IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 June 2022, accepted 6 July 2022, date of publication 11 July 2022, date of current version 15 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3189790

RESEARCH ARTICLE

Graph Convolutional Networks and Attention-Based Outlier Detection

RUI QIU[1], XUSHENG DU[2], JIONG YU[2], JIAYING WU[3], AND SHU LI[1]
[1]School of Software, Xinjiang University, Ürümqi 830046, China
[2]School of Information Science and Engineering, Xinjiang University, Ürümqi 830046, China
[3]Unit 32317 of PLA, Ürümqi 830000, China

Corresponding author: Xusheng Du (duxusheng@stu.xju.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61862060, Grant 61462079, Grant 61562086, and Grant 61562078.

ABSTRACT Outlier detection is a significant research direction in machine learning and has many applications in finance, network security, and other areas. Outlier detection of Euclidean datasets is a mainstream problem in outlier detection. Most detection methods often ignore the connection of its nodes. To collect the representation information of feature sets and node connections to improve the detection of outliers in Euclidean datasets Accuracy rate, we propose a novel Graph Convolutional and Attention-Based Outlier Detection (GCA).The GCA first converts the Euclidean structure data into directed graphs using locally sensitive hashing; then, by applying a Graph Convolutional Network, the data features and their connectivity graph are fed into the neural network; secondly, it fuses the extracted features and the features reconstructed by the attention mechanism; finally, calculating the outlier factors of the objects. Comparing eight state-of-art algorithms on ten real-world datasets shows that GCA achieves the highest Area Under ROC Curve (AUC) on datasets and also achieves equally good results in Accuracy (ACC) and False Alarm Rate (FAR). This study fills the gap of upgraded GCNs in detecting outliers to the best of our knowledge and provides a new way to convert Euclidean data to graphs.

INDEX TERMS Outlier detection, graph convolutional network, directed graph, attention mechanism, feature fusion.

I. INTRODUCTION

Outlier detection, one of the fundamental tasks of data mining, has become an active branch of information science after a long research history and has received wide attention in the fields of database, data mining, machine learning, and statistics. The definition of outlier is generally considered: if a point is relatively less dense on its own than the high-density pattern clusters in its vicinity or if its density is relatively higher than the low-density pattern regularity in its vicinity [1]. Outlier detection covers a wide variety of dataset applications, including Euclidean datasets for text, audio, etc., and non-Euclidean datasets for graphs such as social networks and transportation networks. The significance lies not only in the more accurate perception of ordinary objects but also in the enormous amount of information and mining value it contains in itself. Outlier detection has an extensive range of applications in many fields: bank fraud [2]–[4], video surveillance [5]–[10], network anomalies [11]–[14], finding new celestial objects [15], [16], etc. The available outlier detection algorithms can be broadly classified into: distance-based algorithms [17]–[20], density-based algorithms [21]–[23], clustering-based algorithms [24], [25], statistical methods [26], integration-based methods [27], numerous neural network-based algorithms [28] and graph-based algorithms [29] etc.

Most graph algorithms only work on graph-structured data, and the same goes for GCN [30]. However, the node representation of GCN requires both node information and adjacency information, which allows upgrading GCN to process Euclidean data. Previous outlier detection algorithms for Euclidean datasets represented by samples and dimensions tend to focus on independent features, neglecting the connection between points, and graph-based algorithms and neural

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

network-based outlier detection algorithms work separately. For GCN, considering the relationship of the graph space, that is, the probability conduction matrix, it can be applied to the Euclidean dataset, but the neighbor weights are not dynamically learned. GAT can dynamically learn neighbor weights, but it ignores the existence of nodes themselves and cannot be applied to Euclidean datasets. From this point of view, we propose Graph Convolutional and Attention–Based Outlier Detection (GCA).In this paper, we adopt the aggregation model of two feature extraction channels. The first channel generates the adjacency matrix for the feature dataset by Locally Sensitive Hash (LSH) algorithm and trains the adjacency matrix by Graph Convolutional Network (GCN). The second channel uses the attention mechanism to reconstruct the feature dataset by assigning weights. The feature matrices output from the two channels are merged into a single feature set using typical correlation analysis feature fusion, and finally, outliers are determined using the Local Outlier Factor (LOF) algorithm.

Our work has the following advantages and contributions:

1) We propose a new outlier detection method based on GCN and Attention dual-channel feature fusion reconstruction methods. To the best of our knowledge, the GCA algorithm is the first application of the GCN fusion algorithm to the outlier detection problem.
2) We use the LSH algorithm to convert the point-to-point relationship into a graph so that Euclidean datasets can be trained in graph neural networks.
3) We upgraded the traditional GCN from operating only on undirected graphs to being able to operate on directed graphs to retain more information.
4) Experimental results from ten real datasets, including those from the medical industry, demonstrate that the GCA algorithm is comparable to the current mainstream outlier point detection algorithms.

II. RELATED WORK

The outlier application scenarios are very diverse, not only in the fields of network security and finance, but also in the fields of public health and astronomy. Long-established and mainstream outlier detection methods for datasets can be classified as proximity-based methods, statistical methods, clustering-based methods, integration-based methods, and there are also numerous neural network-based algorithms and graph-based algorithms.

The core idea of proximity-based methods is to define a proximity metric between data and determine outliers based on the value of this metric. The typical methods are distance-based and density-based. The principle is that the proximity of an outlier object to its nearest neighbors significantly deviates from the proximity of other regular objects in the dataset to their nearest neighbors. Specific samples with similar characteristic attributes can be considered similar in their target attributes. The former reflects proximity in terms of distance, and points far away from most of their neighbors or do not have enough neighbors are most likely to be outliers. The latter reflects proximity in terms of density, and it is generally believed that outliers generally exist in low-density areas and non-outliers appear in dense areas. The most commonly used distance-based outlier detection and identification method is the k-nearest neighbor (KNN) and its extended algorithm [30] that focuses on the concept of the local neighborhood, where the k-value is the nearest K neighbors. The paper discards the previous practice of assigning fixed k-values to all test samples and provides training in the classification process of KNN to learn k-values for samples with different k values, making the algorithm run at a similar cost to the traditional KNN algorithm but with improved accuracy. Algorithms based on nested loops using randomization and pruning rules [31], [32] provide nearly linear time performance on most data sets.

Distance-based methods also include solution set methods [33], based on the main idea of using a solution set to solve the outlier prediction and detection problems. Three solution algorithms are proposed to compute the solution set: the solution set algorithm, the robust solution set algorithm, and the mini-robust solution set algorithm. The classical density-based outlier detection methods are the Local Outlier Factor (LOF) [34] algorithm, which defines a local outlier factor for each object by comparing the density of each point with its neighboring points, and the determination for outliers is transformed into the determination of the outlier factor. In addition to the LOF algorithm, The INFLuenced Outlierness degree algorithm (INFLO) [35] adds the influence of neighbors and reverse neighbors to the density of LOF and becomes a classical algorithm in outlier detection based on symmetric neighborhood relations. The connectivity-based outlier (COF) [23] factor algorithm is similar to that of LOF. The contamination parameter can specify the proportion of outliers in the data.

It is arguably the group of statisticians who first discovered the existence of outliers, as outliers can be easily identified in the statistical data process. Thus statistically based methods were developed early and have various branches. Statistically based methods assume that ordinary objects in a dataset are generated by a stochastic process that can be viewed as a generative model, where objects in the model's high probability region are considered normal and vice versa. The fitting of the generative model is generally divided into parametric and nonparametric methods, and the classical literature [26] in the parametric method has proposed more than 100 methods such as one-dimensional Gaussian distribution and mixed Gaussian distribution models. Box line plot is a simple method to represent the five-number summary, which includes five values, maximum, minimum, and three quartiles, and is a standard algorithm based on statistics. Laurikkala *et al.* [36] used a box line plot to identify multivariate outliers directly. Typical methods in nonparametric methods Histogram visualization methods have been used in the field of intrusion detection [37] for a long time, also using histogram methods is the Histogram Based Outlier Score [38] (HBOS) algorithm

for numerical elements uses two types of histograms: static box-width histograms and dynamic box-width histograms.

Most of the clustering methods determine outliers by the relationship between data objects and clusters, and representative methods include the ODC (Outlier Detection and Clustering) [39] algorithm proposed by Ahmed and Mahmood, and the KMOR (K-means with outlier removol) [40] algorithm proposed by Gan and Ng both algorithms perform clustering while detecting outliers. The former uses a new unsupervised method, i.e., the improved k-means algorithm, to detect and remove outliers and then perform clustering, while the latter introduces an iterative process to optimize the objective function based on extended k-means, and all outliers are saved in the clusters.

The most classic and currently popular ensemble-based method is the Isolation Forest [27], proposed by Liu *et al.*, widely used in large high-dimensional datasets. The main principle is that anomalous samples can be isolated by less random feature segmentation than ordinary samples, and the method explicitly isolates anomalies instead of profiling normal points. Feature bagging [41] is similar to bagging in basic idea, except that the object is the feature, and the final result is obtained by fraction normalization and combination method after selecting the base detector. Extreme Boosting Based Outlier Detection (XGBOD) [42] and Locally Selective Combination in Parallel Outlier Ensembles (LSCP) [43] also belong to the category of integration methods.

Network-based methods have become a hot research topic in recent years. AutoEncoder [44] various extensions [45] are used for outlier detection. AE is a model with an automatic coding function composed of coding combined with the neural network, which learns the representation of the input information by using it as a learning target. Single-Objective Generative Adversarial Active Learning (SO-GAAL) and Multiple-Objective Generative Adversarial Active Learning (MO-GAAL) [46] proposed by Liu *et al.* Using the idea of Generative Adversarial Networks (GAN), Generator is used to generate potential outliers (anomalous data), combine noise and real data, let Discriminator discriminate between noise and real data, and finally use Discriminator as an anomaly detection classifier.

The outlier application scenarios currently, most of the actual graph-based outlier detection is to find outliers in the graph data. The primary outlier detection based on the graph structure is Outrank [47], which constructs a fully connected undirected graph and applies a Markov random walk process on the graph. The smooth distribution of the random walk is directly used as the outlier score. The Random Walk [48] model combines a graphical representation with local information around each object to construct a local information graph and calculates the outlier score by performing a random walk process on the graph. A cut-point clustering algorithm (CutPC) based on a natural neighbor graph is proposed [29]. The CutPC method performs noise cutting when a cut-point value is above the critical value. This algorithm can also be used for outlier detection

A comparison between the popular outlier detection methods and the method proposed in this paper is made in 1.

III. PROPOSED METHOD

The core idea of GCA for outlier detection is: first, use the GCN to learn the feature embedding of the objects in dataset X with their neighbors. Among them, the connectivity graph is constructed through LSH; secondly, the attention mechanism is used to assign different weights to the features of the objects in dataset X, and reconstruction is based on the magnitude of feature weights between objects; finally, the feature matrix output by the GCN and the feature matrix output by the attention mechanism are fused together, and the LOF algorithm is used to detect them and the final outlier factor are calculated for each object. Statistical-based anomaly detection algorithms usually assume that the data obey a specific probability distribution, an assumption that is often not valid. Moreover, clustering methods usually only give a 0/1 judgment and cannot quantify the degree of an anomaly for each data point. In comparison, the density-based LOF algorithm is more straightforward and more intuitive. It does not require much about the data distribution and can quantify the degree of the anomaly of each data point. However, for independent LOF, the original feature dataset is not evident in the case of the low efficiency of LOF algorithm detection; after feature extraction and rearrangement, the detection rate can be improved. The entire architecture of the model is represented in Fig.1.

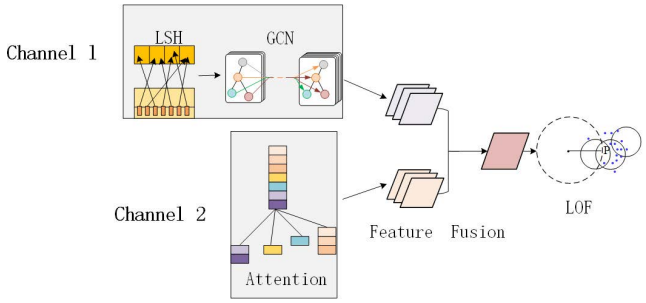

FIGURE 1. The entire structure of GCA for outlier detection, channel 1 is input to GCN for training by creating a connection map through LSH, channel 2 is trained by the Attention mechanism, and the result is input to LOF by feature fusion.

A. CONSTRUCT GRAPH

The graph construction application Locality Sensitive Hashing (LSH), which relies on a hash function, maps the points aggregated in a specific range into the same hash bucket, and the high-dimensional data points in different ranges are mapped into different hash buckets. In the query, the other points in the hash bucket where the queried point is located are the potential neighbors of the queried point. We connect the queried points to their potential neighbors to increase the relevance of the points in the Euclidean dataset and produce

TABLE 1. Description of the typical test method and its advantages and disadvantages.

Detection Model	Category	Description	Advantages	Disadvantages
AutoEncoder(AE)	Network-based	AE uses a backpropagation algorithm to make the output value equal to the input value of the artificial neural network, which first compresses the input into a potential spatial representation and then reconstructs the output through this representation.	High generalizability, unsupervised without data annotation.	For anomaly recognition scenarios, the training data needs to be standard.
Connectivity outlier factor algorithm (COF)	Density - based	A point p is an anomaly if the average connected distance of the point p is greater than the average related distance of its k nearest neighbors.	Data adaptation to different types.	Defining the distance between data can sometimes be difficult.
Cut-point clustering (CutPC)	Graph - based	Using cluster pruning technique as a method to reduce the computational effort.	Highly effective.	The quality of the generated clusters has a powerful influence on the quality of the outlier points produced by this algorithm.
Isolation Forest(IForest)	Integration-based	Divide the entire data into multiple subsets and perform anomaly detection operations on the subsets.	Applicable to continuous data and unsupervised learning	A higher sample size reduces the ability of isolated forests to isolate outliers.
K-Nearest Neighbor (KNN)	Distance -based	When doing classification forecasting, the majority voting method is generally chosen, and when doing regression, the average method is usually chosen.	Both classification and regression problems can be handled	Particularly dependent on training data and prone to dimensional disasters.
Outrank	Graph - based	The smooth distribution of the random walk is directly used as the outlier score.	Low time complexity.	Neglected topic relevance.
SO-GAAL	Neural Network-based	SO-GAAL directly generates informative potential outliers to assist the classifier in describing a boundary that can separate outliers from normal data effectively.	The algorithm can directly generate potentially informative outliers.	Its runtime has no advantage for small datasets.
MO-GAAL	Neural Network-based	Extend the structure of GAAL from a single generator to multiple generators with different goals.	The algorithm can prevent the generator from falling into the mode collapse problem.	Computational requirements increase linearly with the amount of data.
Graph Convolutional and Attention (GCA)	Neural Network-based	Computing outliers using LOF after double feature enhancement with GCN and attention.	Combining the advantages of graph-based and neural network-based.	Higher time complexity.

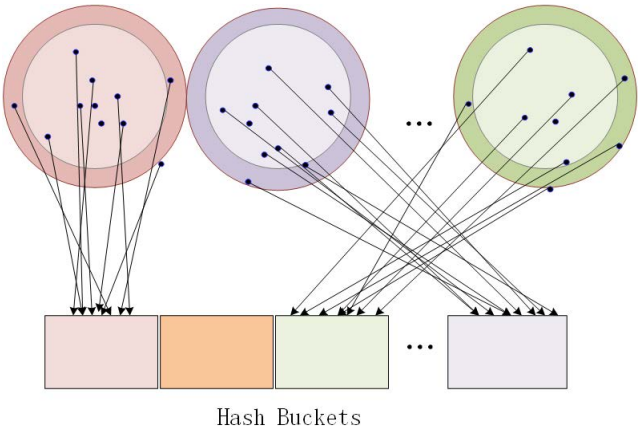

FIGURE 2. Data points are mapped to different hash buckets. The similarity of the data points in the original space is positively correlated with the probability of being in the same bucket after mapping.

a new connectivity graph. The mapping principle of the LSH midpoint is shown in Fig.2.

The projection of the query uses the L1 distance as (1), Hash presents a family of hash functions:

$$\text{Hash} = \left\lfloor \frac{(X * a - b)}{w} \right\rfloor \tag{1}$$

X is the input dataset, a and b are random tuple arrays, and w is the width of the hash bucket, which ensures that points in the same quantized segment can be mapped to the same hash bucket. The function $Hash$ is (R, εR, P_1, P_2) sensitive, and any two points p and q in R_d space satisfy the following two properties as (2) and (3):

$$If\ |p-q| \le R\ then\ Pr\left(Hash\left(p\right) = Hash\left(q\right)\right) \ge P_1 \tag{2}$$

$$If\ |p-q| \ge \varepsilon R\ then\ Pr\left(Hash\left(p\right) = Hash\left(q\right)\right) \le P_2 \tag{3}$$

m hash functions are constructed as a hash table HashT by (4):

$$HashT = \langle Hash_1, Hash_2, \ldots, Hashm \rangle \tag{4}$$

The probability of mapping points of the same quantized segment to the same hash bucket is further enhanced by constructing 1 hash tables as (5):

$$HashBucket = \langle HashT_1, HashT_2, \ldots, HashT_l \rangle \tag{5}$$

Given the k, the nearest neighbors of all data points are retrieved, and the nearest neighbors are concatenated. The matrix A is defined as the weightless matrix of data points pointing to the k neighboring data points, X denotes the feature matrix of all nodes, and X_i denotes the feature of the i_{th} node.

Spectral methods generally use Laplacian matrices to represent the structure of graphs, and in this paper, we use

transition probability matrices to define the Laplacian of a graph as a Hermitian matrix, and then we can perform graph convolution operations on directed graphs as (6):

$$L = I - \left(\frac{\phi^{\frac{1}{2}} \times P \times \phi^{-\frac{1}{2}} + \phi^{-\frac{1}{2}} \times P^T \times \phi^{\frac{1}{2}}}{2} \right) \tag{6}$$

where I is the unit matrix, P^T denotes the conjugate transpose of the transfer probability matrix P of the directed graph D, and the φ matrix denotes the matrix whose diagonal is the Perron vector of G and is zero elsewhere. The following algorithm describes querying and joining after the points are hash mapped.

Algorithm 1 Constructing Graph

Input: Given dataset $\boldsymbol{X}$, the number of hashtable $\boldsymbol{m}$, the number of connections $\boldsymbol{k}$, Full zeros matrix $\boldsymbol{A}$
Output: Laplacian matrices $\boldsymbol{L}$
1. **for** iteration=1:m
2. **for** each row in X as k do
3. put k into Hash(k);
4. **end**
5. **end**
6. **for** iteration=1:k
7. L $\longleftarrow$ L $\cup$ {p|search(p) = search(q)}
8. **end**
9. **for** i=1:length of X
10. **for** j=1:k
11. $A_{\text{ji}} = 1$;
12. **end**
13. **end**
14. L =Laplace(A);
15. **return** L

B. ENHANCE FEATURE

1) FEATURE ENHANCEMENT USING GRAPH CONVOLUTION NETWORKS

GCN learns an embedding separately for each node. The embedding of a node is acquired by embedding its neighboring nodes, which means that the graph convolution operator needs to propagate the embedding using the interaction between nodes in the graph. This process is progressively performed layer by layer through the graph convolution operation, after which the embedding of all nodes is linearly transformed, and the output layer, i.e., the last layer of the embedding, is used as the input for the downstream task. Our forward model then takes the simple form (7):

$$H\ (l+1) = \sigma\left(LaH^l\omega^l\right) \tag{7}$$

$H(l+1)$ and $H(l)$ denote the input and output of the l-th layer $(H(0)=X)$, respectively. W^l denotes the weight. LaH^l denotes the weighted average of all nodes. Since the strength of the relationship between each nodes and its neighbors is different, the weights between nodes should not be limited to {0,1} but any suitable weight value. σ is the activation function, in this paper, Leaky_ReLU is used as the activation function in the hidden layer; compared with the Relu activation function used in most algorithms, the input value less than 0 can also be updated with parameters without causing the death of neurons.

The graph convolution layer aggregates neighboring nodes to achieve transfer of neighborhood relations and propagates between layers, using the information of edges to aggregate node information, thus generating a new node representation. Fig.3 represents the convolution process in a two-layer GCN network:

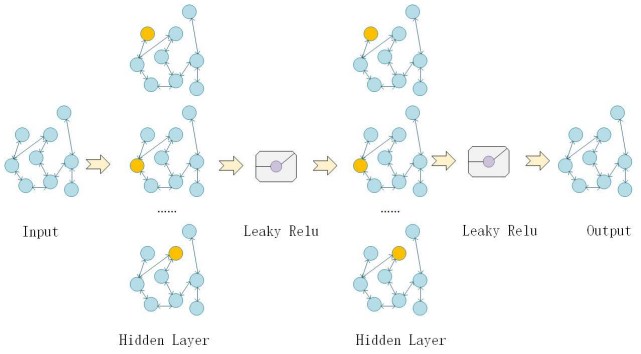

FIGURE 3. Graph neural network model, subgraph extraction is performed after node pre sampling, and graph neural network is generated and trained after subgraph feature fusion.

In multilayer GCN, the Laplacian matrix L is fixed, and it depends on the construction of the topological map. The only parameter to be learned is the parameter ω^l of the transformation matrix, which can be learned and updated by backpropagation. The cross-entropy loss function in (8) can maintain the linear transfer gradient and effectively prevent the gradient from vanishing.

$$Loss = -\left[\mathrm{y}\ log\ y' + (1-y)\ log\ \left(1-y'\right)\right] \tag{8}$$

where y denotes the distribution of the actual sample and y' denotes the distribution predicted by the model. Algorithm 2 represents the construction process in a two-layer GCN network.

Algorithm 2 Training GCN

Input: Given dataset $\boldsymbol{X}$, Laplacian matrix $\boldsymbol{L}$, Learning rate η, Number of iterations $\boldsymbol{t}$
Output: Matrix $\boldsymbol{O}$
1. Initialize $W^{(0)}$, $b^{(0)}$, $W^{(1)}$, $b^{(1)}$.
2. **for** iteration=1:t
3. Layer_1_output= $X * L * W^{(0)} - b^{(0)}$;
4. Layer_2_output= $X*$ Layer_1_output $*W^{(1)} - b^{(1)}$;
4. Loss = Cross-entropy;
5. Update W and b using batch gradient decent $\bigtriangledown\ w(Loss)$;
6. **end**
7. **return** O

2) FEATURE ENHANCEMENT USING ATTENTION MECHANISM

We use the attention mechanism to emphasize the differences between features by assigning different attention weights according to their importance. The input is the original feature matrix, and the output is no longer context vectors but a reassigned feature matrix. Fig.4 expresses the principle of the Attention mechanism:

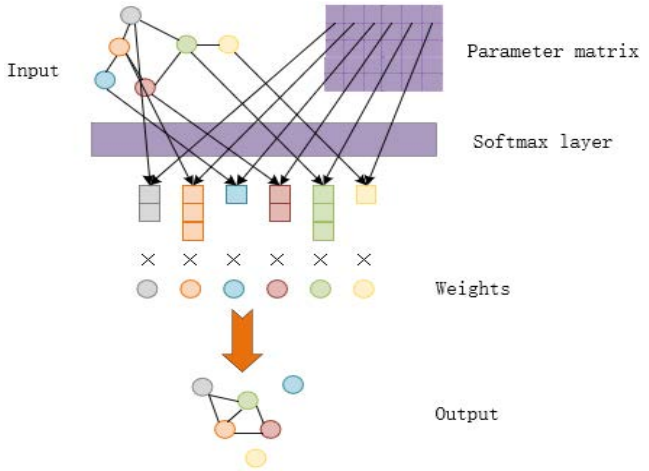

FIGURE 4. The attention mechanism enhances features with weight redistribution. The model scores the input dimensions and then weights the features according to the scores to highlight the impact of essential features on downstream models or modules.

The reassigned feature matrix F is derived from the following (10):

$$F = X \bullet w_{ij} \tag{9}$$

w_{ij} represents the attention weight of a single element generated by a small neural network in the attention layer, starting from the dense layer with a single neuron [49] This layer outputs a score for each output: this score is used to measure how well each output is aligned with the previous hidden state. Finally, all scores are passed through the softmax layer to obtain the final weights of the outputs. The weights are calculated in (10):

$$W_{ij} = \frac{exp\left(v_a^T tanh\left(\psi_1 X_i + \psi_2 \overline{X}j\right)\right)}{\sum_{j=1} exp\left(v_a^T tanh\left(\psi_1 X_i + \psi_2 \overline{X}_{j'}\right)\right)} \tag{10}$$

v, $\psi 1$, $\psi 2$ are the parameter matrices initialized by Glorot uniform initialization, generating random weights and biases by sampling from a uniform distribution function. We use the Glorot initialization formula to maintain the activation variance and back-propagate the gradient variance as the network moves up or down [50]. This initialization is shown in (11).

$$\psi \sim U\left[-\frac{\sqrt{6}}{\sqrt{w_j + w_{j+1}}}, \frac{\sqrt{6}}{\sqrt{w_j + w_{j+1}}}\right] \tag{11}$$

where U represents the consistent distribution of the interval and a_j represents the columns of the weight matrix.The construction of Attention is given in Algorithm 3:

Algorithm 3 Attention Mechanism

Input: Given dataset $\boldsymbol{X}$
Output: Matrix $\boldsymbol{A}$
1.Initialize: $Hidden$,$W^{(1)}$, $b^{(1)}$, $W^{(2)}$, $b^{(2)}$, $W^{(V)}$, $b^{(V)}$
2.Layer1=Fully Connect(X, $W^{(1)}$, $b^{(1)}$);
3.Layer2=Fully Connect($Hidden$, $W^{(1)}$, $b^{(1)}$);
4.Scores=Fully Connect(tanh(Layer1+ Layer2));
5.AttentionWeights=Softmax(Fully Connect(Scores, $W^{(V)}$,$b^{(V)}$));
6. A = attentionWeights.*X;
7. return A.

3) FEATURE FUSION

We fuse the feature matrix O from GCN output and the feature matrix A from attention output to generate a single feature matrix. It is more discriminative than any of the input feature matrices, which is achieved by using the feature fusion technique of CCA [51]. Typical correlation analysis is widely used to uncover correlations between data. The covariance matrix S represents all the information associated with the overall feature pairs obtained by (12).

$$S = \begin{pmatrix} cov\,(O) & cov\,(O, A) \\ cov\,(A) & cov\,(O) \end{pmatrix} = \begin{pmatrix} S_{OO} & S_{OA} \\ S_{AO} & S_{AA} \end{pmatrix} \tag{12}$$

S_{OO} denotes the intra-group covariance matrix of the GCN output matrix, S_{AA} denotes the intra-group covariance matrix of the attention module output matrix, and S_{AO} and S_{OA} represent the inter-group covariance matrix of the GCN output matrix to the attention module output matrix and the attention module output matrix to the GCN output matrix, respectively.

To further strengthen the correlations, CCA uses Lagrange multipliers to maximize the pairwise correlations as (13):

$$corr\left(O^*, A^*\right) = \frac{cov\,(O^*, A^*)}{var\,(O^*)\,var\,(A^*)} \tag{13}$$

The intergroup covariance between the two is maximized at the constraint var (O^*) = var (A^*) = 1. Afterward, the transformed matrices W'_O and W'_A are obtained by solving the eigenvalue equation (14):

$$\begin{cases} S_{OO}^{-1} S_{OA} S_{AA}^{-1} S_{AO} W'_O = \Lambda^2 W'_O \\ S_{AA}^{-1} S_{AO} S_{OO}^{-1} S_{OO} W'_A = \Lambda^2 W'_A \end{cases} \tag{14}$$

Λ^2 is the diagonal matrix of eigenvalues, and W'_O and W'_A are the sorted W_O and W_A corresponding to the non-zero eigenvalues each forming the transformed matrix. O^* and A^* are presented as typical variables with the transformed data as covariance matrices, O^*, A^* having non-zero correlation only at the corresponding indices, while the canonical variation of each data set is uncorrelated. After that, we perform

feature fusion using the Z_1 formula of Classical Correlated Discriminant Features (CCDFs) as (15):

$$Z = \begin{pmatrix} O^* \\ A^* \end{pmatrix} = \begin{pmatrix} W_Q^T O \\ W_A^T A \end{pmatrix} = \begin{pmatrix} W_O & 0 \\ 0 & W_A \end{pmatrix}^T \begin{pmatrix} O \\ A \end{pmatrix} \quad (15)$$

The fused features will be used as input to the LOF, which calculates outliers for the dataset by assigning each object an outlier factor based on anomalous properties relative to its surrounding space.

C. SEARCH OUTLIER

The Local Outlier Factor algorithm is a density-based method for detecting outliers. The matrix Z generated by feature fusion is used as the input to the LOF, and the algorithm calculates the local outliers of the matrix Z by assigning an outlier factor to each object through the outlier properties of each object relative to the surrounding space. In this process, the local outlier factor of point p of matrix Z is expressed as (16):

$$LOF_k(p) = \frac{\sum o \in N_k(p) \frac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|} \quad (16)$$

The o represents another point o in the matrix, $N_k(p)$ represents the set of k nearest neighbors of p,and $lrd_k(p)$ represents the locally reachable density at point p. k-distance(o) denotes the location of the kth distance from o, excluding the point o. $d(o,p)$ means the distance between the point o and the point p. The formula is (17):

$$lrd_k(p) = \frac{|N_k(p)|}{\sum N_k(p)\, max\{k - dis\ tance(o), d(o,p)\}} \quad (17)$$

The larger the $LOF_k(p)$ value, the more likely it is to be an outlier. Finally, the outlier points can be derived by performing a descending sorting operation on the local outlier factors of each object in matrix Z and comparing the magnitude with a custom threshold.

IV. PERFORMANCE EVALUATION

In order to verify the effectiveness of GCA method, this paper compares it with Auto-encoder (AE) algorithm, Connectivity outlier factor algorithm (COF), Cut-point clustering (CutPC) algorithm, Isolation Forest (IForest) algorithm, K-Nearest Neighbor (KNN) algorithm,Outrank algorithm, SO-GAAL and MO-GAAL. We chose Network-based, Density -based, Graph -based, Integration -based, and Distance -based outlier detection algorithms. The performance of the algorithms can be evaluated more intuitively by comparing them with several algorithms from different domains. The data are taken from the average of ten experimental results. This section describes the experimental setup and evaluation criteria and performs the ablation experiments.

A. DATASETS

We selected ten representative datasets from different domains in the UCI repository proposed by the University of California Irvine [52], which is most commonly used for machine learning tests. To eliminate the effect of dimensionality and thus make the influence of each feature dimension on the objective function consistent and to improve the convergence speed of the iterative solution, all data sets are normalized using the maximum and minimum values in the data columns, and the normalized values are between [0, 1], and the maximum-minimum normalization formula is shown in (18):

$$DS_i = \frac{(d_i - d_{m\ in})}{(d_{m\ ax} - d_{m\ in})} \quad (18)$$

where DS_i is the i-th normalized value in the dataset, d_i is the original value of the i-th index, dmax is the maximum value of the i-th index, and dmin is the minimum value of the ith index.

All data sets are taken from the real world. Arrhythmia distinguishes the presence or absence of arrhythmia; Breastw is from the Wisconsin Breast Cancer Data; cardio includes fetal heart rate and uterine contraction characteristics based on the classification of expert obstetricians; Glass describes the oxide content of glass; ionosphre is a classification of radar echoes from the ionosphere; lympho is from the University Medical Center lymphography data; Pima from the National Institute of Diabetes and Digestive and Kidney Diseases; Vowels, a dataset that records 12 LPC cepstrum coefficients for 640-time series, and Wbc, also from breast cancer data; Wine, which focuses on the use of chemical analysis to determine the origin of wine; These datasets from medical or industrial sources are somewhat representative of the real world.

The following 2 describes the details of the datasets, including the size of the datasets and the outliers point rate et al:

TABLE 2. Datasets description.

Datasets	Size	Attribute	Outliers	Outlier Rate
Arrhythmia	452	257	66	14.6%
Breastw	683	9	239	35%
Cardio	1822	21	175	9.6%
Glass	214	9	9	4.2%
Ionosphre	351	33	127	36.1%
lympho	148	18	6	4.1%
Pima	768	8	268	34.8%
Vowels	1456	12	50	3.4%
Wbc	377	30	20	5.3%
Wine	129	13	10	7.7%

B. EVALUATION METRICS

The AUC (Area Under the receiver operating characteristic Curve), ACC (Accuracy), and FAR (False Alarm Rate) are used as the evaluation criteria for algorithm performance. AUC is the area between the ROC (Receiver Operating Characteristic) curve and the horizontal axis; ROC is a curve in which horizontal and vertical coordinates are, respectively, FPR (False Positive Rate) and TPR (True Positive Rate).

AUC can be calculated from (19):

$$AUC = \frac{\sum_{i=1}^{h-1} (x_{i+1} - x_i) \times (y_{i+1} - y_i)}{2} \tag{19}$$

x_i' and y_i' are the horizontal and vertical coordinates of the i-th sample, respectively, and i is a positive integer; $h+$ and $h-$ are the number of positive and negative cases, respectively; h is the total number of samples. ACC and FAR is shown in (20) and (21):

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{20}$$

$$FAR = \frac{FP}{TN + FP} \tag{21}$$

The following 3 describes the definition of each parameter.

TABLE 3. TP, TN, FP, FN conceptual analysis.

Real situation	Forecast situation	
	Positive Example	Counter Example
Positive Example	TP (True Positive)	FN (False Negative)
Counter Example	FP (False Positive)	TN (True Negative)

C. EXPERIMENTAL SETUP

Our experiments were implemented on an Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz PC using the Matlab code (R2021b).

The GCA and the comparison algorithm proposed in this paper are implemented in Matlab. In GCN module, a two-layer GCN is trained, with labels not involved in the training, and the inputs are the original dataset and the directed graph generated by the previous module. The learning rate is dynamically adjusted after setting the initial value.

4 indicates the setting of parameters in the GCA algorithm, while 5 implies the setting of parameters in the comparison algorithm:

The AE algorithm includes the data input layer, hidden layer, and output reconstruction layer, and the input is the feature dataset X. For the COF algorithm, the Number of nearest neighbors is used to construct the SBN path, i.e., the Number of neighbors for each observation to be compared with the link distance. The input of CutPC is the only X feature set. The training and test sets in KNN are split 7:3. The parameters of the MO-GAAL algorithm are inherited from SoGAAL except for the Number of sub_generator.

D. EXPERIMENTAL RESULTS

The experiment is executed, the mean value is calculated for the 10 times results as the final result to ensure more realistic experimental results. Fig.5 shows the experimental AUC results of the GCA algorithm with the remaining eight comparison algorithms on the Arrhythmia, Breastw, Cardio, Glass and Ionosphre data sets. Fig.6 shows the experimental

TABLE 4. GCA's experimental parameter settings.

Parameters	Modules	Descriptions	Setting
k	Figure Construction	Number of connected neighbors	Selected 20 times within the range of 2-200 (for the best test result)
i	Graph Convolution	Number of iterations of GCN network	100
b	Graph Convolution	Batch size for GCN networks	Number of data records
α	Graph Convolution	Initial value of learning rate	0.01
h	Graph Convolution	Scale of hidden layers	The first level is two and the second level is the number of data records.
K	LOF Algorithm	Number of connected neighbors	Selected 20 times within the range of 2-200 (for the best test result)

TABLE 5. Comparison algorithm's experimental parameter settings.

Parameters	Algorithms	Descriptions	Setting
Hiddenlayer number	AE	Minimum number of hidden layer nodes	2
Learning rate	AE	Initial value of learning rate	0.001
Neighbors number	COF	Number of nearest neighbors	Selected 20 times within the range of 2-100 (for the best test result)
Sample size	IForest	The size of sample	256
Trees number	IForest	The number of trees in the iforest model	100
Neighbors number	KNN	Number of near-est neighbors	Selected 20 times within the range of 2-100 (for the best test result)
Damping factor	Outrank	Damping factor	0.85
stop_epochs	SO-GAAL	Stop training generator after stop_epochs	100
Learning rate	SO-GAAL	Learning rate of discriminator and generator	0.001
Sub_generator	MO-GAAL	Number of sub_generator	Selected 20 times within the range of 2-10 (for the best test result)

AUC results of the GCA algorithm with the remaining eight comparison algorithms on the lympho, Pima, Vowels, Wbc, Wine data sets.

From the AUC comparison results, it can be seen that the GCA algorithm proposed in this paper significantly outperforms the remaining eight comparison algorithms in detecting outliers on nine datasets. On the Cardio dataset, the accuracy of the GCA method is second to the KNN method by a tiny margin of 0.005. GCA performed the best on the Vowels dataset, 4.51 percentage points higher than the second most

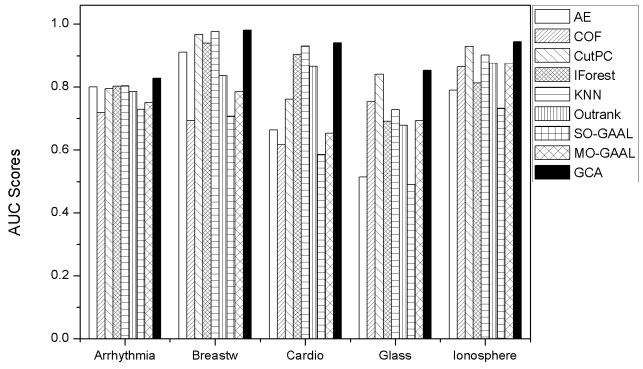

FIGURE 5. AUC metrics of GCA with the eight comparison algorithms on five data sets.

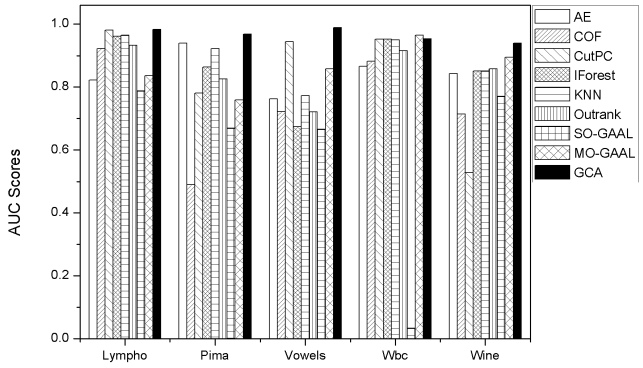

FIGURE 6. AUC metrics of GCA with the eight comparison algorithms on five data sets.

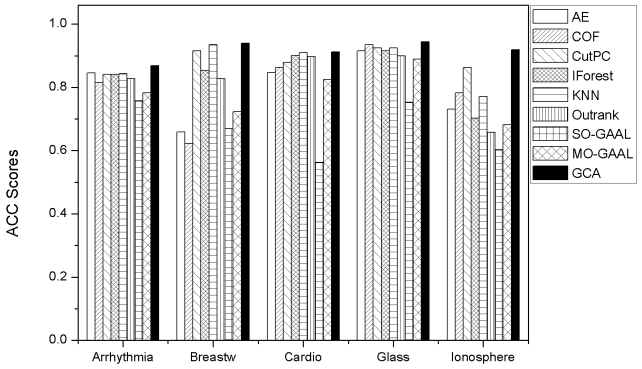

FIGURE 7. ACC metrics of GCA with the remaining eight algorithms on five data sets.

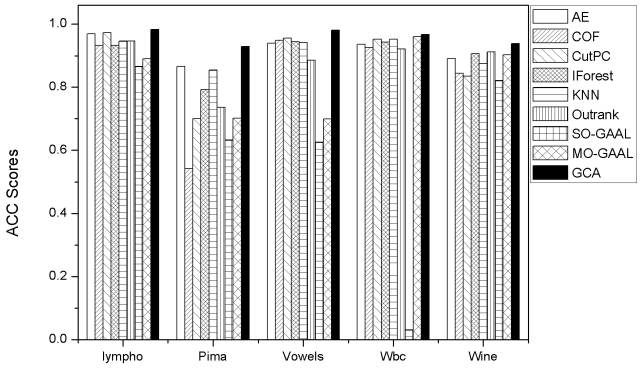

FIGURE 8. ACC metrics of GCA with the remaining eight algorithms on five data sets.

accurate CutPc algorithm. The average AUC of GCA on the ten datasets was 0.9319, and the average AUC of KNN, the next best performer, was 0.8797 on the ten datasets. The average AUCs of AE, COF, CutPC, IForest, Outrank, SO-GAAL, and MO-GAAL are 0.7907, 0.7379, 0.8476, 0.8449, 0.8291, 0.6165, and 0.8066, respectively. The experimental results show that the AUC accuracy of GCA far exceeds other traditional or for network-based detection algorithms; one of the reasons may be that the dataset is small. For example, the GAAL algorithm does not perform well in small datasets.

Fig.7 shows the experimental ACC results of the GCA algorithm with the remaining eight comparison algorithms on the Arrhythmia, Breastw, Cardio, Glass and Ionosphre data sets. Fig.8 shows the experimental ACC results of the GCA algorithm with the remaining eight comparison algorithms on the lympho, Pima, Vowels, Wbc, Wine data sets.

The comparison of ACC metrics in Fig. 7 and Fig. 8 shows the GCA achieved the best results in all datasets. The advantage of GCA is most apparent in the Ionosphere and Wine datasets, which are 4.58 and 3.19 percentage points higher than the second-best performing algorithms, CutPC and IForest, respectively. The average ACC value of the GCA algorithm is 0.9320, and the next best performer is still the KNN algorithm, with an average value of 0.8954. The average AUCs of AE, COF, CutPC, IForest, Outrank, SO-GAAL, and MO-GAAL are 0.8597, 0.8211, 0.8838, 0.8731, 0.8510, 0.6319, 0.8056.

The data on the false positive rate of GCA and the remaining eight comparison algorithms on ten datasets are recorded in 6.

The comparison of the FAR metrics in 6 shows that the GCA algorithm achieves the lowest false positive rate on the nine datasets. GCA is 5 percentage points lower than the second-best KNN algorithm in terms of the average FAR metric and 5 percentage points lower than the COF average on FAR by 93%.

In summary, the GCA algorithm performs well compared to the more classical and popular outlier detection algorithms. The significant improvement in detection performance of the proposed GCA algorithm is mainly due to the use of two channels to jointly extract object features, which enhances the capability of the original separate feature mapping and, therefore, can better learn the potential feature data points in the data. Aggregating neighboring nodes can better capture the global information of the graph to represent the nodes' characteristics better. On the one hand, the detailed knowledge of the specific target of interest is learned, focusing more on the critical information than on the whole, enhancing the data features while discarding the interference of some useless knowledge to the LOF algorithm during detection.

TABLE 6. Comparison of GCA with the remaining eight algorithms in terms of FAR metrics.

	Arrhythmia	Breastw	Cardio	Glass	Ionosphere	Lympho	Pima	Vowels	Wbc	Wine
AE	0.0093	0.0344	0.0083	0.0044	0.0209	0.0032	0.0102	0.0031	0.0034	0.0059
COF	0.1088	0.2905	0.0759	0.0343	0.1659	0.0352	0.3520	0.0250	0.0392	0.0840
CutPC	0.0933	0.0054	0.0674	0.0392	0.0103	0.0141	0.2300	0.0086	0.0252	0.0840
IForest	0.0091	0.0228	0.0060	0.0044	0.0231	0.0036	0.0160	0.0029	0.0027	0.0051
KNN	0.0112	0.0120	**0.0049**	0.0040	0.0198	0.0041	0.0121	0.0030	0.0030	0.0108
SO-GAAL	0.1221	0.0354	0.1102	0.0085	0.0321	0.0499	0.3129	0.3475	0.3542	0.1252
MO-GAAL	0.1168	0.0282	0.0815	0.0182	0.2103	0.0421	0.2230	0.1331	0.0397	0.0782
Outrank	0.0097	0.0521	0.0062	0.0042	0.0056	0.0023	0.0323	0.0580	0.0450	0.0096
GCA	**0.0082**	**0.0113**	0.0052	**0.0034**	**0.0067**	**0.0014**	**0.0054**	**0.0021**	**0.0022**	**0.0033**

TABLE 7. Effect of GCN inclusion and attention inclusion on the final AUC results.

	Arrhythmia	Breastw	Cardio	Glass	Ionosphere	Lympho	Pima	Vowels	Wbc	Wine
LOF	0.7363	0.7887	0.8675	0.6812	0.8505	0.8979	0.6872	0.6016	0.9471	0.8112
GCN	0.792	0.9205	0.9126	0.8094	0.9063	0.9026	0.911	0.9267	0.9146	0.8731
Attention	0.8057	0.9377	0.8622	0.817	0.9003	0.9108	0.9264	0.8684	0.9348	0.9084
GCA	**0.8175**	**0.9769**	**0.9306**	**0.8426**	**0.9287**	**0.9671**	**0.9579**	**0.9438**	**0.9535**	**0.9294**

TABLE 8. Effect of GCN inclusion and attention inclusion on the final ACC results.

	Arrhythmia	Breastw	Cardio	Glass	Ionosphere	Lympho	Pima	Vowels	Wbc	Wine
LOF	0.7562	0.7856	0.8375	0.6812	0.8452	0.9256	0.7085	0.6451	0.9379	0.8523
GCN	0.8209	0.9023	0.8965	0.8194	0.9062	0.9631	0.8955	0.9564	0.9485	0.9235
Attention	0.8057	0.9015	0.8922	0.8070	0.8998	0.9482	0.9056	0.8654	0.9053	0.9068
GCA	**0.8584**	**0.9297**	**0.9024**	**0.9343**	**0.9285**	**0.9729**	**0.9296**	**0.98072**	**0.9575**	**0.9379**

TABLE 9. Effect of GCN inclusion and attention inclusion on the final FAR results.

	Arrhythmia	Breastw	Cardio	Glass	Ionosphere	Lympho	Pima	Vowels	Wbc	Wine
LOF	0.0097	0.0208	0.0078	0.0059	0.0082	0.0026	0.0122	0.0037	0.0042	0.0052
GCN	0.0085	0.0125	0.0064	0.0039	0.0075	0.0016	0.0650	0.0031	0.0029	0.0042
Attention	0.0089	0.0139	0.0069	0.0045	0.0069	0.0018	0.0942	0.0032	0.0032	0.0036
GCA	**0.0082**	**0.0113**	**0.0052**	**0.0034**	**0.0067**	**0.0014**	**0.0054**	**0.0021**	**0.0022**	**0.0033**

E. ABLATION STUDY

To demonstrate the positive effect of GCN and Attention on outlier detection, we evaluated the effect of each module on detection performance with LOF as the baseline, AUC, ACC, and FAR metrics on ten datasets. Also, the effect of the number of layers of GCN on the final results was investigated.

In order to verify the effect of GCN, the feature matrices are directly outlier detection after GCN layer training; to verify the effect of Attention, the outlier detection results are obtained by inputting the reconstructed matrix of Attention into LOF.

7, 8, and 9 show the effects of GCN inclusion and attention inclusion on the final AUC, ACC, and FAR, respectively. The first row indicates the detection results of LOF. The second row corresponds to the case after using GCN alone. The third row shows the improvement of LOF by using Attention alone. The fourth row shows the results of the proposed algorithm GCA.

Comparing the GCA model output with the experimental results of other separate modules shows that using only the LOF feature matrix is not enhanced. The node information is not significant, leading to the low accuracy of the output results.

Outlier detection using the GCN module shows a significant improvement in accuracy, indicating that the interconnections in the features have an enormous impact on the feature matrix, building significant topology. Thus, GCN can effectively extract global information and enhance the overall model. The use of LOF alone makes the graph not have structural information and only contains node information, which reduces the accuracy rate and can only achieve specific results.

Similarly, the Attention module achieves comparable effect accuracy, indicating that Attention can capture the degree of importance among the nodes and assign higher weights to the virtual nodes. It can be seen that focusing on the critical part of the feature matrix is distinguishable from unimportant information and can significantly impact the results.

A comprehensive analysis can conclude that using GCN can take advantage of the critical role of graph linking relationships, focus on global features, and effectively enrich the features of entities. Combined with Attention, the

information of each node can be effectively aggregated, and the global and partial parts work together.

Drawing on Kipf and Welling experiments on the depth of the GCN on semi-supervised classification work [53], this paper explores the effect on the final results when the GCN is at 2, 4, and 6 layers, where the parameter dataset is essentially constant except for the depth during the comparison experiments. It can be seen that the optimal results are achieved when the GCN is at 2 layers. The AUC results for different number of layers of GCN are shown in Fig.9 and Fig.10. Fig.9 shows the variation of AUC when the number of GCN layers varies in Arrhythmia, Breastw, Cardio, Glass and Ionosphre. Fig.10 shows the variation in datasets lympho, Pima, Vowels, Wbc, Wine. Based on the above ablation experiments, it can be concluded that GCN, attention mechanism, and feature fusion all improve the accuracy of outlier finding, and the best results are obtained when the GCN is 2 layers.

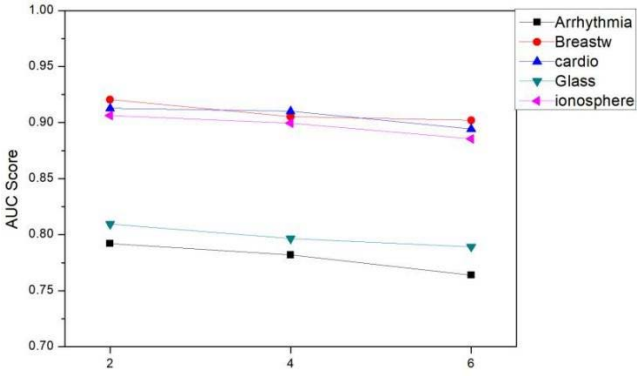

FIGURE 9. Effect of GCN's number of layers on the final results on five data sets.

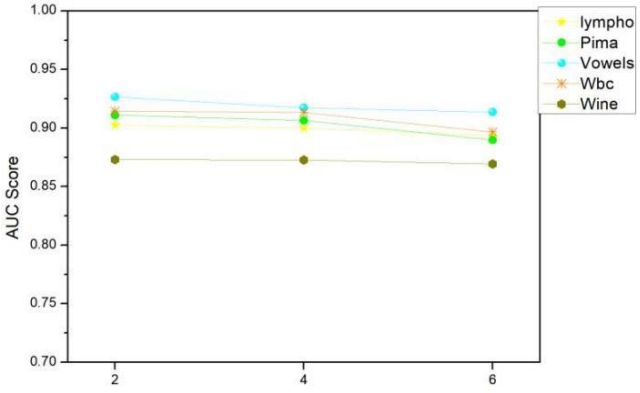

FIGURE 10. Effect of GCN's number of layers on the final results on five data sets.

Also, according to the observations in the experiments, the experiment time increases significantly with each additional layer. With each extra layer, the adequate context size of each node increases with the size of its neighbors. As the number of layers increases, the nodes learn more information about their neighboring nodes, leading to an increasing convergence of nodes, resulting in a general decrease in the results.

V. CONCLUSION

In order to solve the problem that outliers are difficult to detect in the data set with obscure features and further improve the accuracy of the outlier detection algorithm, this paper proposes the Graph Convolutional and Attention–Based Outlier Detection algorithm to solve the problem of obscure features by using GCN and Attention for feature In this paper, we propose a Graph Convolutional and Attention–Based Outlier Detection algorithm to solve the problem of inconspicuous features. Experimentally, we use ten publicly available UCI datasets after maximum-minimum normalization to compare with the remaining eight classical and widely used outlier detection algorithms. The GCA algorithm excels in the AUC, ACC, and FAR metrics, significantly improving outlier detection performance. Further, we conducted ablation experiments to verify the effectiveness of the method used in our model by comparing LOF outlier detection with the addition of a GCN module alone and an Attention module alone and by experimenting with the number of GCN layers.

The reasons for the superior performance of GCA in outlier detection mainly come from the following three aspects:

1) Our model uses LSH to reconstruct the map to re-extract the original features as well as eliminate redundant information, discard some useless features, facilitate the operation of the LOF algorithm, and improve the accuracy of outlier detection.
2) Graph Convolutional Network scaling the comprehensive features, Attention mechanism focuses on the key features, increases the attention to the key information, amplifies the useful features, and widens the gap between outliers and normal values.
3) The features extracted and combined from dual channels are more discriminative than GCN or Attention alone by Canonical Correlation Analysis Feature fusion.

Based on the above analysis, the GCA algorithm has some advantages in comparison with other algorithms. However, the GCA algorithm still has a lot of room for improvement. In this paper, we have only implemented outlier detection in the text. In future work, we will study the application in image datasets to solve more complex image outlier detection problems.

REFERENCES

[1] A. Ayadi, O. Ghorbel, A. M. Obeid, and M. Abid, "Outlier detection approaches for wireless sensor networks: A survey," *Comput. Netw.*, vol. 129, pp. 319–333, Dec. 2017.
[2] K. G. Al-Hashedi and P. Magalingam, "Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019," *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100402.
[3] C. P. Caroline and S. Thomas George, "An outlier detection approach on credit card fraud detection using machine learning: A comparative analysis on supervised and unsupervised learning," in *Intelligence in Big Data Technologies—Beyond the Hype*. Singapore: Springer, 2021, pp. 125–135 .
[4] P. Roy, P. Rao, J. Gajre, K. Katake, A. Jagtap, and Y. Gajmal, "Comprehensive analysis for fraud detection of credit card through machine learning," in *Proc. Int. Conf. Emerg. Smart Comput. Informat. (ESCI)*, Mar. 2021, pp. 765–769.

[5] X. Hu, S. Hu, Y. Huang, H. Zhang, and H. Wu, "Video anomaly detection using deep incremental slow feature analysis network," *IET Comput. Vis.*, vol. 10, no. 4, pp. 258–267, 2016.

[6] Y. S. Chong and Y. H. Tay, "Abnormal event detection in videos using spatiotemporal autoencoder," in *Proc. Int. Symp. Neural Netw.* Cham, Switzerland: Springer, 2017, pp. 189–196.

[7] H. Yang, B. Wang, S. Lin, D. Wipf, M. Guo, and B. Guo, "Unsupervised extraction of video highlights via robust recurrent auto-encoders," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4633–4641.

[8] W. Wang, Y. Xie, and X. Wang, "Detection of multicamera pedestrian trajectory outliers in geographic scene," *Wireless Commun. Mobile Comput.*, vol. 2022, pp. 1–12, Apr. 2022.

[9] A. Vijayan, B. Meenaskshi, A. Pandey, A. Patel, and A. Jain, "Video anomaly detection in surveillance cameras," in *Proc. Int. Conf. Advancement Technol. (ICONAT)*, Jan. 2022, pp. 1–4.

[10] D. S. Terzi, R. Terzi, and S. Sagiroglu, "Big data analytics for network anomaly detection from netflow data," in *Proc. Int. Conf. Comput. Sci. Eng. (UBMK)*, Oct. 2017, pp. 592–597.

[11] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han, "Enhanced network anomaly detection based on deep neural networks," *IEEE Access*, vol. 6, pp. 48231–48246, 2018.

[12] Y. Wang, J. Ma, A. Sharma, P. K. Singh, G. S. Gaba, M. Masud, and M. Baz, "An exhaustive research on the application of intrusion detection technology in computer network security in sensor networks," *J. Sensors*, vol. 2021, May 2021, Art. no. 5558860.

[13] A. Gaddam, T. Wilkin, M. Angelova, and J. Gaddam, "Detecting sensor faults, anomalies and outliers in the Internet of Things: A survey on the challenges and solutions," *Electronics*, vol. 9, no. 3, p. 511, Mar. 2020.

[14] N. Moustafa, K. R. Choo, I. Radwan, and S. Camtepe, "Outlier Dirichlet mixture mechanism: Adversarial statistical learning for anomaly detection in the fog," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 8, pp. 1975–1987, Aug. 2019.

[15] T. Driver, M. Dor, K. Skinner, and P. Tsiotras, "Space carving in space: A visual-SLAM approach to 3D shape reconstruction of a small celestial body," in *Proc. AAS/AIAA Astrodyn. Spec. Conf.*, Aug. 2020, pp. 9–13.

[16] G. Gao, S. Gao, G. Hu, and X. Peng, "Spectral redshift observation-based SINS/SRS/CNS integration with an adaptive fault-tolerant cubature Kalman filter," *Meas. Sci. Technol.*, vol. 32, no. 9, Sep. 2021, Art. no. 095103.

[17] K. Zhang, M. Hutter, and H. Jin, "A new local distance-based outlier detection approach for scattered real-world data," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Berlin, Germany: Springer, Apr. 2009, pp. 813–822.

[18] B. Tang and H. He, "A local density-based approach for outlier detection," *Neurocomputing*, vol. 241, pp. 171–180, Jun. 2017.

[19] H. Liu, X. Xu, E. Li, S. Zhang, and X. Li, "Anomaly detection with representative neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Sep. 14, 2021, doi: 10.1109/TNNLS.2021.3109898.

[20] M. Ur Rehman and D. Muhammad Khan, "A novel density-based technique for outlier detection of high dimensional data utilizing full feature space," *Inf. Technol. Control*, vol. 50, no. 1, pp. 138–152, Mar. 2021.

[21] C.-H. Lin, K.-C. Hsu, K. R. Johnson, M. Luby, and Y. C. Fann, "Applying density-based outlier identifications using multiple datasets for validation of stroke clinical outcomes," *Int. J. Med. Informat.*, vol. 132, Dec. 2019, Art. no. 103988.

[22] X. Qin, L. Cao, E. A. Rundensteiner, and S. Madden, "Scalable kernel density estimation-based local outlier detection over large data streams," Tech. Rep., 2019.

[23] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung, "Enhancing effectiveness of outlier detections for low density patterns," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining* Berlin, Germany: Springer, May 2002, pp. 535–548.

[24] H. Liu, J. Li, Y. Wu, and Y. Fu, "Clustering with outlier removal," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2369–2379, Jun. 2021.

[25] S. Askari, "Fuzzy C-means clustering algorithm for data with unequal cluster sizes and contaminated with noise and outliers: Review and development," *Expert Syst. Appl.*, vol. 165, Mar. 2021, Art. no. 113856.

[26] V. Barnett and T. Lewis, "Outliers in statistical data," in *Wiley Series in Probability and Mathematical Statistics* (Applied Probability and Statistics), 1984.

[27] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 413–422.

[28] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," in *Proc. 19th IEEE Int. Conf. Mobile Data Manage. (MDM)*, Jun. 2018, pp. 125–134.

[29] L.-T. Li, Z.-Y. Xiong, Q.-Z. Dai, Y.-F. Zha, Y.-F. Zhang, and J.-P. Dan, "A novel graph-based clustering method using noise cutting," *Inf. Syst.*, vol. 91, Jul. 2020, Art. no. 101504.

[30] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient kNN classification with different numbers of nearest neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1774–1785, May 2018.

[31] S. D. Bay and M. Schwabacher, "Mining distance-based outliers in near linear time with randomization and a simple pruning rule," in *Proc. KDD*, 2003, pp. 29–38.

[32] F. Angiulli, S. Basta, and C. Pizzuti, "Distance-based detection and prediction of outliers," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 2, pp. 145–160, Feb. 2005.

[33] Z. Liu, "Voltage and frequency droop control of a microgrid in islanded modes," Ph.D. dissertation, Murdoch Univ., Perth, WA, Australia, 2016.

[34] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, May 2000, pp. 93–104.

[35] W. Jin, A. K. Tung, J. Han, and W. Wang, "Ranking outliers using symmetric neighborhood relationship," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining* Berlin, Germany: Springer, Apr. 2006, pp. 577–593.

[36] J. Laurikkala, M. Juhola, E. Kentala, N. Lavrac, S. Miksch, and B. Kavsek, "Informal identification of outliers in medical data," in *Proc. 5th Int. Workshop Intell. Data Anal. Med. Pharmaco.*, vol. 1, Aug. 2000, pp. 20–24.

[37] E. Eskin, "Anomaly detection over noisy data using learned probability distributions," Tech. Rep., 2000.

[38] M. Goldstein and A. Dengel, "Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm," in *KI-2012: Poster and Demo Track*, vol. 9. 2012.

[39] M. Ahmed and A. Naser, "A novel approach for outlier detection and clustering improvement," in *Proc. IEEE 8th Conf. Ind. Electron. Appl. (ICIEA)*, Jun. 2013, pp. 577–582.

[40] G. Gan and M. K.-P. Ng, "K-means clustering with outlier removal," *Pattern Recognit. Lett.*, vol. 90, pp. 8–14, Apr. 2017.

[41] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2005, pp. 157–166.

[42] Y. Zhao and M. K. Hryniewicki, "XGBOD: Improving supervised outlier detection with unsupervised representation learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[43] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, "Outlier detection with autoencoder ensembles," in *Proc. SIAM Int. Conf. Data Mining*, Jun. 2017, pp. 90–98.

[44] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 665–674.

[45] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier detection for time series with recurrent autoencoder ensembles," in *IJCAI*, Aug. 2019, pp. 2725–2732.

[46] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative adversarial active learning for unsupervised outlier detection," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 8, pp. 1517–1528, Aug. 2020.

[47] H. D. K. Moonesinghe and P.-N. Tan, "Outrank: A graph-based outlier detection framework using random walk," *Int. J. Artif. Intell. Tools*, vol. 17, no. 1, pp. 19–36, 2008.

[48] C. Wang, H. Gao, Z. Liu, and Y. Fu, "A new outlier detection model using random walk on local information graph," *IEEE Access*, vol. 6, pp. 75531–75544, 2018.

[49] F. Chung, "Laplacians and the Cheeger inequality for directed graphs," *Ann. Combinatorics*, vol. 9, no. 1, pp. 1–19, 2005.

[50] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, Mar. 2010, pp. 249–256.

[51] M. Haghighat, M. Abdel-Mottaleb, and W. Alhalabi, "Fully automatic face normalization and single sample face recognition in unconstrained environments," *Expert Syst. Appl.*, vol. 47, pp. 23–34, Apr. 2016.

[52] A. Arthur and N. David, "UCI machine learning repository: Data sets," Tech. Rep., 2007. [Online]. Available: https://archive.ics.uci.edu/ml/datasets.php

[53] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.