IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 5, 2021, accepted April 12, 2021, date of publication April 16, 2021, date of current version April 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3073696

Anomalous Subgraph Detection in Given Expected Degree Networks With Deep Learning

MINGAN LUAN[1,2], (Graduate Student Member, IEEE), BO WANG[2], YANPING ZHAO[2], AND FENGYE HU[2], (Senior Member, IEEE)

[1]State Key Laboratory of Automotive Simulation and Control, Jilin University, Changchun 130025, China
[2]College of Communication Engineering, Jilin University, Changchun 130025, China

Corresponding author: Bo Wang (roywang@jlu.edu.cn)

This work was supported in part by the Foundation of State Key Laboratory of Automotive Simulation and Control under Grant 20180108, in part by the National Natural Science Foundation of China under Grant 61771217, in part by the Science and Technology Project of Jilin Province Department of Education under Grant JJKH20190164KJ, and in part by the Natural Science Foundation of Jilin Province under Grant 20180101051JC.

ABSTRACT Anomalous subgraph detection within networks is an important issue in many emerging applications. Existing algorithms, such as graph structure methods and spectral feature methods, usually focus on the special stochastic model (such as the Erdős-Rényi random graph) or may not efficiently extract the anomalous behaviors of the networks, which result in detection performance degradation. To mitigate the limitations, in this paper, we first present an anomalous subgraph detection framework associated with deep neural networks (DNN) for detecting anomalous behaviors within the networks. Furthermore, based on the developed framework, we propose a residual matrix-based convolutional neural network (RM-CNN) algorithm with respect to the given expected degree models, which are more general networks than the Erdős-Rényi random graphs. In particular, the trained RM-CNN can efficiently capture the anomalous changes of the network and then achieve the detection performance improvement. Simulation experiments display that the proposed RM-CNN algorithm is superior to the compared algorithms in both detection performance and detection speed.

INDEX TERMS Anomalous subgraph detection, given expected degree models, deep learning, convolutional neural network.

I. INTRODUCTION

Network anomaly detection has become an important issue in may applications, ranging from detecting malicious attacks of wireless networks [1], [2], vehicle anomaly detection [3], to array diagnostics [4]. Generally, we are interested in network data regarding communication connections (relationships) between nodes (entities) in anomalous detection problem, and the data are usually represented as a graph form [5], where nodes (entities) and their communication connections are denoted by vertices and edges of the graph, respectively. According to the stochastic characteristics of the network data, the stochastic models of the network can be classified into several categories, including Erdős-Rényi random graph [6], stochastic block random graph [7], power law random graph models [8], etc.

Commonly, there are just a small amount of the data exhibiting anomalous behaviors in anomaly detection problem. Therefore, the anomalous detection is also named as the anomalous subgraph detection. In the past decade, there are a tremendous amount of works to be done for the anomalous detection, and many detection algorithms are developed for overcoming the related problems. These algorithms can be divided into two kinds of methods: graph structure methods [9]–[15] and spectral feature methods [16]–[19].

Graph structure methods obtain the detection result by measuring certain network structure features or invariants, such as the number of triangle subgraphs, scan statistics, total degree, and so on. For example, [13] fused the vertex connection probability with the triangle subgraph to improve the detection performance of the algorithm and obtained the detection boundary of statistic through the emerging concentration measure theory. Reference [14] applied the local belief propagation algorithm to collect the vertex neighborhood message to implement the subgraph detection in the Erdős-Rényi random graph models. To make full use of the structural information between vertices, [15] proposed an anomaly detection method in a strong noise environment by mapping the matrix of random graph into a higher-order tensor. However, these methods have limitations in models

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

and other aspects. For instance, [14] required a large amount of prior information, which makes the algorithm hard to implement in many real scenarios. In particular, most algorithms are only effective for special random graph model, i.e., Erdős-Rényi random graphs.

On the other hand, spectral feature methods exploit the eigenvectors of the adjacency or residual matrix of random graphs to detect the anomalous patterns of the random networks. The typical algorithms [17]–[19] include the eigenvector L_1 norms method, chi-squared statistic method, and sparse principal component analysis method, etc. These methods can detect the existence of anomalies in complex networks, but in the large-scale network, these algorithms suffer from performance degradation and also have significant computation complexity.

In order to overcome the model limitation and performance degradation mentioned above, we attempt to explore an efficient anomaly detection method, which can be significantly suitable for the general network models, i.e., the given expected degree models [20]. Recently, deep learning [21] has been widely used in various engineering fields, such as rolling bearing fault diagnosis, waveform recognition, wireless localization, hypothesis testing and vehicular congestion detection, [22]–[26], etc. The related literatures show that through constructing different deep neural network (DNN) structures, the learning-type algorithms can provide better performance than their counterparts. For example, [26] proposed a robust deep learning method to realize congestion detection in vehicular management. Reference [23] explored the neural network that optimized for the hypothesis testing problem. Thanks to the powerful feature extraction capability of deep learning, we extend the new tool into the anomaly detection problem for the given expected degree networks in this paper. In view of the powerful capability of the deep learning, this paper develops an anomalous subgraph detection framework, named DNN-based subgraph detection framework.

Based on the developed detection framework, we propose a residual matrix-based convolutional neural network (RM-CNN) algorithm for the anomalous subgraph detection associated with the given expected degree models. The given expected degree models [20] (Chung-Lu random graphs) are the general random network model and has complex structure, which is a main challenge for anomaly detection. Besides, the dimensions of anomalous subgraph are much smaller than the background graph in general. Namely, the so-called signal to noise ratio (SNR) of the anomalous detection problem is quite low [13]. As a result, the anomalous behavior may not be extracted and directly results in the low detection performance of the traditional algorithms. To overcome those problems, our paper develops a convolutional neural network (CNN) based on the residual matrix of anomalous networks for extracting the anomalous structure. Specifically, the RM-CNN first captures the anomalous features of the graphs and obtains the optimal network parameters at the training stage. Then, according to the *Neyman-Pearson* theorem, we construct a detection statistic, which can determine the anomalous behavior of the graph with the maximum probability of detection for a given probability of false alarm. Theoretical analysis and simulation experiments show that the proposed algorithm has significant advantages in both detection performance and computational complexity concerned with the on-line phase than the compared algorithms.

The main contributions of this paper are as follows.

- This paper introduces a DNN-based subgraph detection framework. The DNN collects graph features that can extract the network structure during the offline phase, and we design a detection statistic by the extracted features. Compared with traditional algorithms, the framework not only has powerful feature extraction capabilities but also can conveniently extend to other network models. Moreover, to the best of our knowledge, it is the first attempt to apply deep learning technology to the subgraph anomaly detection problem.
- Based on the proposed DNN-based subgraph detection framework, we develop a RM-CNN algorithm associated with the given expected degree models to detect the anomalous behaviors of the networks. Theoretical analysis shows that the presented algorithm has lower computation complexity regarding the on-line phase than some typical methods. Simulation results also validate that compared with the typical algorithms, the proposed algorithm has a significant improvement in terms of both detection performance and detection speed.

The remainder of this paper is structured as follows. Section II introduces the random graph models and the problem statement. In Section III, we design the DNN-based anomalous subgraph detection framework. Next, Section IV provides the RM-CNN algorithm. The simulation examples are shown in Section V. The conclusion of this paper is summarized in Section VI.

Notation: Throughout this work, we apply the lowercase a, boldface lowercase $\mathbf{a}$, and boldface uppercase $\mathbf{A}$ stand for scalar, vector, and matrix, respectively. $\mathbb{E}(\cdot)$ stands for the expectation operator. $\|\cdot\|_F$ and $\|\cdot\|_2$ represent the Frobenius norm and 2 norm, respectively. $*$ denotes the convolution operator, and $\lfloor\cdot\rfloor$ stands for the round down operation. Finally, we list the acronyms appeared in the this paper in Table 1.

TABLE 1. The list of acronyms.

Acronyms	Description
DNN	Deep Neural Network
RM	Residual Matrix
CNN	Convolutional Neural Network
SNR	Signal to Noise Ratio
ROC	Receiver Operating Characteristics

II. GRAPH MODEL AND PROBLEM STATEMENT

In this section, we introduce some notations used in this paper, describe the given expected degree random graph

models, and formulate the anomalous subgraph detection problem.

A. NOTATION AND RANDOM GRAPH MODEL

Consider a network consisting of N nodes, and let the random graph $G = (V, E)$ represent the random network, where V is the vertex set with $N = |V|$ and E is the edge set. The notation $G_F = (V_F, E_F)$ stands for the anomalous subgraph in which V_F and E_F are the anomalous vertex and edge sets with $N_F = |V_F|$, respectively. In this paper, the random graph G that contains the anomalous subgraph is obtained via the union operation [17], i.e, $G = G_N \cup G_F = (V_N \cup V_F, E_N \cup E_F)$, where $G_N = (V_N, E_N)$ represents background graph. We define d_n as the expected degree of the vertex n and let $\mathbf{d} = (d_1, \cdots, d_n, \cdots, d_N)$ be the expected degree sequence of the random graph G. Correspondingly, the observed degree sequence of G is denoted by $\mathbf{k} = (k_1, \cdots, k_n, \cdots, k_N)$, where k_n is the number of one-hop neighbors of vertex n [27]. Note that $\mathbb{E}(k_n) = d_n$.

In this paper, we discuss the anomalous subgraph detection problem with respect to the given expected degree models (Chung-Lu random graphs) and assume that the expected degree sequence $\mathbf{d}$ is known. The expected degree sequence $\mathbf{d}$ regarding the given expected degree models is an arbitrary degree distribution, which is the generalization form of other random graph models, e.g., Erdős-Rényi and power law random graphs, etc. Let $\mathbf{A}$ denote the adjacency matrix of the random graph G, which stocks the relationship data between vertices. That is to say, if there is an edge between vertices i and j, (i, j) element $a_{ij} = 1$ of $\mathbf{A}$, otherwise, $a_{ij} = 0$. The variable a_{ij} belongs to a Bernoulli distribution with probability $p_{ij} \to [0, 1]$, which represents the link probability between vertices i and j. Correspondingly, for the given expected degree models, the link probability matrix is given by a $N \times N$ matrix $\mathbf{P}$, where $\mathbf{P} = \mathbf{d}\mathbf{d}^T \Big/ \sum_{n=1}^{N} d_n$. The random graph models considered in this paper are unweighted, undirected and no self-loops. Hence, $\mathbf{A}$ and $\mathbf{P}$ are symmetric matrices and $a_{ii} = 0$ for $i = 1, 2, \cdots, N$. From the above description, we have $d_i = \sum_{j=1}^{N} p_{ij}$, $k_i = \sum_{j=1}^{N} a_{ij}$, and $\mathbb{E}(\mathbf{A}) = \mathbf{P}$. Additionally, if vertices i and j belong to the vertex set V_F, the link probability in regard to edge set E_F equals to p_f, i.e., the anomalous subgraph is an Erdős-Rényi graph with the link probability p_f.

B. ANOMALOUS SUBGRAPH DETECTION PROBLEM

The main purpose of the anomalous subgraph detection problem is to determine whether there exists an anomalous subgraph in the observation graph or not. The anomalous behavior is that the anomalous subgraph has denser connections than the normal pattern. For example, in vehicular management [28], we usually represent the vehicle as a vertex of the random graph G and detect if there exists an unexpected activity in G, such as a traffic jam or accident, which can be denoted by an anomalous subgraph embedded in background graph. Statistically speaking, we can formalize the anomalous subgraph detection problem into a binary hypothesis test. The null hypothesis H_0 denotes the normal observed graph $G = G_N$, whereas the alternative hypothesis H_1 stands for the background graph G_N that is embedded an anomalous subgraph G_F, which can be written as

$$\begin{cases} H_0 : G = G_N \\ H_1 : G = G_N \cup G_F, \end{cases} \tag{1}$$

where the number of anomalous vertices $N_F \ll N$ in general.

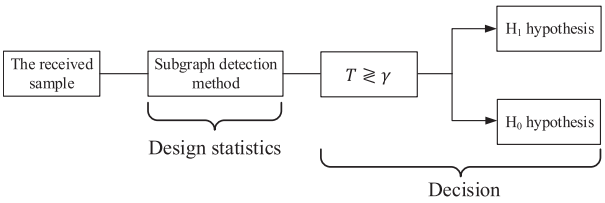


FIGURE 1. Conventional anomalous subgraph detection framework.

As shown in Fig. 1, the traditional subgraph detection methods usually contain two steps: first, design a test statistic T according to the features selected by the detector; next, compare T with the corresponding threshold to make the decision. Then, the probability of false alarm p_{fa} and the probability of detection p_d of the detector are

$$\begin{cases} p_{fa} = P(T > \gamma | H_0) \\ p_d = P(T > \gamma | H_1), \end{cases} \tag{2}$$

respectively, where γ is the detection threshold. If $T > \gamma$, the detector declares the alternative hypothesis H_1; otherwise, the null hypothesis H_0. Therefore, the main goal of the subgraph detection algorithm is to design a test statistic T so that it can maximize p_d for a given p_{fa} [29].

III. ANOMALOUS SUBGRAPH DETECTION FRAMEWORK BASED ON DEEP LEARNING

Motivated by the powerful information extraction capabilities of DNN, we develop a DNN-based subgraph detection framework in this section. As shown in Fig. 2, in which the blue and red dashed lines indicate normal and abnormal communication links, respectively, and the framework mainly includes two stages: offline training and on-line detection. The role of DNN is to extract graph features for constructing detection statistic.

A. OFFLINE TRAINING

1) TRAINING SET

At the offline phase, we first construct a random graph set to train the designed DNN, given by

$$(\mathbf{X}, \mathbf{Y}) = \{(x^{(1)}, y^{(1)}), \cdots, (x^{(l)}, y^{(l)}), \cdots, (x^{(L)}, y^{(L)})\}, \tag{3}$$

where $(x^{(l)}, y^{(l)})$ represents the lth, $l = 1, 2, \cdots, L$, sample of the training set $(\mathbf{X}, \mathbf{Y})$. Take the sample $(x^{(l)}, y^{(l)})$ as an example, $x^{(l)}$ denotes the input data for DNN, and $y^{(l)}$ is the label of the lth sample. More specifically, we declare the alternative hypothesis H_1 and the null hypothesis H_0 when

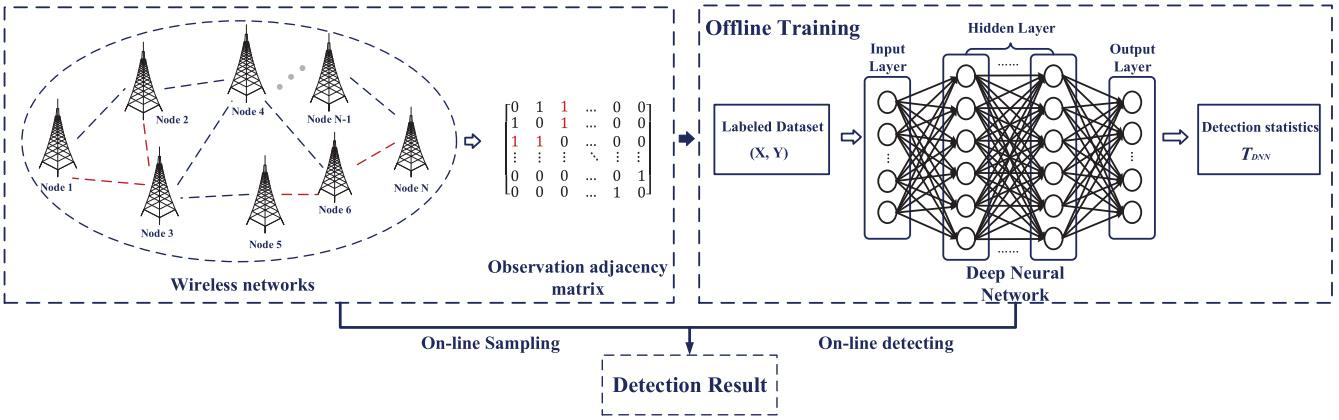


FIGURE 2. Anomaly subgraph detection framework based on deep neural networks.

$y^{(l)} = 2$ and $y^{(l)} = 1$, respectively. Please note that the sample set $\mathbf{X}$ consists of the matrix representations of the observed random graphs. In general, there are several types of the matrix representations associated with the random graph, such as the laplacian matrix, adjacency matrix, and residual matrix, etc. Based on the specific graph models, we can choose one certain matrix representation to form the sample set. Moreover, we assume that the vertex set of each sample in the training set is fixed but the edge is variable according to the link probability matrix $\mathbf{P}$.

2) TRAINING OF DNN

It can be seen from Fig. 2 that the DNN roughly includes the input layer, hidden layer, and output layer. Then, the hidden layer can further divide into the convolutional layer, pooling layer, etc. In general, the output of the hidden layer can be simplified into the following form

$$Y_H^{(l)} = f^{(K)}(f^{(K-1)} \cdots f^{(1)}(x^{(l)})), \tag{4}$$

where K is the number of convolutional layers, and the output of the kth convolutional layer, $k = 1, 2, \cdots, K$, can be expressed as

$$f^{(k)}(z^{(k-1)}) = f(W^{(k)} z^{(k-1)} + b^{(k)}), \tag{5}$$

where $W^{(k)}$ and $b^{(k)}$ denote the weight matrix and the offset of the kth convolutional layer, respectively. $z^{(k-1)}$ represents the output of the $(k-1)$th convolutional layer.

Let the $Y_H^{(l)}$ denote the graph feature vector extracted from the sample $x^{(l)}$, such as edge, diagonal and neighborhood features, etc, which are hard to be extracted by the traditional detection ways. In order to make the output of the DNN suitable for the binary hypothesis testing problem, we select softmax operator to fine-tune $Y_H^{(l)}$, then the final output is

$$f_\theta(x^{(l)}) = \begin{bmatrix} f_\theta^{(H_0)}(x^{(l)}) \\ f_\theta^{(H_1)}(x^{(l)}) \end{bmatrix}, \tag{6}$$

where $\theta = [\mathbf{W}, \mathbf{b}]$ denotes the network parameter set of the DNN, $\mathbf{W}$ and $\mathbf{b}$ stand for the weight matrix and the offset sets, respectively. $f_\theta^{(H_i)}(x^{(l)}) \in [0, 1]$ represents the score of $x^{(l)}$ corresponding to H_i, $i = 0, 1$, can be expressed as

$$f_\theta^{(H_0)}(x^{(l)}) = \frac{e^{W_S^0 Y_H^{(l)}}}{\sum_{i=0}^{1} e^{W_S^i Y_H^{(l)}}}, \tag{7}$$

$$f_\theta^{(H_1)}(x^{(l)}) = \frac{e^{W_S^1 Y_H^{(l)}}}{\sum_{i=0}^{1} e^{W_S^i Y_H^{(l)}}}, \tag{8}$$

where W_S^i is a weight matrix of the fine-tune operator, and $\sum_{i=0}^{1} f_\theta^{(H_i)}(x^{(l)}) = 1$. Moreover, for the sample $x^{(l)}$, the conditional probability associated with the ith hypothesis under θ [21] can be written as

$$H_i : P(y^{(l)} = i | x^{(l)}; \theta) = f_\theta^{(H_i)}(x^{(l)}), \ i = 0, 1. \tag{9}$$

Then, we define the following log-likelihood [30], [31]

$$\begin{aligned} J(\theta) &= \log P(Y|X; \theta) \\ &= \sum_{l=1}^{L} y^{(l)} \log f_\theta(x^{(l)}) + (1 - y^{(l)}) \log(1 - f_\theta(x^{(l)})). \end{aligned} \tag{10}$$

So far, the goal of offline training is to learn an optimal parameter θ^* to maximize $J(\theta)$, i.e.,

$$\theta^* = \arg\max_\theta J(\theta). \tag{11}$$

We apply the back-propagation algorithm to solve the problem (11). Hereto, we can obtain the posterior probability $P(H_i|x) = f_{\theta^*}^{(H_i)}(x)$ for hypothesis H_i. According to the *Neyman-Pearson* theorem, we can construct the following detection statistic for anomalous subgraph detection problem

$$T_{\mathrm{DNN}}(x) = \frac{P(x|H_1)}{P(x|H_0)} = \frac{P(H_1|x)}{P(H_0|x)} \cdot \frac{P(H_0)}{P(H_1)} = \frac{f_{\theta^*}^{(H_1)}(x)}{f_{\theta^*}^{(H_0)}(x)}, \tag{12}$$

which can maximize the probability of detection p_d for a given probability of false alarm p_{fa}. Without loss of generality, we set $P(H_0) = P(H_1) = 0.5$ [29].

In order to make the steps of training the DNN offline clearer, we briefly summarize the training process. Firstly,

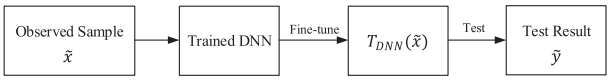


FIGURE 3. On-line detection.

a training set is constructed based on the specific form of the neural network. Secondly, we send samples to the DNN hidden layer to obtain feature vectors for capturing the state of the graph, and then achieve the final output (6). Finally, based on the back-propagation algorithm, the DNN can learn the optimal parameter θ^*, and the detection statistic $T_{\mathrm{DNN}}(x)$ can be constructed for anomalous subgraph detection.

B. ON-LINE DETECTION

Fig. 3 shows the on-line detection process of the algorithm. The trained DNN is first fed with the observed sample $\widetilde{x}$ to output the feature vector $\widetilde{Y}_H$, then combine (6) and (12) to get the detection statistic $T_{\mathrm{DNN}}(\widetilde{x})$. By comparing the statistic $T_{\mathrm{DNN}}(\widetilde{x})$ with the threshold γ, we can determine whether the observed graph contains anomalous subgraph

$$T_{\mathrm{DNN}}(\widetilde{x}) \underset{H_0}{\overset{H_1}{\gtrless}} \gamma. \tag{13}$$

The determination of the threshold γ will be described in the next section.

Remark 1: In order to overcome the limitations of the typical subgraph detection methods and improve the detection performance, we propose a DNN-based subgraph detection framework in this section. In fact, the intention of our framework is to improve detection performance by coupling the optimization of DNN with statistical decision-making. More in detail, we first define the log-likelihood loss function J_θ, which can obtain the posterior probability of the sample by learning the optimal parameters of the DNN. Then, we define a detection statistic T_{DNN} based on the *Neyman-Pearson* theorem, which is the optimal detection statistic for our anomalous subgraph detection problem.

IV. ANOMALOUS SUBGRAPH DETECTION ALGORITHM

Based on the DNN subgraph detection framework mentioned above, we develop a RM-CNN algorithm for anomalous detection regarding the given expected degree models in this section.

A. NETWORK CONSTRUCTION OF RM-CNN

It is shown in some typical algorithms (such as the Chi-Squared statistic methods [18] and the sparse principal component analysis [19]) that the residual matrix of a random graph can sense the deviation of the network. Therefore, we utilize the residual matrix $\mathbf{B}$ as the input of the CNN to capture the graph features. The residual matrix $\mathbf{B}$ of the observed random graph is given by

$$\mathbf{B} = \mathbf{A} - \mathbb{E}(\mathbf{A}) = \mathbf{A} - \frac{\mathbf{d}\mathbf{d}^T}{\sum_{i=1}^{N} d_i}, \tag{14}$$

where $\mathbb{E}(\mathbf{A})$ denote the expectation of the adjacency matrix $\mathbf{A}$. Then, the network structure of RM-CNN is shown in Fig. 4, and its details is given below.

1) INPUT LAYER

The factors that affect the offline training time of CNN are the depth of the network hidden layer and the size of the sample. In general, the size of the anomalous subgraph is considerable smaller than the background graph, which means that the sample is full of useless noise for the subgraph information. To reduce the training burden of the network, we first pre-denoise the input samples of the network. Specifically, we define the deviation degree sequence $\mathbf{k}_{dev}$ as

$$\mathbf{k}_{dev} = (b_1^2/\mathbb{D}(k_1), b_2^2/\mathbb{D}(k_2), \cdots, b_N^2/\mathbb{D}(k_N)), \tag{15}$$

where $\mathbb{D}(k_i) = \sum_{j=1}^{N} p_{ij}(1-p_{ij})$ represents the variance of k_i, and $b_i = \mathbb{E}(k_i) - k_i$ is the bias of k_i. Obviously, the larger $\mathbf{k}_{dev}(i)$, the more anomalous the corresponding vertex i. Therefore, we select M vertices corresponding to the largest M ($M > N_F$) elements in $\mathbf{k}_{dev}$ to form the vertex set $V_D = \{v_1, v_2, \cdots, v_M\}$ and use the residual submatrix $B_{M\times M}$ associated with the vertex set V_D as the input of RM-CNN. For simplicity, we use B_M instead of $B_{M\times M}$. It is worth pointing out that although the value of M has no significant impact on the detection performance, we generally choose $M > N_F$ to guarantee the algorithm's performance. Note that since $N_F \ll N$, the dimension M of the B_M is also much smaller than that of the background graph. Consequently, the training time of the RM-CNN offline training is reduced after pre-denoising. In addition, due to the elements of matrix B_M are all real numbers, the dimension of the input layer S_0 is $M \times M \times 1$.

2) CONVOLUTIONAL LAYERS

In the CNN, each convolutional layer is composed of multiple distinct convolution kernels, which can effectively extract the graph features of the random graph by performing convolution calculation with the input data. Compared with the graph features selected artificially, the convolution kernel extracts graph features from the training data can better sense the structure of the random graph. For a single convolutional layer, the convolution operation is written as

$$\begin{aligned} C(i,j,g) &= (I * K_g)(i,j) \\ &= \sum_m \sum_n I(m,n)K(i-m, j-n, g), \end{aligned} \tag{16}$$

where $C(\cdot,\cdot,g)$ represents the feature map of the gth convolution kernel, I is the input of the convolutional layer, and K_g stands for convolution kernel.

To weigh the algorithm performance and offline training time, we let the RM-CNN perform two convolution operations. As shown in Fig. 4, the first convolutional layer C_1 in RM-CNN applies 32 convolution kernels of size 5×5. Then, the RM-CNN obtains 32 distinct feature maps, i.e., graph features. Similarly, the second convolutional layer C_2 includes 50 convolution kernels of size 5×5 and extracts

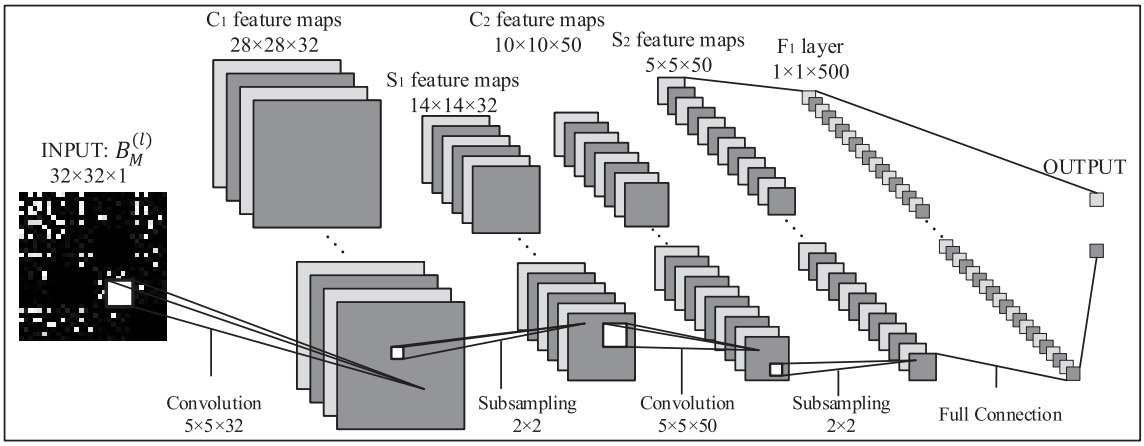


FIGURE 4. RM-CNN network architecture.

50 graph features. In the two convolutional layers, the convolution stride and padding are set as 1 and 0, respectively. Note that the size of the convolution kernel can be adjusted according to the size of the input sample. In the RM-CNN, the output of the convolutional layer is the linear function of the input. However, due to the complex network structure of the given expected degree models, the extracted linear features may not capture the anomalous behaviors of the networks. Hence, we introduce the activation function to extract the nonlinear graph features for detection performance improvement and also overcome the vanishing gradient problem. In this paper, the rectified linear unit (ReLU) is approved as the activation function, which can be expressed as $f(t) = \max(t, 0)$, where t is the convolutional layer output.

3) POOLING AND FULLY CONNECTED LAYERS

The pooling operation can reduce the size of the output of the convolutional layer and increase the offline training speed. The first (second) pooling layer S_1 (S_2) performs the maximum pooling operation on each 2×2 region of the input data, and the pooling stride is set to 2. It can be seen from Fig. 4 that the size of the sample is effectively reduced after the pooling layers.

The role of the fully connected layer is to integrate the features extracted from each convolutional layer. After two fully connected operations, the RM-CNN can obtain the graph feature vector Y_H. By fine-tuning the output of the second fully connected layer, we obtain the final output of RM-CNN, given by

$$f_\theta(B_M) = \begin{bmatrix} f_\theta^{(H_0)}(B_M) \\ f_\theta^{(H_1)}(B_M) \end{bmatrix}, \tag{17}$$

where $f_\theta^{(H_i)}(B_M)$ denotes the score of the ith hypothesis with the input of B_M.

As a summary, the required parameters for offline training are listed in Table 2. Note that the network depth only counts the number of convolutional layers.

TABLE 2. Network training parameters.

Training Parameters	Specification
Network Depth	2
Input Size	$32 \times 32 \times 1$
Convolution Kernel Size	First Layer $5 \times 5 \times 32$ Second Layer $5 \times 5 \times 50$
Max-Pooling	2×2
Full Connected Kernel Size	First Layer $5 \times 5 \times 500$ Second layer $1 \times 1 \times 2$
Epoch	20
Learning rate	0.001

Next, we analyze the number of the floating point operations (FLOPs) and the parameters of the RM-CNN. The number of the parameters for the ith convolutional layer is given by

$$N_c^{(i)} = (K_i^2 c_{i-1} + 1)c_i. \tag{18}$$

Then, the number of the parameters for the ith fully connected layer is expressed as

$$N_f^{(i)} = (N_{in}^{(i)} + 1)N_{out}^{(i)}, \tag{19}$$

where $N_{in}^{(i)}$ and $N_{out}^{(i)}$ denote the number of the input and output elements, respectively. For example, for the fully connected layer F_1, we have $N_{in}^{(1)} = 5 \times 5 \times 50$ and $N_{out}^{(1)} = 1 \times 1 \times 500$. Moreover, for the ith convolutional layer, we have

$$\text{FLOPs}_c^{(i)} = N_c^{(i)} m_i^2, \tag{20}$$

where m_i represents the length of the output feature map. Note that, for the fully connected layer, FLOPs_f is equal to N_f.

Remark 2: The computational complexity at the offline training stage mainly depends on the computational cost of the pre-denoising for the dataset and the solution for parameter θ^*. The computational complexity for finding the first M possible anomalous vertices of the lth sample is $O(N^2)$. The computational complexity of the solution for parameter θ^*

is reflected in the total number of samples and the number of epochs, which is $O(LN_e \sum_{i=1}^{D} m_i^2 K_i^2 c_{i-1} c_i)$. N_e, L, and D denote the number of epoch, samples, and convolution layers, respectively. It is worth noting that the size of m_i depends on the size of the input matrix of the ith layer, K_i, stride, and padding. Thus, the total computational complexity of the entire offline training is $O(LN^2 + LN_e \sum_{i=1}^{D} m_i^2 K_i^2 c_{i-1} c_i)$.

B. RM-CNN ALGORITHM

In this subsection, we introduce a RM-CNN algorithm for the given expected degree models as a specific implementation of the DNN-based subgraph detection framework.

1) OFFLINE TRAINING

As shown in Fig. 2, the system first collects L labeled observed graphs to form the labeled dataset

$$(\mathbf{X}, \mathbf{Y}) = \{(\mathbf{A}^{(1)}, y^{(1)}), \cdots, (\mathbf{A}^{(l)}, y^{(l)}), \cdots, (\mathbf{A}^{(L)}, y^{(L)})\}, \tag{21}$$

where $\mathbf{A}^{(l)}$ denotes the adjacency matrix of the lth ($l = 1, 2, \cdots, L$) sample. For a single sample $\mathbf{A}^{(l)}$, we need to pre-denoise it by formulas (14) and (15), and then we can obtain the residual matrix $B_M^{(l)}$ of lth sample. Therefore, we can define the training set, given by

$$(\mathbf{X}_{B_M}, \mathbf{Y}) = \{(B_M^{(1)}, y^{(1)}), \cdots, (B_M^{(l)}, y^{(l)}), \cdots, (B_M^{(L)}, y^{(L)})\}. \tag{22}$$

According to (10), the log-likelihood can be written as

$$J(\theta) = \sum_{l=1}^{L} y^{(l)} \log f_\theta(B_M^{(l)}) + (1 - y^{(l)}) \log(1 - f_\theta(B_M^{(l)})). \tag{23}$$

Through the back-propagation algorithm, we can obtain the optimal parameters θ^* of the RM-CNN. Then, the detection statistic of the proposed algorithm is given by

$$T_{\mathrm{RM-CNN}}(B_M) = \frac{f_{\theta^*}^{(H_1)}(B_M)}{f_{\theta^*}^{(H_0)}(B_M)}, \tag{24}$$

where $f_{\theta^*}^{(H_0)}(B_M)$ and $f_{\theta^*}^{(H_1)}(B_M)$ denote the score of B_M about H_0 and H_1 hypothesis, respectively. The procedure of the offline training process of RM-CNN is summarized in Algorithm 1.

Algorithm 1 RM-CNN Offline Training

1: Input: L labeled observed graphs $\{\mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(l)}, \cdots, \mathbf{A}^{(L)}\}$, expected degree sequence $\mathbf{d}$.
2: Output: RM-CNN network optimal parameter θ^*.
3: Initialize: Network parameters $\theta = \theta^1$, M, $r = 1$, and Epoch;
4: **for** $l = 1 : L$ **do**
5: Calculate the lth sample residual matrix $B_M^{(l)}$ by formulas (14) and (15);
7: **end for**
8: Build the training set $(\mathbf{X}_{B_M}, \mathbf{Y}) = \{(B_M^{(1)}, y^{(1)}), \cdots, (B_M^{(l)}, y^{(l)}), \cdots, (B_M^{(L)}, y^{(L)})\}$ and sent it to the RM-CNN;
9: **for** $r = 1 : Epoch$ **do**
10: Use the back-propagation algorithm to update the
11: parameters θ^r with a loss function of (23);
12: Update $r = r + 1$;
13: **end for**

2) ON-LINE DETECTION

During the on-line detection stage of the proposed RM-CNN algorithm, we first collect the online sample $\widetilde{\mathbf{A}}$. Then, the detector calculates the residual matrix $\widetilde{B}_M$ and feeds it to the trained RM-CNN to obtain statistic $T_{\mathrm{RM-CNN}}(\widetilde{B}_M)$. By comparing $T_{\mathrm{RM-CNN}}(\widetilde{B}_M)$ with a given threshold γ, we can judge whether the sample $\widetilde{G}$ contains the anomalous subgraph. If $T_{\mathrm{RM-CNN}}(\widetilde{B}_M) > \gamma$, we declare that $\widetilde{G}$ contains the anomalous subgraph (H_1 hypothesis); otherwise, we declare that $\widetilde{G}$ is normal (H_0 hypothesis). Note that due to the considerable number of RM-CNN parameters and the complex structure of the given expected degree models, it is difficult to achieve the probability distribution of the statistic $T_{\mathrm{RM-CNN}}$ for obtaining the theoretical detection threshold. Thus, we apply the Monte Carlo method to set the detection threshold γ of the statistic $T_{\mathrm{RM-CNN}}$ for a given p_{fa}. Specifically, we first collect the dataset under the hypothesis H_0, which can be expressed as

$$\Xi_{B_M} = \{B_M^{(H_0,1)}, B_M^{(H_0,2)}, \cdots, (B_M^{(H_0,L_{H_0})})\}. \tag{25}$$

Then, we feed them into the well-trained RM-CNN and obtain the detection statistic $T_{\mathrm{RM-CNN}}(B_M^{(H_0,l)})$, $l = 1, 2, \cdots, L_{H_0}$. We build the set $\Xi_{T_{\mathrm{RM-CNN}}|H_0}$ in descending order based on $T_{\mathrm{RM-CNN}}(B_M^{(H_0,l)})$, $l = 1, 2, \cdots, L_{H_0}$. To this end, we define the detection threshold γ for a given probability of false alarm α as

$$\gamma = \Xi_{T_{\mathrm{RM-CNN}}|H_0}(\lfloor \alpha L_{H_0} \rfloor). \tag{26}$$

The details of the above procedure are summarized in Algorithm 2.

Remark 3: In this subsection, we propose a RM-CNN algorithm by utilizing the RM-CNN to replace the DNN in Section II. The proposed pre-denoising process greatly reduces the network training time while ensuring the detection performance of the algorithm. Even though the algorithm complexity of the RM-CNN offline training is high, its on-line detection phase has a relatively lower computational complexity than some traditional anomaly subgraph detection algorithms. Therefore, the proposed algorithm can provide the faster detection speed than the typical algorithm. For example, the computational complexity of the sparse principal component analysis method is $O(N^4 \sqrt{\log N}/\varepsilon)$, where ε controls accuracy. The computational complexity of the eigenvector L_1 norms method is $O(|E|kh + Nk^2h + k^3h)$, where k and h denote the number of eigenvector

Algorithm 2 RM-CNN On-Line Detection

1: Input: observed graph $\widetilde{\mathbf{A}}$, expected degree sequence $\mathbf{d}$.
2: Output: Decision about absent or present of anomalous graph G_F.
3: Initialize: Parameter M, threshold γ;
4: Obtain the residual matrix $\widetilde{B}_M$ of $\widetilde{\mathbf{A}}$ by using (14) and (15);
5: Feed $\widetilde{B}_M$ into the trained RM-CNN to obtain the output;
6: Establish detection statistic $T_{\mathrm{RM-CNN}}(\widetilde{B}_M)$ with (24);
7: If $T_{\mathrm{RM-CNN}}(\widetilde{B}_M) > \gamma$, G_F is present; otherwise, G_F is absent.

and the number of restarts [16], respectively. In addition, the computational complexity of the on-line phase is $O(N^2 + \sum_{i=1}^{D} m_i^2 K_i^2 c_{i-1} c_i)$ for the RM-CNN algorithm. It can be observed that the computational complexity of these methods mainly depends on the parameters N and k. Moreover, for large-scale graphs, N and k are commonly greater than 10^3 and 10^2, respectively. As a result, the computational complexity of the RM-CNN in on-line detection phase is relatively lower than the compared algorithms.

V. SIMULATION RESULTS

In this section, simulation experiments are provided to validate the detection performance of the RM-CNN algorithm relative to the Eigenvector L_1 norm method, total degree method, and support vector machine (SVM) method. In addition, the SVM takes the number of triangles and the maximum degree as features [9]. We consider the given expected degree models that are generated by the Kronecker product method [32], and set the average degree of the expected degree sequence as 12. Assume that the subgraph embedded in background graph is a dense Erdős-Rényi random graph. Moreover, the SNR of the anomalous detection problem [17] is given by

$$\mathrm{SNR} = ||\mathbf{A}_F||_2 / ||\mathbf{A} - \mathbb{E}(\mathbf{A})||_2. \tag{27}$$

From the above formula, it is observed that the parameters which affect the SNR mainly include the vertex number N of the random graph, the anomalous vertex number N_F of the subgraph, and the anomalous probability p_f. In what follows, we will examine the performance of the proposed algorithm by changing the values of the above parameters. Without special notes, all results of the simulation experiments are obtained through $\mathrm{M_c} = 400$ Monte-Carlo (MC) realizations, and L_{H_0} is set to 100. For a given probability of false alarm p_{fa}, we determine the corresponding threshold γ based on (26) and calculate the corresponding probability of detection p_d. Specifically, we introduce an indicator function

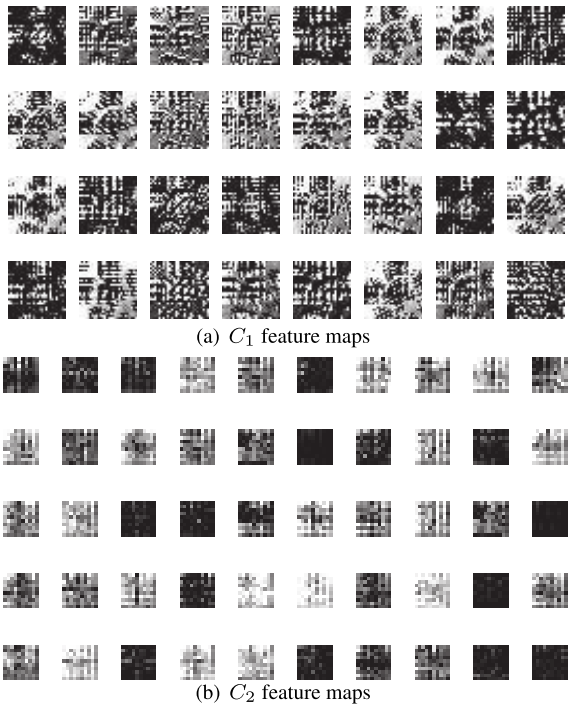
$$\mathbf{1}(T^{(i)}_{\mathrm{RM-CNN}}) = \begin{cases} 1, & \text{if True,} \\ 0, & \text{if False.,} \end{cases}$$

i.e., if the judgment of the ith MC realization is true, then $\mathbf{1}(T^{(i)}_{\mathrm{RM-CNN}}) = 1$, otherwise, $\mathbf{1}(T^{(i)}_{\mathrm{RM-CNN}}) = 0$. So, for a given p_{fa}, we have

$$p_d = \frac{\sum_{i=1}^{\mathrm{M_c}} \mathbf{1}(T^{(i)}_{\mathrm{RM-CNN}})}{\mathrm{M_c}}. \tag{28}$$

A. FEATURE MAPS VISUALIZATION

To better understand the process of extracting graph features of the RM-CNN, this subsection provides the visualization of feature maps. Consider a given expected network with $N = 1024$, and an Erdős-Rényi subgraph with $N_F = 15$ and $p_f = 0.7$ is embedded into the background graph. Fig. 5 (a) shows that after the first convolutional layer, the RM-CNN learns some features of the graph, such as the edge and diagonal features of the random graph. Similarly, as shown in Fig. 5 (b), more graph features are learned by the second convolutional layer. The spatial size (length) of the single feature map by the two-layer convolution operation is 28×28 and 10×10, respectively.



(a) C_1 feature maps

(b) C_2 feature maps

FIGURE 5. Graph features visualization.

B. ALGORITHM PERFORMANCE EVALUATION

In this subsection, we compare the proposed algorithm with the traditional methods from different aspects.

In order to evaluate the effect of the pre-denoising parameter M on the proposed algorithm performance, we conduct the following experiment. For the given expected degree models, we set $N = 1024$, $p_f = 0.5$, $p_{fa} = 0.1$, and $N_F = 15$. From Fig. 6, it is observed that the performance of the algorithm is improved as M increases. For small M scenario,

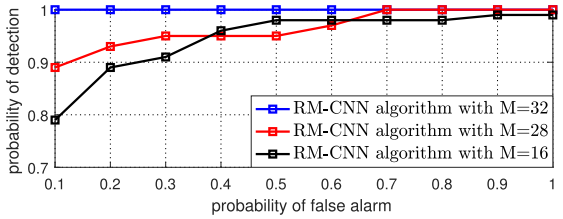


FIGURE 6. ROC curves for different pre-denoising coefficient M.

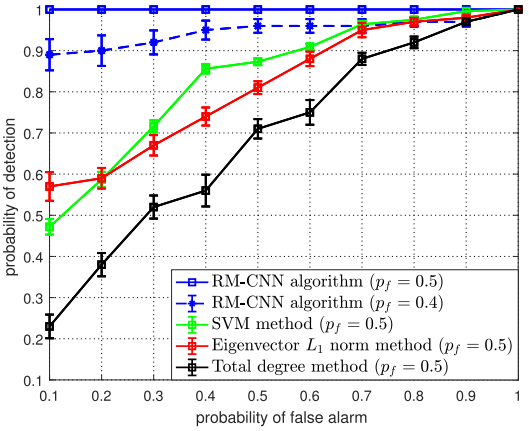


FIGURE 7. ROC curves for the compared methods and the RM-CNN algorithm under the given expected degree models. The error bars represent 0.25 × standard deviation.

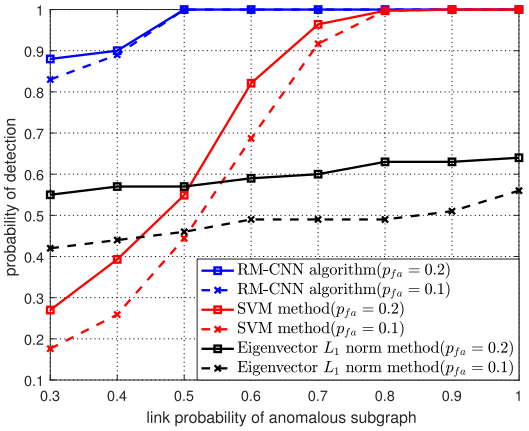


FIGURE 8. The probability of detection versus the link probability of anomalous subgraph for the compared methods and the RM-CNN algorithm under the given expected degree models.

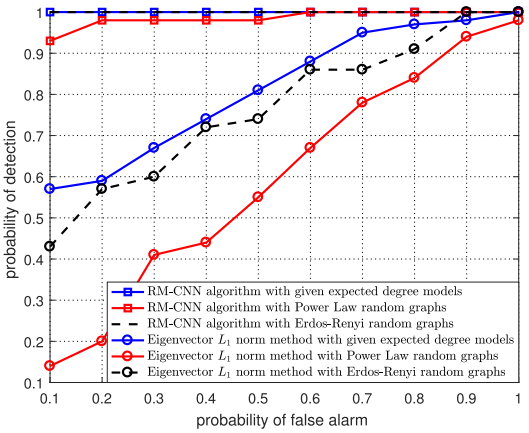


FIGURE 9. ROC curves for the RM-CNN algorithm and the Eigenvector L_1 norms method under different random graphs.

the RM-CNN algorithm may slightly lose some useful graph information, which leads to the performance degradation of the algorithm. To balance the offline training time and the detection performance, we choose $M = 32$ in the following simulations.

In Fig. 7, we consider the given expected degree models with $N = 1024$ and the anomalous subgraph with $N_F = 15$. We provide the ROC curves for $p_f = 0.4$ or $p_f = 0.5$ in Fig. 7. By varying the value of p_{fa}, we choose different thresholds based on the formula (26) to obtain the corresponding p_d. Moreover, in order to confirm the stability of the RM-CNN algorithm, we provide the standard deviation in Fig. 7. The corresponding numerical results are presented in Table 3. From Fig. 7 and Table 3, one can observe that, with the same conditions, the detection performance and the stability of the RM-CNN algorithm outperforms the compared methods.

Next, we show the results of the probability of detection p_d versus the link probability p_f in Fig. 8. Specifically, we set $p_{fa} = 0.1$ and 0.2, and let the anomalous probability p_f vary from 0.3 to 1. The simulation results show that the proposed algorithm has better probability of detection than the compared algorithms. Although the link probability of anomalous subgraph is weak, the RM-CNN algorithm can also capture the anomalous graph features more efficiently, and provide a good detection performance.

In Fig. 9, we consider several random graph models, i.e., the given expected degree, power law random, and Erdős-Rényi random graph models. The network parameters

TABLE 3. The partial numerical results of different methods from Fig. 7.

p_d (standard deviation) \ p_{fa} / Method (p_f)	0.1	0.3	0.5	0.7	0.9
RM-CNN algorithm (0.5)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)
RM-CNN algorithm (0.4)	0.89 (0.075)	0.92 (0.058)	0.96 (0.033)	0.96 (0.033)	0.97 (0.022)
SVM method (0.5)	0.47 (0.114)	0.71 (0.105)	0.87 (0.057)	0.96 (0.041)	0.99 (0.021)
L_1 norm method (0.5)	0.57 (0.139)	0.67 (0.099)	0.81 (0.062)	0.95 (0.069)	0.98 (0.02)
Total degree method (0.5)	0.23 (0.114)	0.52 (0.112)	0.71 (0.094)	0.88 (0.059)	0.97 (0.032)

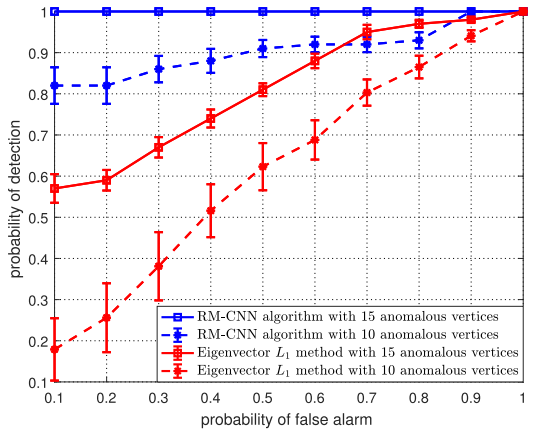


FIGURE 10. ROC curves for the compared methods and the RM-CNN algorithm with two cases (i.e., $N_F = 10$ and $N_F = 15$, respectively). The error bars represent 0.25 × standard deviation.

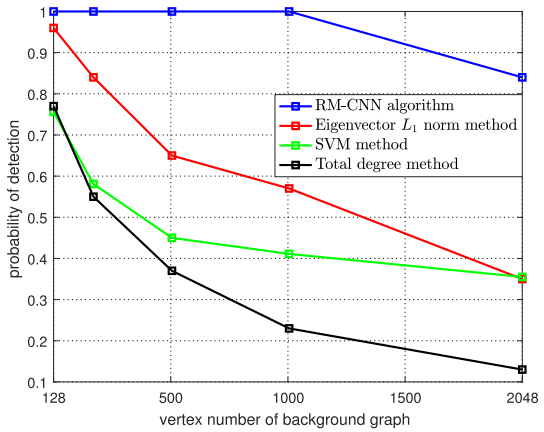


FIGURE 11. The probability of detection versus the vertices number N of the background graph.

are set as $N = 1024$, $N_F = 15$, and $p_f = 0.5$. Fig. 9 depicts the ROC curves of different methods, and we can note that the proposed algorithm is optimal for all scenarios.

Next, we consider the impact of the number of the anomalous vertex N_F and total vertex N on the algorithm performance in regard to the given expected degree models. We first show the robustness of our algorithm in Fig. 10 by fixing $N = 1024$, $p_f = 0.5$, $p_{fa} = 0.1$, and varying the value of N_F. From Fig. 10, it is observed that the detection performance improves as the value of N_F increases for detectors. In addition, we can note that the RM-CNN algorithm has excellent detection performance even when the number of anomalous vertices is small. Then, let us consider the effect of the value of N on the algorithm performance. We set $N_F = 15$ and vary N from 128 to 2048, and then generate a series of random graphs regarding the given expected degree models with the same average degree. In addition, the other parameters are the same as those for Fig. 10. As depicted in Fig. 11, the detection performance of the algorithm significantly decreases as N increases. The main reason for the above phenomenon is that since a higher N_F (N) means a higher (lower) SNR, the detector can be easier (more tough) to discover the anomaly. The results provided in Fig. 10 and 11 display that the RM-CNN algorithm is more sensitive to the anomalous behaviors than the comparison methods.

Finally, we take the same parameter set in Fig. 7 and evaluate the detection time of different algorithms. We perform the experiments in MATLAB (R2016b) environment and run on a laptop with Intel Core i5-8250U CPU @ 1.60GHz and a NVIDIA GeForce MX150 GPU. The average detection time operated by the Eigenvector L_1 norm method and the Total degree method are 0.774 and 0.616 seconds, respectively. However, the average detection time of the RM-CNN algorithm is only 0.406 seconds. In spite of the proposed algorithm requires plentiful offline training time, its detection speed is significantly faster than the other algorithms. Moreover, as the dimension of the random graph increases, the advantage of the algorithm regarding detection speed will become more obvious.

VI. CONCLUSION

In this paper, we have analyzed the anomalous subgraph detection problem under complex random graph models and proposed a DNN-based subgraph detection framework. As a specific application of the above framework, we have presented a RM-CNN algorithm regarding given expected degree models. Due to the complex structure of the given degree models, the traditional algorithms may not provide good detection performance. Moreover, the dimension of the anomalous subgraph embedded in the background graph is small, which makes the problem more challenging. Based on the proposed DNN-based subgraph detection framework, the RM-CNN algorithm can efficiently extract the nonlinear graph features with respect to the anomalous subgraph. As a result, the performance of the proposed algorithm is superior to the compared algorithms in the terms of the detection performance and the detection speed of on-line phase.

REFERENCES

[1] T. Sui, X. Tao, S. Xia, H. Chen, H. Wu, X. Zhang, and K. Chen, "A real-time hidden anomaly detection of correlated data in wireless networks," *IEEE Access*, vol. 8, pp. 60990–60999, Mar. 2020.

[2] Y. Zhao, B. Wang, M. Luan, and F. Hu, "Local statistic with dynamic vertex selection for change-point detection in stochastic block networks," *IEEE Access*, vol. 8, pp. 127954–127967, Jul. 2020.

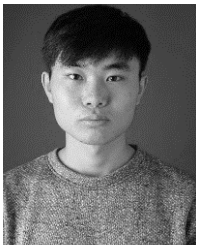
[3] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin, and V.-L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection," *IEEE Access*, vol. 8, pp. 30387–30399, Feb. 2020.

[4] B. Wang, F. Hu, Y. Zhao, and T. N. Guo, "Anomaly detection and array diagnosis in wireless networks with multiple antennas: Framework, challenges and tools," *IEEE Netw.*, vol. 32, no. 1, pp. 152–159, Jan. 2018.

[5] B. A. Miller, N. T. Bliss, P. J. Wolfe, and M. S. Beard, "Detection theory for graphs," *Lincoln Lab. J.*, vol. 20, no. 1, pp. 10–30, 2013.

[6] N. Arcolano, K. Ni, B. A. Miller, N. T. Bliss, and P. J. Wolfe, "Moments of parameter estimates for Chung-Lu random graph models," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2012, pp. 3961–3964.

[7] H. Wang, M. Tang, Y. Park, and C. E. Priebe, "Locality statistics for anomaly detection in time series of graphs," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 703–717, Feb. 2014.
[8] W. Aiello, F. Chung, and L. Lu, "A random graph model for power law graphs," *Exp. Math.*, vol. 10, no. 1, pp. 53–66, Jan. 2001.
[9] Y. Park, C. E. Priebe, and A. Youssef, "Anomaly detection in time series of graphs using fusion of graph invariants," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 1, pp. 67–75, Feb. 2013.
[10] A. Rukhin, "Asymptotic analysis of various statistics for random graph inference," Ph.D. dissertation, Dept. Appl. Math Statist., Johns Hopkins Univ., Baltimore, MD, USA, 2009.
[11] F. Chen and D. B. Neill, "Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 1166–1175.
[12] E. Arias-Castro and N. Verzelen, "Community detection in random networks," 2013, *arXiv:1302.7099*. [Online]. Available: http://arxiv.org/abs/1302.7099
[13] B. Wang, Y. Zhao, F. Hu, and Y.-C. Liang, "Anomaly detection with subgraph search and vertex classification preprocessing in chung-lu random networks," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5255–5268, Oct. 2018.
[14] A. Kadavankandy, K. Avrachenkov, L. Cottatellucci, and R. Sundaresan, "The power of side-information in subgraph detection," *IEEE Trans. Signal Process.*, vol. 66, no. 7, pp. 1905–1919, Apr. 2018.
[15] F. Sheikholeslami and G. B. Giannakis, "Identification of overlapping communities via constrained egonet tensor decomposition," *IEEE Trans. Signal Process.*, vol. 66, no. 21, pp. 5730–5745, Nov. 2018.
[16] B. A. Miller, N. T. Bliss, and P. J. Wolfe, "Subgraph detection using eigenvector L_1 norms," in *Proc. Adv. Neural Inf. Process Syst.*, 2010, pp. 1633–1641.
[17] B. A. Miller, M. S. Beard, P. J. Wolfe, and N. T. Bliss, "A spectral framework for anomalous subgraph detection," *IEEE Trans. Signal Process.*, vol. 63, no. 16, pp. 4191–4206, Aug. 2015.
[18] B. A. Miller, N. T. Bliss, and P. J. Wolfe, "Toward signal processing theory for graphs and non-Euclidean data," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2010, pp. 5414–5417.
[19] N. Singh, B. A. Miller, N. T. Bliss, and P. J. Wolfe, "Anomalous subgraph detection via sparse principal component analysis," in *Proc. IEEE Stat. Signal Process. Workshop (SSP)*, Jun. 2011, pp. 485–488.
[20] F. Chung and L. Lu, "Connected components in random graphs with given expected degree sequences," *Ann. Combinatorics*, vol. 6, no. 2, pp. 125–145, Nov. 2002.
[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[22] Q. Wang, P. Du, J. Yang, G. Wang, J. Lei, and C. Hou, "Transferred deep learning based waveform recognition for cognitive passive radar," *Signal Process.*, vol. 155, pp. 259–267, Feb. 2019.
[23] D. A. Pados and P. Papantoni-Kazakos, "New nonleast-squares neural network learning algorithms for hypothesis testing," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 596–609, May 1995.
[24] X. Li, W. Zhang, and Q. Ding, "Understanding and improving deep learning-based rolling bearing fault diagnosis with attention mechanism," *Signal Process.*, vol. 161, pp. 136–154, Aug. 2019.
[25] J. Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang, "Device-free wireless localization and activity recognition: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6258–6267, Jul. 2017.
[26] Q. Wang, J. Wan, and X. Li, "Robust hierarchical deep learning for vehicular management," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4148–4156, May 2019.
[27] L. Akoglu, M. Mcglohon, and C. Faloutsos, "Oddball: Spotting anomalies in weighted graphs," in *Proc. Adv. Knowl. Discovery Data Mining*, 2010, pp. 410–421.
[28] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.
[29] S. M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 1998.
[30] W. Lee, M. Kim, and D.-H. Cho, "Deep cooperative sensing: Cooperative spectrum sensing based on convolutional neural networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3005–3009, Mar. 2019.
[31] P. Planinsic and D. Gleich, "Temporal change detection in SAR images using log cumulants and stacked autoencoder," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 2, pp. 297–301, Feb. 2018.
[32] D. Chakrabarti, Y. Zhan, and C. Faloutsos, "R-MAT: A recursive model for graph mining," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2004, pp. 442–446.



MINGAN LUAN (Graduate Student Member, IEEE) received the B.S. degree in electronics and information engineering from the Changchun Institute of Technology, China, in 2018. He is currently pursuing the Ph.D. degree with the College of Communication and Engineering, Jilin University, China. His research interests include signal detection and localization, machine learning, and graph signal processing.

BO WANG received the Ph.D. degree in communication and information systems from Jilin University, Changchun, China, in 2006. From July 2014 to July 2015, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, Tennessee Technological University (TTU), Cookeville, TN, USA. He is currently a Professor with the College of Communication and Engineering, Jilin University. His research interests include graph signal processing, array signal processing, signal detection, and source localization.

YANPING ZHAO received the Ph.D. degree in communication and information systems from Jilin University, China, in 2014. She is currently a Lecturer with the College of Communication Engineering, Jilin University. Her research interests include speech signal processing and sparse signal processing.

FENGYE HU (Senior Member, IEEE) received the B.S. degree from the Department of Precision Instrument, Xi'an University of Technology, Xi'an, China, in 1996, and the M.S. and Ph.D. degrees in communication and information systems from Jilin University, Changchun, China, in 2000 and 2007, respectively. In 2011, he was a Visiting Scholar in electrical and electronic engineering from Nanyang Technological University, Singapore. He is currently a Full Professor with the College of Communication Engineering, Jilin University. He has authored or coauthored 50 publications in the IEEE journals and conferences. His current research interests include wireless body area networks, wireless energy and information transfer, energy harvesting, cognitive radio, and space–time communication. He was an Executive Co-Chair of the IEEE/CIC International Conference on Communications in China (ICCC), China, in 2019. He is also an Editor of *IET Communications*, *China Communications*, and *Physical Communication* Special Issue on Ultra-Reliable, Low-Latency and Low-Power Transmissions in the Era of Internet-of-Things. He organized the first and second Asia-Pacific Workshop on Wireless Networking and Communications (APWNC 2013 and APWNC 2015) and the Future 5G Forum on Wireless Communications and Networking Big Data (FWCN 2016).