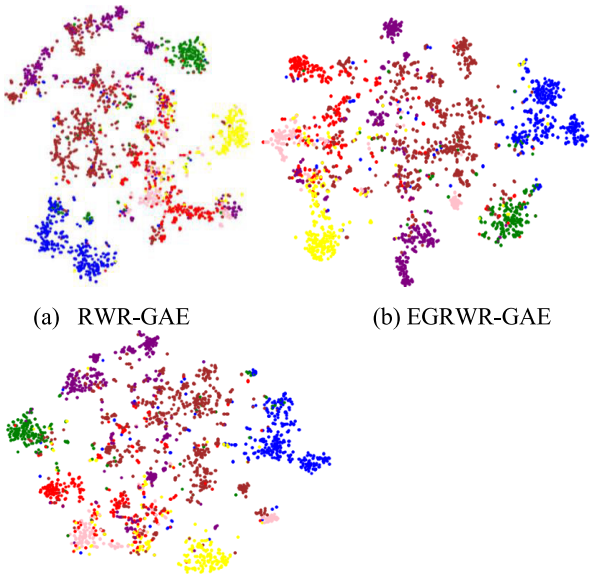

FIGURE 2. Visualization of the results of the three models for the clustering task on the Cora dataset.

VI. DISCUSSION

Graph auto-encoders have been widely used as an important model for deep learning in several fields [7], [17]. Our work differs from models such as ARGA [25], RWR-GAE [26], ARWR-GE [27], NPGNN [35], EGAE [39], and DGAE [38] in that we take an integrated approach. We change the model structure of existing graph auto-encoders and we propose the EGRWR-GAE and EGSRWR-GAE models. The existing optimization models for graph auto-encoders have all achieved very good results, but we find that integrating different network models using the idea of integration can be a good solution to the limitations of the single model node features proposed. However, after we compared more than a dozen network models we found that the network model here works best using GAT and SuperGAT and slightly less well using other network models. On the link prediction task, our model achieves a 2.2% improvement in AUC on the Pubmed dataset compared to the state-of-the-art model NPGNN. On the node clustering task, our model optimizes the ARI metric by 9.8% on the Cora dataset and 3.6% on the Citeseer dataset compared to the state-of-the-art EGAE model. In the PubMed dataset, the ARI metrics were improved by 3.8% compared to the state-of-the-art DGAE model. On the one hand, although our model achieves relatively good results, we find that the model runs more slowly with large datasets than with small ones, since many random walks on large graphs result in a lot of data. To address this problem, we propose to split the large graph into several sub-graphs and perform random walks on each sub-graph to speed up convergence. On the other hand, our model may not work as well as the number of layers increases, due to the shortcomings of the GAT model itself. To address this problem we suggest using regularization techniques such as adding noise to alleviate this problem.

VII. CONCLUSION

All existing graph auto-encoder models only use GCNs as encoders to learn the embedding representation of nodes. However, GCN is only applicable to transductive learning with poor scalability and there are limitations in node feature extraction. Our proposed models EGRWR-GAE and EGSRWR-GAE can address the low embedding problem of existing graph auto-encoder models, and we demonstrate the superiority of the EGRWR-GAE and EGSRWR-GAE models through link prediction and node clustering tasks on the Cora, Citeseer, and Pubmed datasets. In the future, we intend to combine our models with recommendation algorithms to better implement recommendation tasks. In addition to the integration approach incorporated, more techniques deserve to be investigated in depth in the future.

ACKNOWLEDGMENT

The authors would like to thank his tutor Xiumei Wen and Hui Pang for guiding him to complete this article.

REFERENCES

[1] G. Salha, R. Hennequin, J.-B. Remy, M. Moussallam, and M. Vazirgiannis, "FastGAE: Scalable graph autoencoders with stochastic subgraph decoding," *Neural Netw.*, vol. 142, pp. 1–19, Oct. 2021.
[2] X. Li, H. Zhang, and R. Zhang, "Adaptive graph auto-encoder for general data clustering," 2020, *arXiv:2002.08648*.
[3] H. Pei, B. Wei, K. C.-C. Chang, Y. U. Lei, and B. Yang, "Geom-GCN: Geometric graph convolutional networks," in *Proc. ICLR*, 2020, pp. 1149–1150.
[4] Z. Huang, A. Silva, and A. Singh, "A broader picture of random-walk based graph embedding," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, San Francisco, CA, USA, Aug. 2021, pp. 655–664.
[5] Y. Zhang, Q. Yao, and L. Chen, "Interstellar: Searching recurrent architecture for knowledge graph embedding," 2019, *arXiv:1911.07132*.
[6] A. Salehi and H. Davulcu, "Graph attention auto-encoders," 2019, *arXiv:1905.10715*.
[7] M. Tang, C. Yang, and P. Li, "Graph auto-encoder via neighborhood Wasserstein reconstruction," 2022, *arXiv:2202.09025*.
[8] G. Kollias, V. Kalantzis, T. Idé, A. Lozano, and N. Abe, "Directed graph auto-encoders," 2022, *arXiv:2202.12449*.
[9] H. Pei, B. Wei, K. C.-C. Chang, Y. U. Lei, and B. Yang, "Geom-GCN: Geometric graph convolutional networks," 2020, *arXiv:2002.05287*.
[10] M. Ma, S. Na, and H. Wang, "AEGCN: An autoencoder-constrained graph convolutional network," *Neurocomputing*, vol. 432, pp. 21–31, Apr. 2021, doi: 10.1016/j.neucom.2020.12.061.
[11] Z. Weng, W. Zhang, and W. Dou, "Adversarial attention-based variational graph autoencoder," *IEEE Access*, vol. 8, pp. 152637–152645, 2020.
[12] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. SIGKDD*, 2014, pp. 701–710.
[13] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. WWW*, May 2015, pp. 1067–1077.
[14] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. SIGKDD*, Aug. 2016, pp. 855–864.
[15] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[16] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 385–394, doi: 10.1145/3097983.3098061.
[17] P. V. Tran, "Multi-task graph autoencoders," 2018, *arXiv:1811.02798*.
[18] X. Wu and Q. Cheng, "Deepened graph auto-encoders help stabilize and enhance link prediction," 2021, *arXiv:2103.11414*.
[19] G. Salha, R. Hennequin, and M. Vazirgiannis, "Simple and effective graph autoencoders with one-hop linear models," in *Proc. ECML PKDD*, 2020, pp. 319–334.

[20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
[21] P. Veličković, G. Cucurul, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2018, *arXiv:1710.10903*.
[22] D. Kim and A. Oh, "How to find your friendly neighborhood: Graph attention design with self-supervision," 2022, *arXiv:2204.04879*.
[23] W. Feng, J. Zhang, Y. Dong, Y. Han, H. Luan, Q. Xu, Q. Yang, E. Kharlamov, and J. Tang, "Graph random neural networks for semi-supervised learning on graphs," in *Proc. NeurIPS*, 2020, pp. 22092–22103.
[24] E. Turner, "Graph auto-encoders for financial clustering," 2021, *arXiv:2111.13519*.
[25] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2019, pp. 2609–2615.
[26] M. Vaibhav, P.-Y. Huang, and R. Frederking, "RWR-GAE: Random walk regularization for graph auto encoders," 2019, *arXiv:1908.04003*.
[27] W. Dou, W. Zhang, Z. Weng, and Z. Xia, "Graph embedding framework based on adversarial and random walk regularization," *IEEE Access*, vol. 9, pp. 1454–1464, 2021.
[28] F. Tian, B. Gao, Q. Cui, and T.-Y. Liu, "Learning deep representations for graph clustering," in *Proc. 28th AAAI Conf. Artif. Intell.*, Honolulu, HI, USA, 2014, pp. 1293–1299.
[29] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proc. 31st Conf. Artif. Intell.*, San Francisco, CA, USA, 2016, pp. 213–219.
[30] J. Chang and D. Blei, "Relational topic models for document networks," in *Proc. 12th Int. Conf. Artif. Intell. Statist.*, 2009, pp. 81–88.
[31] R. Xia, Y. Pan, L. Du, and J. Yin, "Robust multi-view spectral clustering via low-rank and sparse decomposition," in *Proc. AAAI*, Jun. 2014, vol. 28, no. 1, pp. 2149–2155.
[32] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *Proc. IJCAI*, 2015, pp. 2111–2117.
[33] L. Lin, L. Yongquan, and L. Guangming, "Graph autoencoder based on restart random walk," *Comput. Appl. Res.*, vol. 38, no. 10, pp. 3009–3013, 2021, doi: 10.19734/j.issn.1001-3695.2021.03.0083.
[34] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang, "GraphMAE: Self-supervised masked graph autoencoders," 2022, *arXiv:2205.10803*.
[35] H. Liang and J. Gao, "How neural processes improve graph link prediction," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Singapore, May 2022, pp. 3543–3547, doi: 10.1109/ICASSP43922.2022.9746010.
[36] Z. Chen, L. Fu, J. Yao, W. Guo, C. Plant, and S. Wang, "Learnable graph convolutional network and feature fusion for multi-view learning," *Inf. Fusion*, vol. 95, pp. 109–119, Jul. 2023, doi: 10.1016/j.inffus.2023.02.013.
[37] Z. Wu, X. Lin, Z. Lin, Z. Chen, Y. Bai, and S. Wang, "Interpretable graph convolutional network for multi-view semi-supervised learning," *IEEE Trans. Multimedia*, early access, Mar. 23, 2023, doi: 10.1109/TMM.2023.3260649.
[38] L. Zhong, J. Yang, Z. Chen, and S. Wang, "Contrastive graph convolutional networks with generative adjacency matrix," *IEEE Trans. Signal Process.*, vol. 71, pp. 772–785, 2023, doi: 10.1109/TSP.2023.3254888.
[39] H. Zhang, P. Li, R. Zhang, and X. Li, "Embedding graph auto-encoder for graph clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 25, 2022, doi: 10.1109/TNNLS.2022.3158654.
[40] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, "Attributed graph clustering: A deep attentional embedding approach," 2019, *arXiv:1906.06532*.

CHENGXIN XIE was born in Fuyang, Anhui, China. He received the bachelor's degree in network engineering from the Anhui Xinhua College, in 2021, and the master's degree from the Hebei University of Architecture, in 2021, where he is currently pursuing the degree in computer science and technology. His research interests include graph neural networks, data mining, and machine learning. He has three software publications and has published four related journals and conference papers.

XIUMEI WEN was born in Zhangjiakou, China, in 1972. She received the bachelor's degree in computer application technology from the Northwest Institute of Textile Industry, Xi'an, China, in 1995, and the master's degree in computer application technology from the Hebei University of Technology, Tianjin, in 2003. Since 2012, she has been a Professor with the Hebei University of Architecture, Zhangjiakou. Since 2018, she has also been the Director of the Zhangjiakou Big Data Technology Innovation Centre. She has authored/coauthored 12 books, and holds 20 software copyrights. Her research interests include big data technology and information processing. She has published more than 30 papers in refereed journals and conferences in the above fields. Her awards and honors include Top Ten Teachers from the Hebei University of Architecture, the Ten Outstanding Young Teachers in Zhangjiakou, the Third Class Merit Award in Zhangjiakou, and the Second Class Merit Award from the Hebei University of Architecture.

FANXING MENG was born in Zhangjiakou, China, in 1972. He received the B.S. degree in computer science and technology from the Hebei University of Architecture, Zhangjiakou, in 2003, and the M.S. degree from the Shanghai University of Finance and Economics, Shanghai, China, in 2009.

He was the Director of the Language Laboratory and the Campus Card Management Center. Since 2015, he has been a Senior Experimentalist in computer science and technology with the Hebei University of Architecture. He has coauthored 11 books and holds eight software copyrights. His research interests include the technique and application of databases, data processing, and computer network technology. He has published over 20 papers in refereed journals and conferences in the above areas. His awards and honors include the Outstanding Educator from the Hebei University of Architecture and the Government Procurement Evaluation Expert, Hebei, China.

HUI PANG received the bachelor's degree in computer science from Hebei Normal University, in June 2002, and the master's degree in engineering (computer technology) from the Hebei University of Technology, in January 2009.

In 2002, she came to work with the Department of Computer Science and Technology Teaching and Research, Hebei College of Architecture and Engineering, as a Teacher, and a Secretary of the General Branch with the Department of Computer Science. In 2015, she was promoted to an Associate Professor. In 2018, she was appointed as a master's supervisor. She has long been engaged in teaching and scientific research in the disciplines of computer science and technology and the Internet of Things engineering. She has presided over and led two provincial research projects, led two municipal research projects, and participated in seven provincial and departmental research projects. She has also participated in one provincial-level teaching and research project and published more than ten papers as the first author in this discipline, including two EI retrieved papers and two core journals. She has participated in writing nine computer teaching materials, of which one is the chief editor and six are associate editors, two of which are "Eleventh Five-Year Plan" teaching materials. She guided students to participate in the China Student Computer Design Competition and won the second and third prizes in the Hebei Province. Her main research interests include computer operating systems, data mining, and data processing.