Neural Networks 166 (2023) 273–285

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

RegraphGAN: A graph generative adversarial network model for dynamic network anomaly detection

Dezhi Guo, Zhaowei Liu *, Ranran Li

School of Computer and Control Engineering, Yantai University, Shandong, China

ARTICLE INFO

Article history:
Received 3 May 2023
Received in revised form 18 June 2023
Accepted 15 July 2023
Available online 20 July 2023

Keywords:
Anomaly detection
Generative adversarial network
Dynamic networks

ABSTRACT

Due to the wide application of dynamic graph anomaly detection in cybersecurity, social networks, e-commerce, etc., research in this area has received increasing attention. Graph generative adversarial networks can be used in dynamic graph anomaly detection due to their ability to model complex data, but the original graph generative adversarial networks do not have a method to learn reverse mapping and require an expensive process in recovering the potential representation of a given input. Therefore, this paper proposes a novel graph generative adversarial network by adding encoders to map real data to latent space to improve the training efficiency and stability of graph generative adversarial network models, which is named RegraphGAN in this paper. And this paper proposes a dynamic network anomaly edge detection method by combining RegraphGAN with spatiotemporal coding to solve the complex dynamic graph data and the problem of attribute-free node information coding challenges. Meanwhile, anomaly detection experiments are conducted on six real dynamic network datasets, and the results show that the dynamic network anomaly detection method proposed in this paper outperforms other existing methods.

1. Introduction

Dynamic graph technology has grown significantly in recent years due to the continued growth of dynamic graph applications in social network (Wang et al., 2019), human knowledge networks (Ji, Pan, Cambria, Marttinen, & Philip, 2021), business networks (Zheng et al., 2020), and cybersecurity (Gao, Xiaoyong, Hao, Fang, & Yu, 2020). Graph are widespread in the real world, and all kinds of connections within things and between things can be represented by graph. Meanwhile, in the real world, various relationships are constantly changing, such as human knowledge networks, business networks, e-commerce networks, and the Internet of Things, etc. Therefore, the corresponding graphs are also dynamic. Data from real-world networks tends to change over time (Jiao et al., 2021; Peng et al., 2021). Therefore, research on real-world networks needs to focus not only on the static graph domain, but also on the dynamic graph domain.

Consider the example of an electronic shopping network: new buying relationships are created every day, and the buyer–seller relationship evolves over time, always changing dynamically. As people's purchasing power grows, user-goods networks become more complex, making some fake purchase requests harder to be detected, and attackers in user-goods networks in e-commerce scenarios often place fake purchase orders to illegally increase the influence of certain goods. Thus identifying anomalous behavior in real-world networks helps protect the interests of ordinary users and maintain the health of the system. Meanwhile, in the field of dynamic graph technology, the detection of anomalous edges in dynamic graphs is considered to be an important category in the development of dynamic graph technology (Yu et al., 2018; Zheng, Li, Li, Li, & Gao, 2019). However, due to the particularity of dynamic graphs, the structure of dynamic graphs will change over time, which leads to many existing works (Hooi et al., 2016; McConville, Liu, & Miller, 2015; Zhao & Yu, 2013) that only consider a specific one of the content, structure, or time of dynamic graphs. Changes are happening in the dynamic graph at all times, which makes anomaly detection on the dynamic graph more difficult to achieve than on the static graph. Most importantly, generating attribute data representing each node and edge from the general raw dynamic graph data is considered to be a challenging step, with difficulties arising mainly from the requirements of the exploding volume of time-varying attribute data, or from privacy concerns that render the attributes of the data inaccessible. Furthermore, learning discriminative knowledge from dynamic graph with coupled structure and temporal information is equally difficult. Therefore, considering the features of dynamic graph in an integrated manner is a challenge in dynamic graph anomaly detection.

* Corresponding author.
E-mail addresses: guodezhi@s.ytu.edu.cn (D. Guo), lzw@ytu.edu.cn (Z. Liu), 1216848095@s.ytu.edu.cn (R. Li).

https://doi.org/10.1016/j.neunet.2023.07.026
0893-6080/

Although some current work (Cai et al., 2021; Liu, Pan, et al., 2021; Yang, Zhou, Wen, Zhou, & Wu, 2020; Zheng et al., 2019) based on deep learning becomes a powerful solution for dynamic graph learning. However, these models often require the use of large amounts of already labeled data to train supervised models for the purpose of performing dynamic graph anomaly detection. In fact, in practice, exceptions are usually rare instances of data that represent only a small fraction of the normal data. Therefore, although possible, it is also very difficult to collect a large number of marked exception instances. Actually, the majority of applications of dynamic graph anomaly detection do not allow for the acquisition of large-scale labeled data. Since task-specific labels can be extremely scarce for graph datasets, some work (Liu & Song, 2022; You et al., 2020; Zhou, Song, Yu, & Zheng, 2023) have similarly emphasized the importance of unsupervised training. As a result, how to train a model in the absence of a sufficient amount of anomalous data is a significant difficulty for dynamic graph anomaly detection.

Here, a semi-supervised anomaly detection model for dynamic graphs is proposed in this paper. The model takes advantage of the generative model in generative adversarial networks (GAN) (Goodfellow et al., 2020) which has a good reconstruction capability or learning data distribution capability. During training, only normal data is needed to train the model, and a large amount of abnormal data is no longer needed, so the resulting generative model can only generate or reconstruct normal data. In testing, the test sample is fed into the trained generative model. If the output of the generative model is the same as or close to the input after reconstruction, it indicates that the test is normal data, otherwise it is abnormal data. Much of the existing work (Shrivastava et al., 2017; Zheng, Zheng, & Yang, 2017) also employs generator reconstruction for anomaly detection. In contrast, this paper also focuses on the mapping from real data to latent representations. Specifically, in this paper, an efficient dynamic graph node encoding method (Liu, Pan, et al., 2021) is used to extract global spatial, local spatial and temporal information, and a graph generation adversarial network is developed to learn spatial and temporal knowledge of dynamic graphs. To address the problem that currently existing graph generative adversarial networks ignore the mapping from real data to potential representations and are difficult to use them for complex dynamic graph datasets because of the expensive optimization process required for their use. This paper proposes that the training process of graph generation adversarial networks for dynamic graph anomaly detection should not only learn to identify the differences between the samples generated by the generator and the real data, but also add an encoder to complete the mapping from the real data space to the latent random variables. Ensure the recovery of latent representation of real data in case of input and achieve more stable and efficient detection goals. In this study, inspired by bidirectional generative adversarial networks (Donahue, Krähenbühl, & Darrell, 2016), encoders are added to the training of graph generative adversarial networks to learn the mapping from the real data space to the underlying representation. An encoder from data space to feature space and a generation from feature space to data space in the novel graph generative adversarial network proposed in this paper form an reverse structure, and thus it is named RegraphGAN in this paper.

In summary, the main contributions of this paper are:

- The graph generative adversarial network RegraphGAN proposed in this paper considers not only the differences between generated and real data, but also the differences between random variables and data encoding to provide better potential representations for generators as input, stabilize generators to generate samples, and avoid the expensive optimization process of traditional graph generative adversarial networks.
- This paper adopts edge-based substructure sampling and spatio-temporal node encoding method to cooperate with RegraphGAN to create a complete dynamic graph anomaly detection model, and proves the cutting-edge of this work by testing on six real-world datasets.

The rest of this article is organized as follows. Section 2 reviews the related work. In Section 3, the definition of the problem and the specific implementation of what is presented in this paper are outlined. Section 4 presents the results of the experiments. Finally, Section 5 summarizes the results of the study and suggestions for further research.

2. Related work

This section lists some existing works related to the research work of this paper, in which some existing dynamic network anomaly detection methods, generative adversarial network models in graph networks, and some attempts of adversarial generative networks in anomaly identification are shown.

2.1. Anomaly detection in dynamic graph

In recent years, as various dynamic relationships in the real world are expressed as dynamic graph, the application of anomaly detection on dynamic graph has been widely demanded, and many excellent dynamic graph anomaly detection methods have been proposed by scholars in the research community. Some shallow mechanisms to detect anomalous edges, such as Goutlie (Aggarwal, Zhao, & Philip, 2011), detects outliers in network streams using structural connectivity models and generates dynamic graph partitions to maintain connectivity behavioral models. To mine implicit information, HNRL (Wang, Liu, Xu, & Yan, 2022) proposes a heterogeneous network representation learning method, HNRL uses a heterogeneous network representation learning method to map transaction networks to low-dimensional spaces to mine node and edge types, improving the accuracy of embedding results in node classification tasks. By tracking metrics that include information about network topology and changes in edge weights, CAD (Sricharan & Das, 2014) finds node associations. To discern anomalous edge features, CM-Sketch (Ranshous, Harenberg, Sharma, & Samatova, 2016) takes into account both local structural information and history behavior. StreamSpot (Manzoor, Milajerdi, & Akoglu, 2016) is a clustering-based method that models the behavior of network streams using an unique similarity function for attribute comparison of heterogeneous networks and a centroid-based clustering algorithm. To ensure a high mapping distance between abnormal and normal occurrences in sketch space, Spotlight (Eswaran, Faloutsos, Guha, & Mishra, 2018) employs a randomized sketching technique.

In addition, deep learning-based approaches collect anomalous data in dynamic graph through the application of deep learning techniques. To generate node embeddings with Clique embedding targets, NetWalk (Yu et al., 2018) uses a dynamic update pool approach based on a random wandering encoder to simulate network evolution, and then uses a dynamic clustering-based anomaly detection algorithm to rate the degree of anomaly of each edge. The neural network model created by AddGraph (Zheng et al., 2019) goes on to accurately represent the spatial and temporal characteristics of dynamic graph. A GCN (Kipf & Welling, 2016a) is used as a tool for extracting structural elements, while GRU-Attention is created for the purpose of combining short and long term dynamic progress. StrGNN (Cai et al., 2021) extracts subgraphs of h-hop bounding edges and captures spatial and temporal information using stacked GCNs

and GRUs. The learned model is trained from start to finish thanks to the use of negative sampling from a "contextually relevant" noise distribution. HVGRAE (Yang et al., 2020) constructed a hierarchical model where variogram autoencoders and recurrent neural networks were combined by it. For the detection of aberrant edges, the likelihood of edge reconstruction is employed to quantify the abnormal edges. TA-Struc2Vec (Li, Liu, Ma, Yang, & Sun, 2022) proposes a graph learning algorithm TA-Struc2Vec for Internet financial fraud detection, which can learn topological features and transaction amount features in financial transaction network graphs, and represent them as low-dimensional dense vectors, through training Classifier models enable intelligent and efficient classification and prediction. A node code containing spatial and temporal information is constructed in the framework of Taddy (Liu, Pan, et al., 2021), while Taddy models spatial and chronological information using a transformation network.

2.2. Generative adversarial networks in graph

The generative confrontation network has been widely used in the image field. With the explosion of generative adversarial networks, it has also been applied to the graph domain. Through adversarial training in very small and very large games, GraphGAN (Wang, Wang, Wang, et al., 2023) integrates two graph representation learning methodologies, namely generative and discriminative methodologies. In the GraphGAN framework, the discriminator and generator can cooperate to their mutual advantage: the generator improves its generative performance guided by the discriminator's signal, while the discriminator better distinguishes between ground truth and generated samples driven by the generator. In addition, GraphGAN proposes an implementation of graph softmax as a generator that addresses the inherent limitations of traditional softmax. The complexity of real-world networks can be accurately captured by the implicitly generative network data model known as NetGAN (Bojchevski, Shchur, Zügner, & Günnemann, 2018). Without having to manually specify any of them, it can produce graphs that accurately represent crucial topological characteristics of complex networks, such as population structure and degree distribution. The model also exhibits strong generalization properties, and can even be used to generate graphs with continuously varying features using latent space interpolation.GraphSGAN (Ding, Tang, & Zhang, 2018) is a new method for semi-supervised learning of graphs. In GraphSGAN, the generator and classifier networks play a new kind of competitive game. In equilibrium, the generator generates fake samples in low-density regions between subgraphs. To distinguish the fake samples from the real samples, the classifier implicitly considers the density properties of the subgraphs. In order to improve the traditional normalized graph Laplace regularization, an efficient adversarial learning algorithm is proposed. GraphSGAN has the advantage of scalability and can be trained using small batch processing. Graph representation learning and overlapping community discovery can both be addressed by CommunityGAN (Jia, Zhang, Zhang, & Wang, 2019). CommunityGAN's embedding reveals the vertices' membership in communities, in contrast to conventional graph representation learning techniques where vector entry values have no clear meaning of embedding. Additionally, CommunityGAN optimizes such embeddings using a specifically created generative adversarial network. The parent topic-level generator and discriminator can alternately iterate to enhance their respective performances and ultimately output a better community structure thanks to the very slight rivalry between them.

2.3. Anomaly detection and generative adversarial networks

As mentioned above, GAN has long been used in various works of image research, and anomaly detection is no exception. With the general development of GAN applications, this concept is more frequently applied to anomaly detection. AnoGAN (Schlegl, Seeböck, Waldstein, Schmidt-Erfurth, & Langs, 2017), for example, learns the distribution of normal data using a generative adversarial network's generator. During testing, the image uses the learnt generator to determine what a normal image should look like, and then compares it to determine whether it is abnormal. GANomaly (Akcay, Atapour-Abarghouei, & Breckon, 2018) uses an encoder–decoder–encoder sub-network in the generator network to allow the model to map the input image to a low-dimensional vector, reconstruct the generated output image using this low-dimensional vector, then convert the created image using an extra encoder network. Map to the representation that underpins it During training, reducing the distance between these pictures and latent vectors aids in learning the data distribution of normal samples. A higher distance metric generated from this learned data distribution during the testing phase shows the presence of outliers in this distribution. ALAD (Zenati, Romain, Foo, Lecouat, & Chandrasekhar, 2018) addresses the flaw that AnoGAN (Schlegl et al., 2017) still requires parameter updates during the testing phase. This approach presents a BiGAN-based method that can be 100 times faster, as well as stable training attempts. It learns to map the input sample x to the possible representation z at the same time during training. Generator G, encoder E, and discriminator D are all encoders.

Inspired by the many research works mentioned above, this paper considers a novel method of applying graph generative adversarial networks to anomaly detection in dynamic graph. In this paper, the idea of graph generative adversarial network is applied to anomaly detection in dynamic graphs, an encoder is added to the traditional graph generative adversarial network, and data samples are associated with latent variables through the encoder network. This approach avoids the expensive inference procedure required by typical graph generative adversarial networks, and the required latent variables are recovered through a single feedforward loop of the encoder network at test time.

3. Method

In this section, the content of the graph generative adversarial network RegraphGAN proposed in this paper is described specifically, and how to detect anomalies in dynamic graphs using the graph generative adversarial network proposed in this paper. In Section 3.1, dynamic graphs and dynamic graph anomaly detection are defined, and in Sections 3.2–3.5, specific implementations of the method proposed in this paper is provided. The meanings of some characters are shown in Table 1.

3.1. Problem definition

The specification of a dynamic graph with a maximal sequence number of T is $\Gamma = \{G^t\}_{t=1}^{T}$, where the snapshot of the dynamic graph at time t is denoted as $G^t = \{\upsilon^t, \varepsilon^t\}$, the set of nodes at time t is denoted as υ^t, and the set of edges at time t is denoted as ε^t. Assuming $v_i^t, v_j^t \in \upsilon^t$, if there is an edge between v_i^t and v_j^t at time t, this edge is defined as $e_{i,j}^t \in \varepsilon^t$.

The definition for dynamic graph anomaly detection can be expressed as, given a dynamic graph $\Gamma = \{G^t\}_{t=1}^{T}$, where $G^t = \{\upsilon^t, \varepsilon^t\}$ represents a snapshot of the dynamic graph at time t. Determine the anomaly score $S(e_{i,j}^t)$ for each edge $e_{i,j}^t \in \varepsilon^t$ at time t, where $S(*)$ is the trained anomaly score function. The probability that an edge is anomalous is represented by the anomaly score, the higher the number, the greater the probability that the edge is anomalous.

Table 1
The interpretation of characters.

Characters	Interpretation
$\Gamma = \{G^t\}_{t=1}^{T}$	Dynamic graph with maximum timestamp of T
$G^t = \{\upsilon^t, \varepsilon^t\}$	Snapshot of the dynamic graph of t moments
υ^t	Set of nodes at time t
ε^t	Set of edges at time t
$v_i^t \in \upsilon^t$	The ith node of the snapshot at time t
$e_{i,j}^t \in \varepsilon^t$	The edge of the ith node and the jth node of the snapshot at time t
A	Diffusion Matrix
$a_{e_{tgt}} = a_{v_1} + a_{v_2}$	The connectivity vector of e_{tgt}
$B(e_{tgt}^t)$	Sub-structure node set
α	Time window size
k	Context nodes number
$S(x)$	Abnormal score function
β	Coefficient of exception score $S(x)$
G	The generator
D	The discriminator
E	The encoder

3.2. Substructure sampling

As pointed out in previous work (Cai et al., 2021; Liu, Li, et al., 2021; Wang, Wang, et al., 2022), anomalies usually occur in local substructures of the graph, so instead of carrying out anomaly detection on the whole dynamic graph, this paper considers that it is better to start from sampling the substructures to detect the anomalous data in them. Since this paper focus on edge anomaly detection, this paper employ edge-based substructure sampling (Liu, Pan, et al., 2021). For each object edge in this study, the network diffusion technique (Hassani & Khasahmadi, 2020; Klicpera, Weißenberger, & Günnemann, 2019) is used by us to abstract an importance-aware and size-specific contextual node sets. For a given static network matrix $M \in \mathbb{R}^{n\times n}$, this paper gives the definition of the diffusion matrix $A \in \mathbb{R}^{n\times n}$:

$$A = \sum_{i=0}^{\infty} \theta_i \mathbf{T}^i. \tag{1}$$

where the generalized transpose matrix is denoted by $\mathbf{T} \in \mathbb{R}^{n\times n}$, and θ is the weighting factor that decides how much information is global and how much is local. At the same time, in order to ensure convergence, $\sum_{i=0}^{\infty} \theta_i = 1$, $\theta_i \in [0, 1]$, and the eigenvalue $\lambda_i \in [0, 1]$ of the matrix $\mathbf{T}$ is required.

The ith row of the diffusion matrix A is able to represent the connectivity of the ith node. Thus after creating a diffusion matrix for a static graph with a single timestamp, for a given node, the connectivity of that node can be obtained using the diffusion director and a connectivity vector can be composed from it. For an edge $e_{tgt} = (v_1, v_2)$, its connection vector is calculated by summing the connection vector of the two target nodes:

$$a_{e_{tgt}} = a_{v_1} + a_{v_2}. \tag{2}$$

In the above equation, from a global perspective, the link between the ith connection point and that each access point is represented by a_i. Then the set of contextual nodes $C(e_{tgt})$ is formed by the way This paper sort the set of connectivity vectors then pick the top-k nodes for the largest values. Finally, the union of the context node set and the target node, indicated as $B(e_{tgt} = v_1, v_2, C(e_{tgt}))$, can be used to represent the sampled node set of the substructure.

Similarly, for dynamic networks, consider using multiple timestamps to capture dynamic evolution. Given a target edge with timestamp, the connectivity vector of the ith timestamp node, the substructure node set is sampled as $a_{e_{tgt}^t}^i$, thereafter, the complete list of foundation system nodes is obtained:

$$B(e_{tgt}^t) = \bigcup_{i=t-\alpha+1}^{t} C^i(e_{tgt}^t). \tag{3}$$

The received field on the time axis is determined by a hyper-parameter of size *alpha* called the time window.

3.3. Space–time node encoding

One challenge of the dynamic graphs studied in this paper over images with original features and attribute graphs is that dynamic graphs are often attribute free. In order to construct information encoding from attribute-free dynamic graphs as input to graph generation adversarial networks, this paper adopts a spatio-temporal node encoding method for dynamic graph transformation. The coding method used in this paper is divided into three parts: relative time coding, diffusion-based spatial coding, and distance-based spatial coding. The global and local structural roles of each node are represented by the two terms of spatial encoding, respectively. Each component of the substructure node set is given a time value by the time code item. The three encoding methods are then combined to create an input node encoding that includes extensive spatiotemporal data.

To start, this paper uses diffusion-based spatial encoding to provide global view data for each node structure in the graph. In order to sort all nodes in the node set $v_j^i \in C^i(e_{tgt}^t)$, this paper is based on the diffusion value pair single timestamp substructure, and also uses the sorting as the generated data source. A learnable encoding function is created, as follows to calculate the diffusion-based spatial encoding:

$$\mathrm{x}_{diff}(v_j^i) = linear(rank(a_{e_{tgt}}^i[idx(v_j^i)])) \in \mathbb{R}^{d_{enc}}. \tag{4}$$

Among these, $idx(\bullet)$ is the index query function, $rank(\bullet)$ is the ranking function, d_{enc} is the node processing dimensionality, and $linear(\bullet)$ is the learnable linear representation.

Meanwhile, to represent local connections near the edges of objects, a distance-based spatial coding approach is applied in this paper (see Figs. 1–8).

To be more precise, for each node $v_j^i \in C^i(e_{tgt}^t)$ which in the single set of timestamped substructure nodes, Its proximity from the intended edge is encoded in order for the collected data to be defined. This paper defines the form of spatial encoding based on distance as follows:

$$\mathrm{x}_{dist}(v_j^i) = linear(min(dist(v_j^i, v_1^i), dist(v_j^i, v_2^i))) \in \mathbb{R}^{d_{enc}}. \tag{5}$$

where $dist(\bullet)$ is the relative distance calculation function, $min(\bullet)$ is the minimum function, $linear(\bullet)$ is the learnable linear mapping.

Each node is represented by time information called time code in the set of substructure nodes. In the case of anomaly identification, the time interval of object edges is a more important component, since the main work of this paper is to predict the legitimacy of object edges. In Vaswani et al. (2017), This paper take into account relative encoding of dynamic graph rather than absolute encoding. The difference between the goal edge occurrence time t and the present timestamp is defined as the relative time-encoded data source for each node in the set of node for the substructure. In a formal sense, relative time encoding is expressed as:

$$\mathrm{x}_{temp}(v_j^i) = linear(\|t - i\|) \in \mathbb{R}^{d_{enc}}. \tag{6}$$

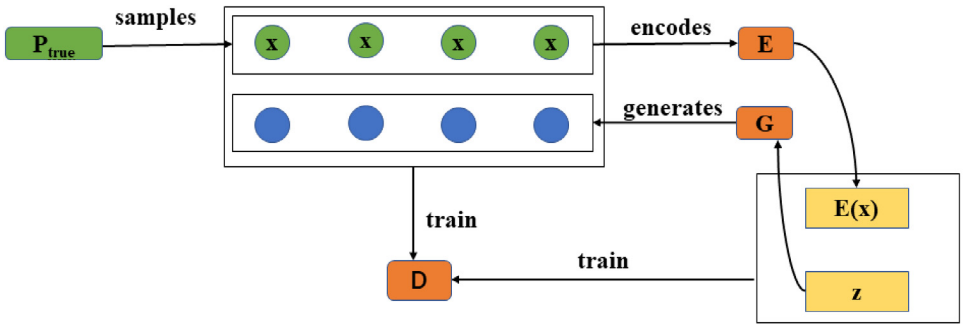

Fig. 1. RegraphGAN model. The model consists of a generator G, an encoder E and a discriminator D.

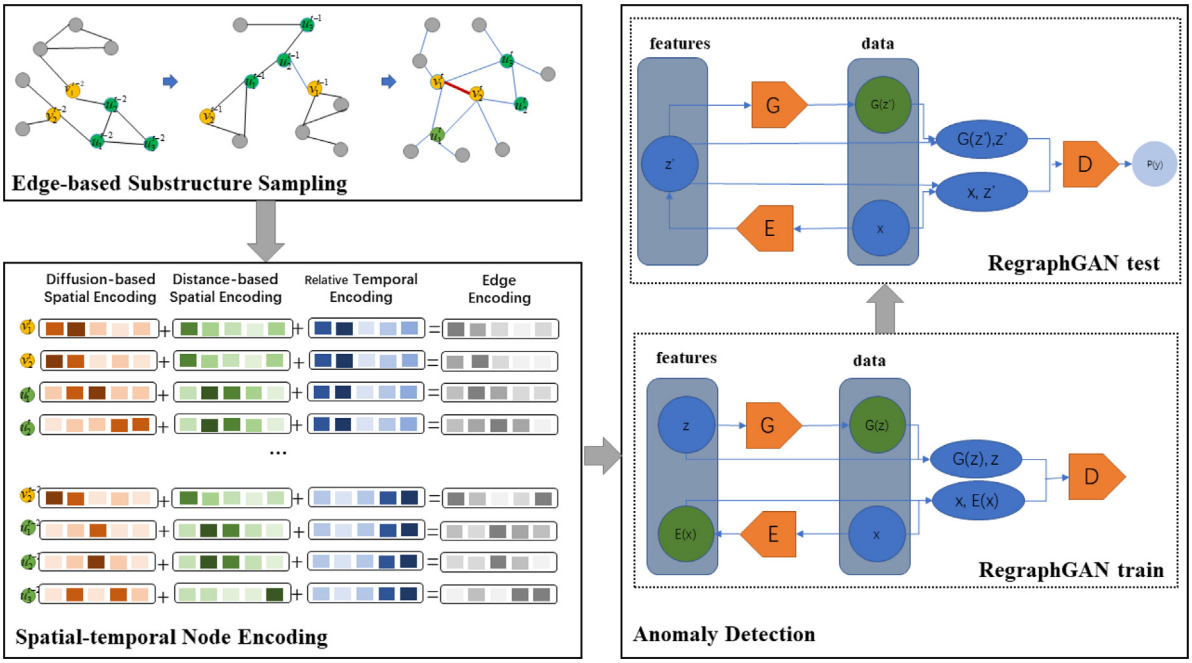

Fig. 2. RegraphGAN's overall structure. The framework is made up of three components: a reverse generative adversarial network anomaly detector, spatial–temporal node encoding, and edge-based substructure sampling.

where $\| \bullet \|$ is relative time calculation function, $linear(\bullet)$ is the learnable linear mapping.

Finally, to improve operational efficiency, the encoding of the fusion nodes is defined not by concatenating them into a high-dimensional vector but as the sum of three encoding terms. Here is the form of code fusion:

$$\mathrm{x}(v_j^i) = \mathrm{x}_{diff}(v_j^i) + \mathrm{x}_{dist}(v_j^i) + \mathrm{x}_{temp}(v_j^i) \in \mathbb{R}^{d_{enc}}. \tag{7}$$

Therefore, for a given target edge, the encoding of each node in the collection of its substructure nodes is calculated and superimposed into an encoding matrix representing the properties of e_{tgt}^t. As shown in the encoding matrix:

$$\mathrm{X}_{e_{tgt}^t} = \bigoplus_{v_j^i \in B(e_{tgt}^t)} [\mathrm{x}(v_j^i)]^{\mathrm{T}} \in \mathbb{R}^{(\alpha(k+2)) \times d_{enc}}. \tag{8}$$

In the formula, $\bigoplus$ and $[\bullet]^{\mathrm{T}}$ are the splicing operation and the transposing operation, respectively.

3.4. RegraphGAN

This section presents a new graph generation adversarial network proposed in this paper. Unlike previous graph generation adversarial networks, the RegraphGAN proposed in this paper simultaneously learns an encoder E that maps input samples x to potential representations z, as well as a generator G and a discriminator D during the training process. thus avoiding the challenging and time-consuming process of recovering potential representations at test time. In a normal graph generative adversarial network, the discriminator only looks at the input (whether it is real or generator-generated), but in RegraphGAN, the discriminator D also needs to discriminate the underlying representations (generator input or encoder output).

In the graph generation adversarial network proposed in this paper:

- Generator G, which tries to approximate the true connectivity distribution of the underlying layer and generates (or

selects, if more precise) the vertices most likely to be connected to v_i from the vertex set v. In this paper VGAE (Kipf & Welling, 2016b) is used as the generator of the model.
- Discriminator D, which distinguishes between the connectivity of the vertex pair (v, v_i) and whether the potential representation is considered as input from the generator or from the encoder.
- Encoder E, which is a mapping of the input samples x of this encoder into a potential representation z. In this paper, the transformer network is used as the encoder of the model.

Suppose $p_X(x)$ is the data distribution for $x \in \Omega_X$. The generator model's objective is to use a probabilistic model to simulate this data distribution. However, this precise modeling of the probability density function is computationally challenging, and the existing graph generation adversarial network (Wang et al., 2023) instead models a translation of the fixed potential distribution into the data distribution $p_Z(z)$ for $z \in \Omega_Z$. A deterministic feedforward network G is used to represent this transition, which is known as a generator: $\Omega_Z \to \Omega_X$ where $p_G(x|z) = \delta(x - G(z))$ and $p_G(x) = \mathbb{E}_{z\sim p_Z}[p_G(x|z)]$. The objective is to hone a generator so that $p_G(x) \approx p_X(x)$.

The traditional graph generation adversarial network framework trains a graph generator with the aim of making it difficult for the discriminative model D to distinguish between samples from the real data distribution and samples from the generated distribution. Both the generator and the discriminator are learned using the adversarial objective $min_G max_D V(D, G)$, with $V(D, G)$ defined as:

$$V(D, G) = \mathbb{E}_{x\sim p_X}[log D(x)] + \mathbb{E}_{x\sim p_G}[log(1 - D)] \tag{9}$$

Unlike the traditional graph generation adversarial network described above, in RegraphGAN we train not only the generator but also the encoder E: $\Omega_X \to \Omega_Z$ additionally. The distribution $p_E(z|x) = \delta(z - E(x))$ caused by the encoder PE converts the data point x into the generative model's latent feature space. The discriminator is additionally changed so that it can take inputs from the potential space and predict $P_D(Y|x, z)$, where $Y = 1$ if x is true (chosen at random from the real data distribution p_X) and $Y = 0$ if x is generated by the generator (production of $G(z)$).

An effective semantic representation should be learned using a model that is trained to predict a feature z given data x. Here, this paper demonstrates how the RegraphGAN aim drives the encoder E to do just that: the encoder must invert the generator at a given z so that $E(G(z)) = z$ in order to spoof the discriminator at that z.

In order to apply anomaly detection to dynamic graphs, the RegraphGAN thinks about different ways to train the encoder $E = G^{-1}$, with a focus on how important it is to learn both E and G. In the training phase, the generator G tries to generate as realistic data samples as possible from the noise, the discriminator D is used to distinguish the generated samples from the real samples, and the input samples x are mapped to the potential representation z by the encoder E. Consequently, this paper utilized a similar approach, resolving the following optimization issue during training: $min_{G,E} max_D V(D,E,G)$, with $V(D, E, G)$ defined as:

$$\begin{aligned} V(D, E, G) = {} & \mathbb{E}_{x\sim p_X}[\mathbb{E}_{z\sim p_E(.|x)}[log D(x, z)]] \\ & + \mathbb{E}_{x\sim p_Z}[\mathbb{E}_{z\sim p_G(.|z)}[1 - log D(x, z)]] \end{aligned} \tag{10}$$

where $px(x)$ and $pz(z)$ represent the distribution of real data and the distribution of latent features, respectively, and $p_E(z|x)$ represents the distribution generated by the encoder and $p_G(x|z)$ represents the distribution generated by the generator.

3.5. Anomaly detection

Algorithm 1 The General Training Plan for RegraphGAN.

Input: Dynamic networks training set: $\Gamma = \{G^t\}_{t=1}^T$, how many training epochs there are: I, sampled number of contextual nodes: k, Size of the time period: α

```
Set the settings for the RegraphGAN model, the scoring function, and the encoding linear mappings.
for i ∈ [1, I] do
  for timestamp G^t = (v^t, ε^t) in {G^t}_{t=α}^T do
    Generate edge ε_n^t samples by G
    for e ∈ ε_n ∪ ε_n^t do
      The objective edge should be e and sample its substructure node set C(e) with α(k + 2) nodes
      Calculate node encoding matrix X(e) via Eqs. (5)–(8)
      Train the RegraphGAN model according to Eq. (10)
      Calculate anomaly score S(e) via Eq. (11)
    end for
    Calculate loss function L by L_D and L_G
    Updating the parameters and back propagation
  end for
end for
```

The training process of the graph generation adversarial network proposed in this paper has been described in the previous section. As a graph generation adversarial network that can be used for dynamic graph anomaly detection, the calculation of anomaly detection score is set as the last step of the whole work in the testing phase, which can judge the quality of the framework. The anomaly score of each edge is calculated by the score function and compared with the set value, and when it is greater than the set value, the edge is judged to be an anomaly, and anomaly detection of dynamic graph is realized. Learn anomaly scores for each edge and use them for dynamic networks anomaly detection. Having trained a model on the normal data to yield G, D and E, this paper then define a score function $S(x)$ that measures how anomalous an example x is, based on a convex combination of a reconstruction loss L_G and a discriminator-based loss L_D:

$$S(x) = \beta L_G(x) + (1 - \beta) L_D(x) \tag{11}$$

Where $Z(x)$ represents the embedding of edge x, $L_G(x) = \|Z(x) - G(E(x))\|$ and $L_D(x)$ can be defined in two ways. In the first place, using a real-world example of the cross-entropy loss σ from the discriminator of x (class 1): $L_D(x) = \sigma(D(E(x)), 1)$, this accurately reflects the discriminator's level of assurance that a sample was drawn from the actual data distribution. Using a "feature-matching loss" is a second technique to define the L_D: $L_D(x) = \|f_D(x, E(x)) - f_D(G(E(x)), E(x))\|$, $f_D(\bullet)$ then returns the discriminator's layer, which is the layer that precedes the logits for the supplied inputs. This determines whether the rebuilt data shares characteristics with the true sample in the discriminator or not. It is assumed that samples with higher $S(x)$ values are more likely to be abnormal.

4. Experiments

In this section, the effectiveness of the proposed approach in this paper is experimentally demonstrated using real-world datasets. In Table 1, this paper gives the specific number of nodes and edges of the datasets.

Table 2
The statistical information for the dataset.

Dataset	#edge	#node	Avg.degree
Bitcoin-Alpha	24,173	3777	12.80
Bitcoin-OTC	35,588	5,001	12.10
UCI Message	13,838	1,899	14.57
Digg	85,155	30,360	5.61
AS-Topology	171,420	34,761	9.86
Email-DNC	1,866	39,264	42.08

4.1. Datasets

Primarily, this paper rely on six real-world datasets. In Table 2 shows the information of edge, node, and avg.degree of the dataset.

UCI Messages (Opsahl & Panzarasa, 2009) is a collection of social network data from the University of California, Irvine online student community. Each user is represented by each node in the constructed dynamic network, and information between users is represented by each edge.

The Bitcoin-Alpha (Kumar, Spezzano, Subrahmanian, & Faloutsos, 2016) dataset is the Bitcoin user trust network, and the data comes from transactions on the www.btc-alpha.com website Nodes in the Bitcoin-Alpha dataset represent users of the site, and edges are created when one user rates another.

Bitcoin-OTC (Kumar et al., 2018) is comparable to Bitcoin-Alpha, and the data originates from transaction information on the platform atwww.bitcoidtc.com Nodes are also platform users, and edges appear when users rate one another.

Digg (De Choudhury, Sundaram, John, & Seligmann, 2009) is a web dataset that has been compiled from the news website digg.com. In the Digg dataset, a user of the site corresponds to a node, and a user's response to another user is represented by each edge.

AS-Topology (Zhang, Liu, Massey, & Zhang, 2005) is a dataset of network connections collected from world wide web autonomous systems. Each autonomous system corresponds to a node in the network, and the connection between two autonomous systems is represented by an edge in the network.

Email-DNC (Rossi & Ahmed, 2015) is the email network from the 2016 Democratic National Committee email breach. Each person in the U.S. Democratic Party corresponds to a node in the network, and an edge is created when a person sends an email to another person.

4.2. Baselines

This paper compared the RegraphGAN framework to five cutting-edge baselines that fall into two categories: strategies for detecting anomalies in deep dynamic networks and networks embedding methods.

Breadth-first traversal and depth-first traversal are considered in node2vec (Grover & Leskovec, 2016) to generate random walks. Also Skip-gram technique is employed in node2vec to learn node embedding.

A based on random walks approach for net embedding is called DeepWalk (Perozzi, Al-Rfou, & Skiena, 2014) is considered as a random walk based net embedding method. Similar to how DeepWalk learns embeddings of unattributed networks, it produces random walks starting at the target node that are a certain length long.

A typical anomaly detection technique for dynamic networks is NetWalk (Yu et al., 2018). With the use of a random walk-based technique, it provides contextual data, and an auto-encoder model is used to learn node embedding. A reservoir-based approach is used to incrementally update the node embeddings over time. A dynamically updated the clustering based on the mastered embedding is used to find the anomaly.

A dynamic net anomaly detection technique called AddGraph (Zheng et al., 2019) is end-to-end. The spatial data is captured by AddGraph using the GCN module, and the short-term and long-term dynamic evolution of the network are extracted using the GRU-attention module.

StrGNN (Manzoor et al., 2016) is a net neural network model that is also end-to-end. In dynamic networks, strGNN is used to find anomalous edges. H-hop enclosing subnets are used by it as input to the network, and GCN and GRU are combined by StrGNN to learn the structural time information of each edge.

TADDY (Liu, Pan, et al., 2021) constructs a comprehensive node encoding strategy to better represent the structural and temporal roles of each node in the evolution graph flow. Simultaneously, information representations are captured from dynamic graphs with coupled spatiotemporal patterns via a dynamic graph transformer model. Two core challenges of anomaly detection in dynamic graphs are addressed: the lack of information encoding for non-attributed nodes, and the difficulty in learning discriminative knowledge from coupled spatiotemporal dynamic graphs.

4.3. Experimental design

Each data set was divided into two subsets in the experiments of this paper: the training set is made up of the first 50% of the timestamps, and the test set is made up of the remaining 50%. Three different anomaly ratios are considered in this paper, while adding the data anomalies in the test set at the rates of 1%, 5%, and 10%. Our main statistic, ROC–AUC (AUC for short), is used to assess the effectiveness of the suggested both the baselines and the framework. The relationship between the true positive rate and the false positive rate is shown by the ROC curve, where "positive" is used as a label to indicate abnormalities. AUC, or the amount of space beneath the ROC curve shows the probability of producing an abnormal edge with a higher score than a normal edge. Higher AUC values indicate better performance in detecting anomalies, while the range of AUC values is limited to 0 to 1.

Also give the parameter settings for this paper, in the preprocessing stage, this paper sets the number of context nodes k to 5 and the time window α takes values from 1 to 4, using $\beta = 0.9$ in the score $S(x)$, and the dimension of encoding denc to 512. The model is trained by Adam optimizer with a learning rate of 0.0001. On the UCI Messages, Digg, and Email-DNC datasets, this paper uses 0.01 as the abnormal value and judge the input edge as abnormal when its abnormal value is greater than 0.01 and as normal when it is less than 0.01. The discriminators are trained once in every 5 training sessions of the generator and encoder. Unlike the above datasets, on the Bitcoin-OTC, Bitcoin-Alpha and AS-Topology datasets, a different training strategy is used in this paper, where the generator and encoder are trained once for each discriminator in one calendar time. The experiment trained 20 epcohs on all datasets and finally achieved the best training model. The snapshot size is set to be 1000 for UCI Messages and Email-DNC, 2000 for Bitcoin-Alpha and Bitcoin-OTC, and 4000 for AS-Topology and Digg respectively.

4.4. Anomaly detection results

The performance of anomaly detection is reported in this subsection, and our suggested RegraphGAN architecture is compared to the industry standard approaches. Table 2 compares the average AUC for all test timestamps to show the performance of anomaly detection. For the findings, This paper condenses the following observations:

Table 3
Auc measures report anomaly detection performance comparison.

Methods	UCI Messages			Bitcoin-Alpha			Bitcoin-OTC		
	1%	5%	10%	1%	5%	10%	1%	5%	10%
node2vec	0.7371	0.7433	0.6960	0.6910	0.6820	0.6875	0.6951	0.6883	0.6745
DeepWalk	0.7514	0.7391	0.6979	0.6985	0.6874	0.6793	0.7423	0.7356	0.7287
NetWalk	0.7758	0.7647	0.7226	0.8385	0.8357	0.8350	0.7785	0.7694	0.7534
AddGraph	0.8083	0.8090	0.7688	0.8665	0.8403	0.8498	0.8352	0.8455	0.8592
StrGNN	0.8179	0.8252	0.7959	0.8574	0.8667	0.8627	0.9012	0.8775	0.8836
TADDY	**0.8912**	0.8398	0.8370	0.9451	0.9341	0.9423	**0.9455**	0.9341	0.9432
RegraphGAN	0.8319	**0.8531**	**0.8665**	**0.9540**	**0.9600**	**0.9650**	0.9195	**0.9454**	**0.9664**
Methods	Digg			AS-Topology			Email-DNC		
	1%	5%	10%	1%	5%	10%	1%	5%	10%
node2vec	0.7364	0.7081	0.6508	0.6821	0.6752	0.6668	0.7391	0.7284	0.7103
DeepWalk	0.7080	0.6881	0.6396	0.6844	0.6793	0.6682	0.7481	0.7303	0.7197
NetWalk	0.7563	0.7176	0.6837	0.8018	0.8066	0.8058	0.8105	0.8371	0.8305
AddGraph	0.8341	0.8470	0.8369	0.8080	0.8004	0.7926	0.8393	0.8627	0.8773
StrGNN	0.8162	0.8254	0.8272	0.8553	0.8352	0.8271	0.8775	0.9103	0.9080
TADDY	0.8617	0.8545	0.8440	0.8953	0.8952	0.8934	0.9348	0.9257	0.9210
RegraphGAN	**0.8831**	**0.8924**	**0.8952**	**0.9008**	**0.9052**	**0.9112**	**0.9382**	**0.9412**	**0.9465**

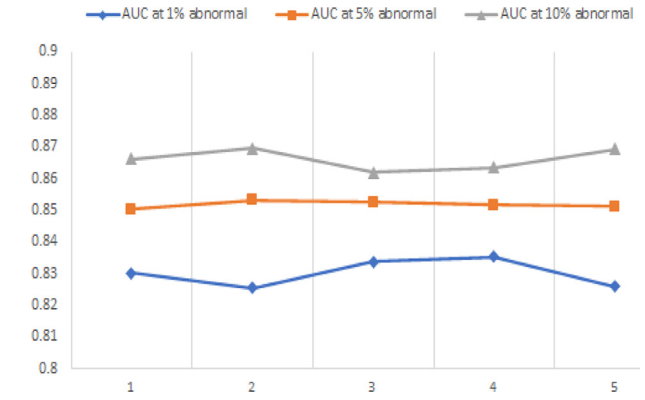

Fig. 3. AUC results for outlier scores obtained from tests taken at different snapshot on the UCI.

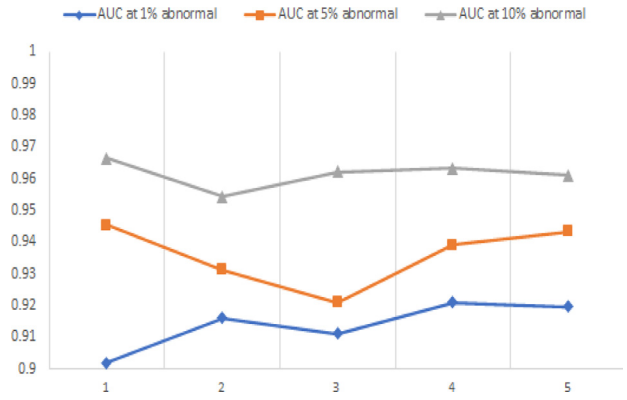

Fig. 5. AUC results for outlier scores obtained from tests taken at different snapshot on the Bitcion-OTC.

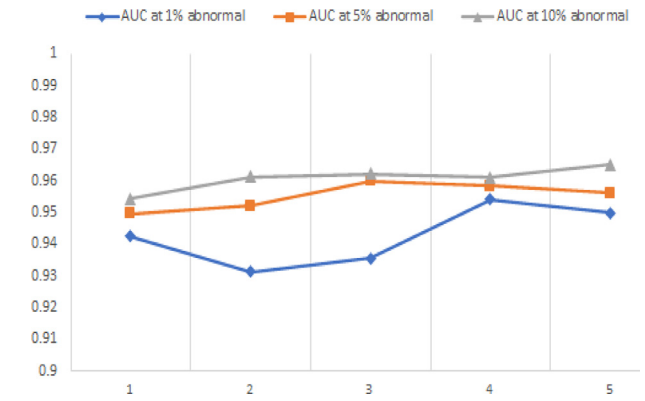

Fig. 4. AUC results for outlier scores obtained from tests taken at different snapshot on the Bitcoin-Alpha.

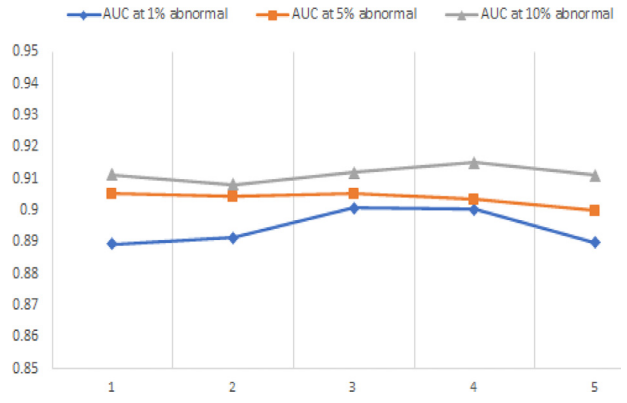

Fig. 6. AUC results for outlier scores obtained from tests taken at different snapshot on the Digg.

- On six datasets of dynamic networks with different anomaly proportions, the proposed RegraphGAN framework consistently outperforms all baselines. RegraphGAN achieves a significant performance improvement on the value of the AUC when compared to the baseline approach that produced the greatest results. The major reason is that RegraphGAN captures both structural and temporal dynamics simultaneously with a reverse generation countermeasure network encoder and assembles an informative node encoding to extract spatial–temporal information.
- The deep dynamic networks anomaly detection techniques NetWalk, AddGraph, StrGNN, and RegraphGAN consistently outperform three networks embedding-based methods. This paper credits the use of temporal information for this performance advantage. These algorithms learn the dynamic development in networks by taking into account the interaction in prior timestamps.

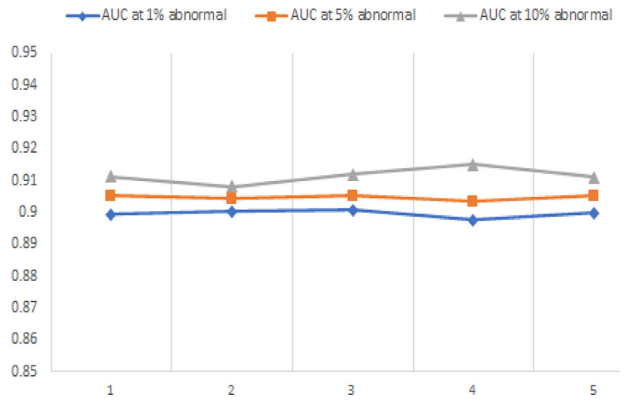

Fig. 7. AUC results for outlier scores obtained from tests taken at different snapshot on the AS-Topology.

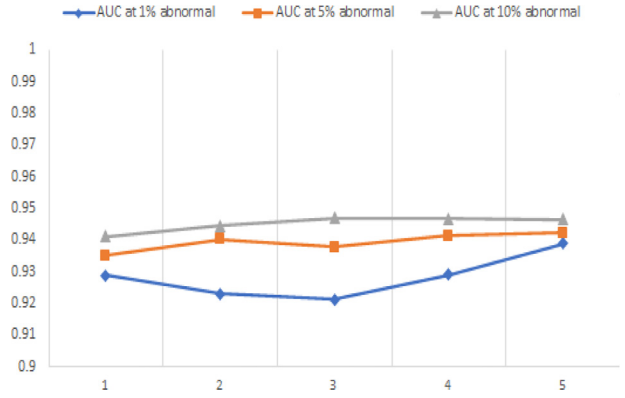

Fig. 8. AUC results for outlier scores obtained from tests taken at different times on the Email-DNC.

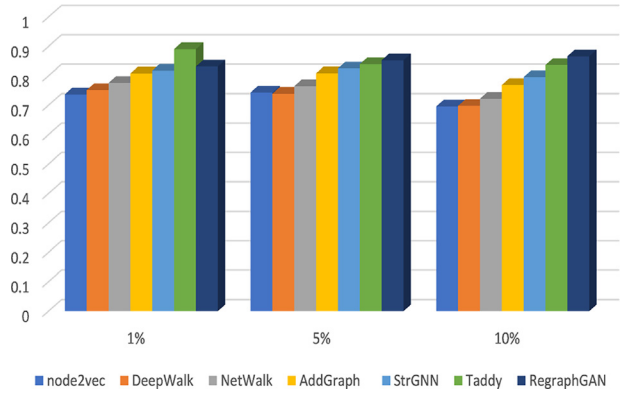

Fig. 9. Performance comparison of different methods for different outliers in UCI dataset.

- Table 3 shows the test results, and Figs. 9–14 show that compared with other baselines, the RegraphGAN proposed in this paper is more or less superior in AUC value on the used data set, showing ideal performance. Since spatial–temporal dynamics are more directly related to the irregularity of Bitcoin transactions, RegraphGAN can effectively capture these dynamics through extensive node embedding and attention techniques. This shows the significant advantages of RegraphGAN in anomaly detection.

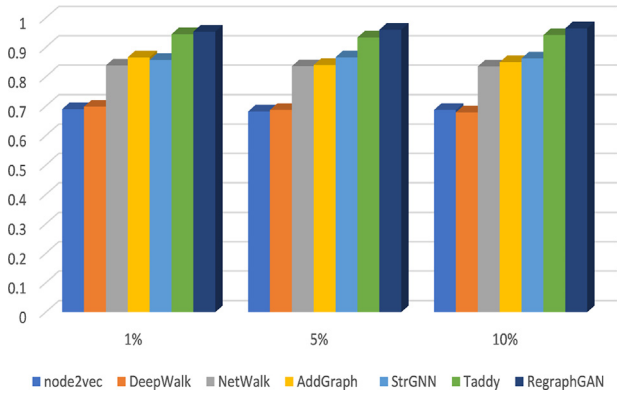

Fig. 10. Performance comparison of different methods for different outliers in Bitcion-Alpha dataset.

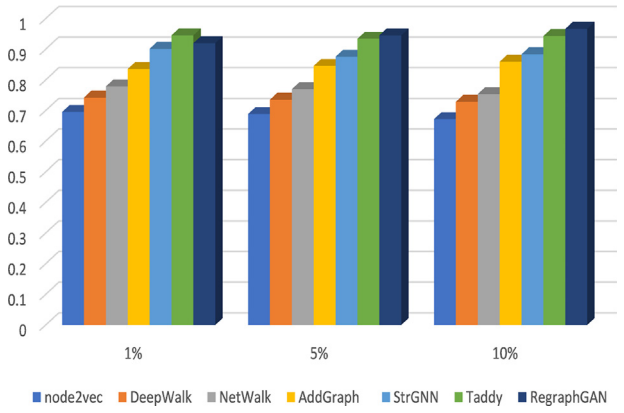

Fig. 11. Performance comparison of different methods for different outliers in Bitcion-OTC dataset.

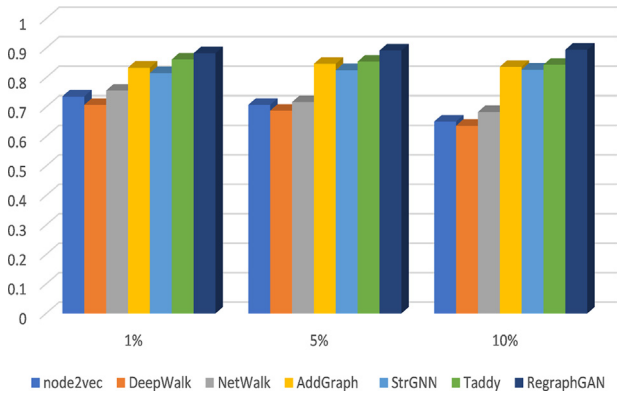

Fig. 12. Performance comparison of different methods for different outliers in Digg dataset.

4.5. Ablation experiments

In this subsection, the sensitivity of some parameters and the role of some components are experimentally illustrated. The comparative experiments in this paper use three data sets of UCI Message, Bitcoin-Alpha, and Bitcoin-OTC.

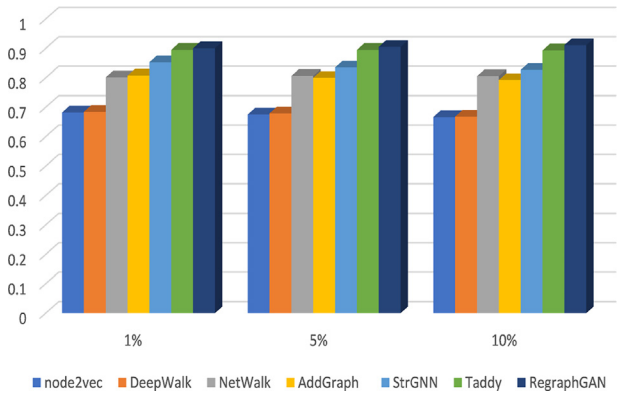

Fig. 13. Performance comparison of different methods for different outliers in AS-Topology dataset.

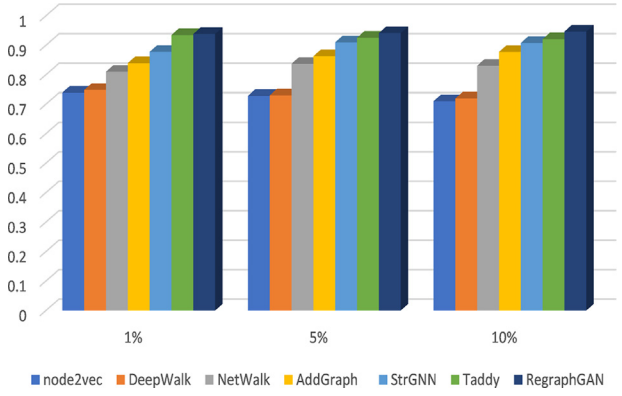

Fig. 14. Performance comparison of different methods for different outliers in Email-DNC dataset.

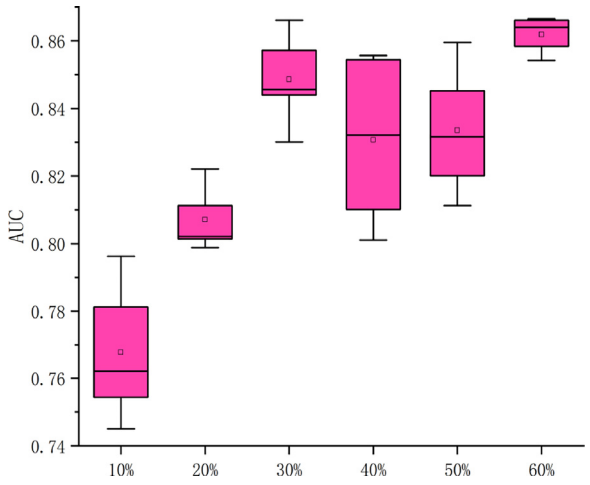

Fig. 15. Results of AUC on UCI using various training ratios.

- First, for different training ratios, the results are shown in Figs. 15 to 17. In this paper, we used 20% to 60% of the training set to train the anomaly detection model, respectively, and the AUC values increased smoothly with increasing training ratios. This is mainly because the model in this paper is semi-supervised and only normal data is needed for training, so a larger proportion of the training set tends to give better results.

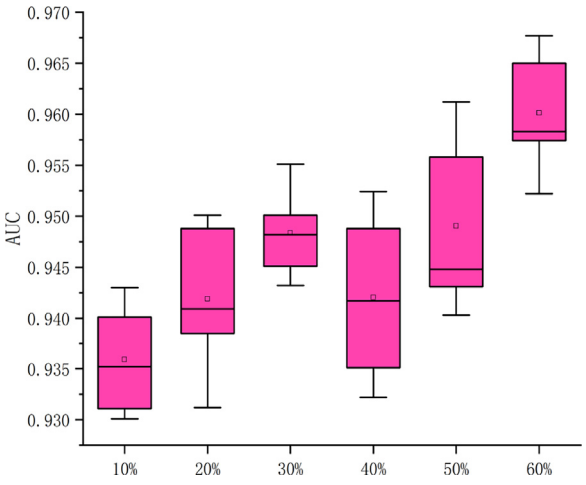

Fig. 16. Results of AUC on Bitcion-Alpha using various training ratios.

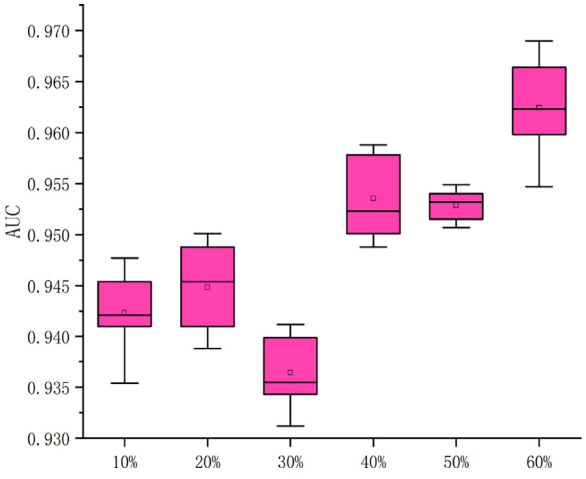

Fig. 17. Results of AUC on Bitcion-OTC using various training ratios.

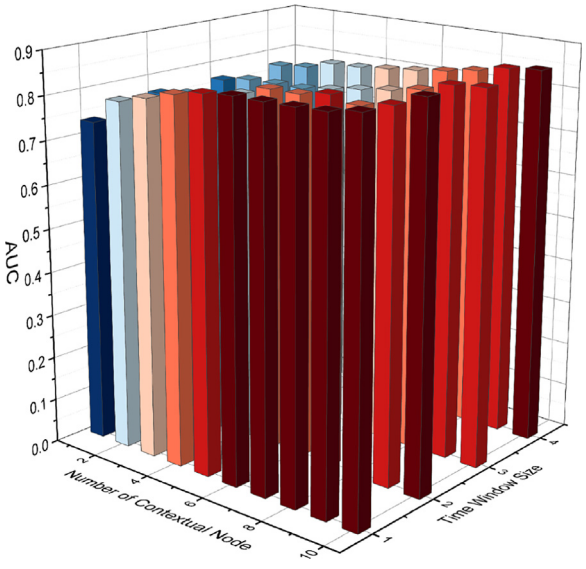

Fig. 18. Parameter sensitivity experiments on UCI.

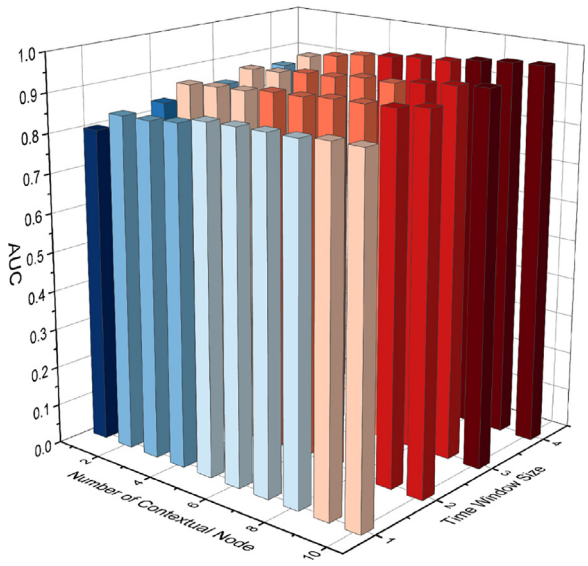

Fig. 19. Parameter sensitivity experiments on Bitcion-Alpha.

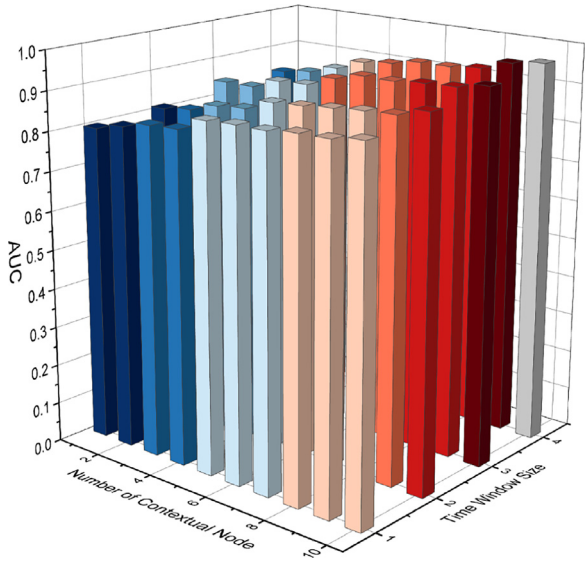

Fig. 20. Parameter sensitivity experiments on Bitcion-OTC.

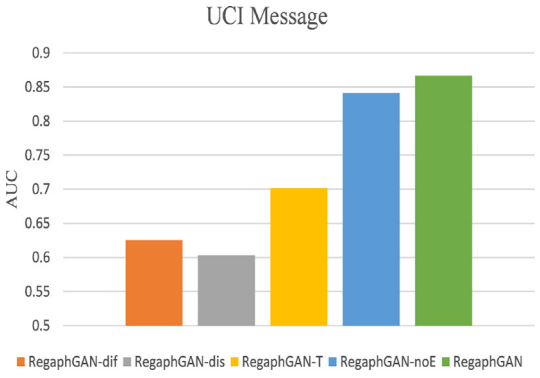

Fig. 21. Comparison of detection results between RegraphGAN and variants on UCI.

- Comparative experiments are also conducted in this paper for different time windows α and number of contextual nodes k. Specifically, the values of α range from 1 to 4 and k from 1 to 10. For different datasets, the best detection results correspond to different time windows. The detection results for different cases are shown in Figs. 18 to 20. The performance increases as the number of context nodes increases, mainly because the properties of an edge tend to depend on its adjacent local structure. The most likely reason is the different degree of time dependence of the edges of different data sets.
- In order to verify the effectiveness of each component, an ablation experiment for the component is designed in this paper. As shown in Figs. 21–23, RegaphGAN-dif indicates that only diffusion-based spatial encoding is used, RegaphGAN-dis indicates that only distance-based spatial encoding is used, RegaphGAN-T indicates that only the relative time encoding method is used, and RegaphGAN-noE denotes the model with the encoder removed. It is not difficult to see from the experimental comparison chart that the complete model performs relatively better in anomaly detection.
- Although RegraphGAN-noE shows a more desirable anomaly detection, efficiency should also be considered. When the number of edges to be tested is n, it needs to go through n times of mapping from the encoder to the underlying representation and then reconstructed by the generator, so the time complexity of our model is about $O(n)$ when tested. When we remove the encoder E, the reconstruction of the generator needs to be obtained from m representations, so the time complexity is about $O(mn)$. For example, we found in our tests that the time consumed by the UCI message test after removing the encoder changed from the 150–200 s to 7000–9000 s, and tests on both bitcoin datasets went from

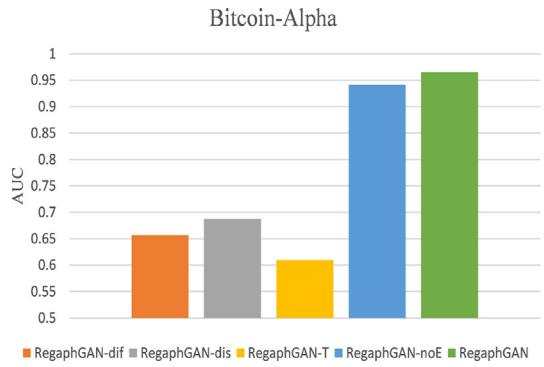

Fig. 22. Comparison of detection results between RegraphGAN and variants on Bitcion-Alpha.

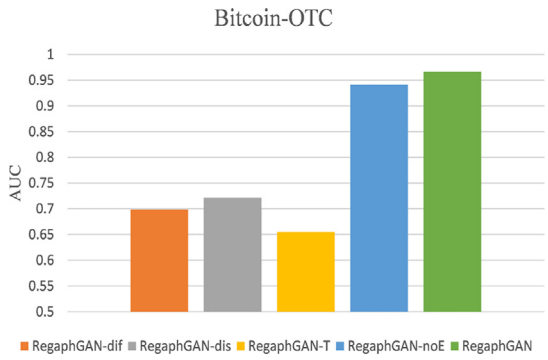

Fig. 23. Comparison of detection results between RegraphGAN and variants on Bitcion-OTC.

40–50 s to 1000–1700 s. As a result, the addition of encoders has increased efficiency by a factor of 20 to 60.

5. Conclusion

In this paper, the generative adversarial network RegraphGAN on graph is proposed for the first time, and an encoder is introduced to improve the traditional graph generative adversarial network, and the framework is applied to the dynamic graph anomaly detection link. In addition, this paper adopts edge-based substructure sampling, spatiotemporal node encoding, combined with the RegraphGAN anomaly detection module, to successfully identify edge anomalies in complex dynamic networks, and achieve good results on six real-world dynamic network datasets. The dynamic graph anomaly detection framework proposed in this paper has been shown to produce very good results in experiments on anomalous edge detection in dynamic networks, and the achieved results are significantly better than those obtained using most existing methods.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62272405, School and Locality Integration Development Project of Yantai City (2022), the Youth Innovation Science and Technology Support Program of Shandong Provincial under Grant 2021KJ080, the Natural Science Foundation of Shandong Province, China under Grant ZR2022MF238, Yantai Science and Technology Innovation Development Plan Project under Grant 2022XDRH023.

References

Aggarwal, C. C., Zhao, Y., & Philip, S. Y. (2011). Outlier detection in graph streams. In *2011 IEEE 27th international conference on data engineering* (pp. 399–409). IEEE.

Akcay, S., Atapour-Abarghouei, A., & Breckon, T. P. (2018). Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian conference on computer vision* (pp. 622–637). Springer.

Bojchevski, A., Shchur, O., Zügner, D., & Günnemann, S. (2018). Netgan: Generating graphs via random walks. In *International conference on machine learning* (pp. 610–619). PMLR.

Cai, L., Chen, Z., Luo, C., Gui, J., Ni, J., Li, D., et al. (2021). Structural temporal graph neural networks for anomaly detection in dynamic graphs. In *Proceedings of the 30th ACM international conference on information & knowledge management* (pp. 3747–3756).

De Choudhury, M., Sundaram, H., John, A., & Seligmann, D. D. (2009). Social synchrony: Predicting mimicry of user actions in online social media. In *2009 international conference on computational science and engineering, vol. 4* (pp. 151–158). IEEE.

Ding, M., Tang, J., & Zhang, J. (2018). Semi-supervised learning on graphs with generative adversarial nets. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 913–922).

Donahue, J., Krähenbühl, P., & Darrell, T. (2016). Adversarial feature learning. arXiv preprint arXiv:1605.09782.

Eswaran, D., Faloutsos, C., Guha, S., & Mishra, N. (2018). Spotlight: Detecting anomalies in streaming graphs. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 1378–1386).

Gao, Y., Xiaoyong, L., Hao, P., Fang, B., & Yu, P. (2020). Hincti: A cyber threat intelligence modeling and identification system based on heterogeneous information network. *IEEE Transactions on Knowledge and Data Engineering*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139–144.

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 855–864).

Hassani, K., & Khasahmadi, A. H. (2020). Contrastive multi-view representation learning on graphs. In *International conference on machine learning* (pp. 4116–4126). PMLR.

Hooi, B., Song, H. A., Beutel, A., Shah, N., Shin, K., & Faloutsos, C. (2016). Fraudar: Bounding graph fraud in the face of camouflage. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 895–904).

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Philip, S. Y. (2021). A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(2), 494–514.

Jia, Y., Zhang, Q., Zhang, W., & Wang, X. (2019). Communitygan: Community detection with generative adversarial nets. In *The world wide web conference* (pp. 784–794).

Jiao, P., Guo, X., Jing, X., He, D., Wu, H., Pan, S., et al. (2021). Temporal network embedding for link prediction via vae joint attention mechanism. *IEEE Transactions on Neural Networks and Learning Systems*.

Kipf, T. N., & Welling, M. (2016a). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Kipf, T. N., & Welling, M. (2016b). Variational graph auto-encoders. arXiv preprint arXiv:1611.07308.

Klicpera, J., Weißenberger, S., & Günnemann, S. (2019). Diffusion improves graph learning. arXiv preprint arXiv:1911.05485.

Kumar, S., Hooi, B., Makhija, D., Kumar, M., Faloutsos, C., & Subrahmanian, V. (2018). Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM international conference on web search and data mining* (pp. 333–341).

Kumar, S., Spezzano, F., Subrahmanian, V., & Faloutsos, C. (2016). Edge weight prediction in weighted signed networks. In *2016 IEEE 16th international conference on data mining* (pp. 221–230). IEEE.

Li, R., Liu, Z., Ma, Y., Yang, D., & Sun, S. (2022). Internet financial fraud detection based on graph learning. *IEEE Transactions on Computational Social Systems*.

Liu, Y., Li, Z., Pan, S., Gong, C., Zhou, C., & Karypis, G. (2021). Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(6), 2378–2392.

Liu, Y., Pan, S., Wang, Y. G., Xiong, F., Wang, L., Chen, Q., et al. (2021). Anomaly detection in dynamic graphs via transformer. *IEEE Transactions on Knowledge and Data Engineering*.

Liu, X., & Song, P. (2022). Incomplete multi-view clustering via virtual-label guided matrix factorization. *Expert Systems with Applications*, *210*, Article 118408.

Manzoor, E., Milajerdi, S. M., & Akoglu, L. (2016). Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1035–1044).

McConville, R., Liu, W., & Miller, P. (2015). Vertex clustering of augmented graph streams. In *Proceedings of the 2015 SIAM international conference on data mining* (pp. 109–117). SIAM.

Opsahl, T., & Panzarasa, P. (2009). Clustering in weighted networks. *Social Networks*, *31*(2), 155–163.

Peng, H., Yang, R., Wang, Z., Li, J., He, L., Philip, S. Y., et al. (2021). Lime: Low-cost and incremental learning for dynamic heterogeneous information networks. *IEEE Transactions on Computers*, *71*(3), 628–642.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 701–710).

Ranshous, S., Harenberg, S., Sharma, K., & Samatova, N. F. (2016). A scalable approach for outlier detection in edge streams using sketch-based approximations. In *Proceedings of the 2016 SIAM international conference on data mining* (pp. 189–197). SIAM.

Rossi, R., & Ahmed, N. (2015). The network data repository with interactive graph analytics and visualization. In *Twenty-ninth AAAI conference on artificial intelligence*.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., & Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging* (pp. 146–157). Springer.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., & Webb, R. (2017). Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2107–2116).

Sricharan, K., & Das, K. (2014). Localizing anomalous changes in time-evolving graphs. In *Proceedings of the 2014 ACM SIGMOD international conference on management of data* (pp. 1347–1358).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Wang, Y., Liu, Z., Xu, J., & Yan, W. (2022). Heterogeneous network representation learning approach for ethereum identity identification. *IEEE Transactions on Computational Social Systems*.

Wang, W., Wang, Y., Duan, P., Liu, T., Tong, X., & Cai, Z. (2022). A triple real-time trajectory privacy protection mechanism based on edge computing and blockchain in mobile crowdsourcing. *IEEE Transactions on Mobile Computing*.

Wang, H., Wang, J., Wang, J., et al. (2023). Graph representation learning with generative adversarial nets. AAAI, Graphgan.

Wang, L., Yu, Z., Xiong, F., Yang, D., Pan, S., & Yan, Z. (2019). Influence spread in geo-social networks: a multiobjective optimization perspective. *IEEE Transactions on Cybernetics*, *51*(5), 2663–2675.

Yang, C., Zhou, L., Wen, H., Zhou, Z., & Wu, Y. (2020). H-vgrae: A hierarchical stochastic spatial-temporal embedding method for robust anomaly detection in dynamic networks. arXiv preprint arXiv:2007.06903.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., & Shen, Y. (2020). Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, *33*, 5812–5823.

Yu, W., Cheng, W., Aggarwal, C. C., Zhang, K., Chen, H., & Wang, W. (2018). Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2672–2681).

Zenati, H., Romain, M., Foo, C.-S., Lecouat, B., & Chandrasekhar, V. (2018). Adversarially learned anomaly detection. In *2018 IEEE international conference on data mining* (pp. 727–736). IEEE.

Zhang, B., Liu, R., Massey, D., & Zhang, L. (2005). Collecting the internet AS-level topology. *ACM SIGCOMM Computer Communication Review*, *35*(1), 53–61.

Zhao, Y., & Yu, P. S. (2013). On graph stream clustering with side information. In *Proceedings of the 2013 SIAM international conference on data mining* (pp. 139–150). SIAM.

Zheng, Y., Hu, R., Fung, S.-f., Yu, C., Long, G., Guo, T., et al. (2020). Clustering social audiences in business information networks. *Pattern Recognition*, *100*, Article 107126.

Zheng, L., Li, Z., Li, J., Li, Z., & Gao, J. (2019). AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN. In *IJCAI* (pp. 4419–4425).

Zheng, Z., Zheng, L., & Yang, Y. (2017). Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE international conference on computer vision* (pp. 3754–3762).

Zhou, S., Song, P., Yu, Y., & Zheng, W. (2023). Structural regularization based discriminative multi-view unsupervised feature selection. *Knowledge-Based Systems*, *272*, Article 110601.