information

MDPI

Article

A Network Scanning Organization Discovery Method Based on Graph Convolutional Neural Network

Pengfei Xue [1], Luhan Dong [2], Chenyang Wang [1,*], Cheng Huang [3] and Jie Wang [1]

[1] College of Electronic Engineering, National University of Defense Technology, Hefei 230031, China; xuepengfei@nudt.edu.cn (P.X.); sa517349@mail.ustc.edu.cn (J.W.)
[2] School of Cybersecurity, Northwestern Polytechnical University, Xi'an 710072, China; dongluhan@mail.nwpu.edu.cn
[3] School of Cyber Science and Engineering, Sichuan University, Chengdu 610207, China; codesec@scu.edu.cn
* Correspondence: wcy@nudt.edu.cn

Abstract

With the quick development of network technology, the number of active IoT devices is growing rapidly. Numerous network scanning organizations have emerged to scan and detect network assets around the clock. This greatly facilitates illegal cyberattacks and adversely affects cybersecurity. Therefore, it is important to discover and identify network scanning organizations on the Internet. Motivated by this, we propose a network scanning organization discovery method based on a graph convolutional neural network, which can effectively cluster out network scanning organizations. First, we constructed a network scanning attribute graph to represent the topological relationship between network scanning behaviors and targets. Then, we extract the deep feature relationships in the attribute graph via graph convolutional neural network and perform clustering to get network scanning organizations. Finally, the effectiveness of the method proposed in this paper is experimentally verified with an accuracy of 83.41% for the identification of network scanning organizations.

Keywords: network scanning; organization discovery; attribute graph; graph convolutional network; machine learning

Academic Editors: Krzysztof Szczypiorski and Daniel Paczesny

Received: 26 August 2025
Revised: 29 September 2025
Accepted: 9 October 2025
Published: 15 October 2025

Citation: Xue, P.; Dong, L.; Wang, C.; Huang, C.; Wang, J. A Network Scanning Organization Discovery Method Based on Graph Convolutional Neural Network. *Information* **2025**, *16*, 899. https://doi.org/10.3390/info16100899

1. Introduction

Network scanning refers to the act of scanning and detecting resources and their attributes in cyberspace, aiming to portray the attributes and states of cyberspace assets in an all-around way. Network assets include physical and virtual resources. Physical resources include a collection of network switching equipment and access equipment, etc. Access equipment refers to hardware devices used for remote access to network resources, encompassing types such as modems, switches, and routers. Virtual resources include information content, virtual users, and application services carried by physical resources. With the rapid development of network and communication technology, the application of network scanning technology is becoming more and more widespread. While this provides important support for network security protection, it also triggers many security risks. The potential negative impacts of network scanning and detection behaviors, such as privacy invasion, identity theft, and data leakage, are becoming more evident, posing potential risks to businesses and individuals.

Identifying the organization of the scanner holds significant value and importance for the following reasons: (1) Enhancing security protection for network devices: Identifying

the scanning organization enables us to design strategies specifically tailored to mitigate or prevent potential cyberattacks. (2) In-depth analysis of scanning behavior: Identifying scanning organizations and in-depth analysis of scanning activities enables us to deeply analyze their behavior. This helps us recognize their data collection priorities and preferred targets (such as specific IP addresses, ports, services, or software). This further assists us in devising strategies to evade scans, thereby avoiding the exposure of additional exploitable vulnerabilities. (3) In-depth analysis of scanning organizations: Identifying scanners and scanning organizations assists us in analyzing the scale and attack activities of potential attack groups.

Therefore, discovering network scanners and the organizations behind them is necessary to preserve network security. However, most of the existing work detects scanners and blocks detection activities on their own systems from a security perspective. In contrast, there has been little research into the identification of the organization behind the network detectors, ignoring the associative relationships between the network detectors. Cyber scanning organizations exhibit a number of behavioral characteristics. These behavioral characteristics include communication patterns, information transfer paths, and frequency of activity in the network, reflecting how the organization operates, its detection goals, and how active it is in the network. Therefore, modeling and extracting deep features of network probers and their interrelationships can effectively help us identify the organizations behind them.

Motivated by this, we propose a network scanning organization discovery method based on a graph convolutional neural network (GCN) in this paper, aiming to discover and identify the organizations behind network scanners. Firstly, we construct a network scanning attribute graph containing attributes such as IP address, port number, timestamp, whois information, and geographic information. Secondly, we construct a graph convolutional neural network to embed features into the nodes and edges of the attribute graph, thus effectively learning the nonlinear properties of the network. Thirdly, we implement a clustering algorithm on the extracted graph embedding representation to achieve clustering and identification of network scanning organizations. Finally, we validate the effectiveness of the method by conducting experiments on the constructed dataset. We construct a network scanning organization dataset containing 1,201,797 pieces of network scanning traffic data. Each piece of data contains information such as IP address, port number, protocol, etc., and describes 19 network scanning organizations. The proposed method in this paper achieves an identification accuracy of 83.41% on this dataset.

The key contributions of this paper are as follows:

- For the first time, we construct an exhaustive dataset of 19 network scanning organizations, including 1,201,797 pieces of network scanning traffic data.
- We propose a network scanning organization discovery method based on GCN, which models the correlations between network scanning behaviors to identify network scanning organizations.
- We construct an attribute graph to represent the network scanning behavior, use a Laplace filter to smooth the feature matrix and extract deep features by GCN, and finally use a clustering algorithm to identify organizations.
- The effectiveness of the proposed method is demonstrated through experiments, with an identification accuracy of 83.41%.

The remainder of this paper is organized as follows. In Section 2, we introduce related work on network scanning behavior identification and network scanning organization discovery. Section 3 describes the proposed network scanning organization discovery method. In Section 4, we evaluate the proposed method through clustering experiments.

In Section 5, we discuss the limitations and future work. Finally, Section 6 concludes the paper.

2. Related Work

With the rapid development of Internet technology, network security issues have become increasingly prominent. Network probing, i.e., various forms of scanning and monitoring of the target network, has become a common threat behavior. Such probing behaviors usually include port scanning, vulnerability scanning, network topology probing, etc., and their purpose is to discover the weak links in the target network for subsequent attacks. Therefore, the study of how to effectively detect and identify network probers has become one of the hot current issues in the network security field.

2.1. Network Scanning Behavior Identification

The goal of network scanning is to obtain comprehensive and complete information about various elements in cyberspace, which includes not only physical resources such as servers, routers, and terminal devices but also virtual resources such as users, services, IPs, and ports. The main methods of network scanning behavior discovery are network traffic analysis, intrusion detection systems, log analysis and auditing, and port scanning tools.

Network Traffic Analysis. Network traffic analysis is currently one of the primary means of detecting network probing behavior. By monitoring and analyzing network traffic, researchers can identify anomalous packet patterns and thus deduce potential probing activities [1,2]. By analyzing network traffic, it is possible to identify anomalous behaviors and thus detect potential cyber attackers. In recent years, machine learning and deep learning methods have been widely used in this field. Camelo et al. [3] propose a spectrum-based procedure that uses a DL-based classifier to achieve traffic classification at any layer on the radio network stack. Jenefa et al. [4] provide a novel deep learning-based technique for network traffic classification. The proposed method leverages both convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to classify network traffic. In addition to this, there are other rule-based [5–7] and machine learning traffic analysis methods [8–10] to identify scanning behavior.

Intrusion Detection System. Intrusion detection systems play an important role in identifying network probes as part of an active defense mechanism. Traditional signature-based IDSs identify known probes by matching network traffic with a library of pre-defined attack signatures. However, with the emergence of new attack techniques, behavioral analysis-based IDSs are gaining traction. In recent years, deep learning techniques have also been applied to IDSs, significantly improving their detection accuracy and generalization capabilities [11,12]. Kurnala et al. [13] introduce a hybrid detection approach that uses deep learning techniques to improve intrusion detection accuracy and efficiency. Ashiku et al. [14] propose the use of deep learning architectures to develop an adaptive and resilient network intrusion detection system (IDS) to detect and classify network attacks.

Log Analysis and Auditing. System logs and network logs record various types of events during system operation, including user logins, service starts and stops, and error messages. Through in-depth analysis of these logs, researchers can trace the time, place, and initiator of the probing behavior. He et al. [15] first present a characterization study of the current state-of-the-art log parsers and evaluate their efficacy on five real-world datasets with over ten million log messages. Other rule-based, anomaly detection, and hybrid approaches have been successively proposed to detect anomalous behavior [16–18].

Port Scan Detection Technology. Some dedicated port scanning monitoring tools and techniques [19,20] can not only detect ongoing scanning activities in a timely manner but also take appropriate defensive measures, such as dynamically adjusting firewall rules to

prevent the source IP of the scan from continuing access. For example, PortSentry v1.2 is a well-known port scanning detection software that listens for abnormal connection requests on the local host and automatically blacklists them. In recent years, researchers have identified port scanning behavior using machine learning algorithms [21–23].

2.2. Network Scanning Organization Discovery

Researchers often use cluster analysis methods to discover hidden patterns and relationships in groups, e.g., to discover groups of users with similar interests in a social network, or to detect anomalous communication patterns in a communication network. Heli et al. [24] proposed the Network Embedding for node Clustering (NEC) algorithm, which learns both graph-structure-based representations and cluster-oriented representations, and then uses K-mean for community detection. Brigitte et al. [25] propose the density-based clustering model TCSC for detecting organizations in heterogeneous networks that are densely connected in both network and attribute space. Cui et al. [26] propose an Adaptive Graph Encoder (AGE), a new framework for attribute graph embedding, which utilizes supervised learning of high and low similarity node pairs through an adaptive encoder, and finally performs Spectral clustering on the similarity matrices that preserve the embedding.

There are fewer existing studies related to network scanning organization discovery, focusing on collecting and analyzing network traffic to trace attackers. Richter et al. [27] track scanning activity through the lens of unsolicited traffic captured at the firewalls. Li et al. [28] design and implement a system for deploying and managing honeysites to attract and record bot traffic. Mazel et al. [29] present novel identification methods to identify ZMap scans with a small number of addresses extracted from the scan.

In summary, most of the existing methods focus on the discovery of network scanning behavior and the discovery of network probers. Few studies have focused on identifying the organizations behind these network probers. Identifying scanning organizations in cyberspace is important for protecting our cyber assets and reducing security threats.

3. Method

In recent years, scanning activities on the internet have exhibited a marked upward trend. A significant number of scanners have been deployed across the internet to gather extensive information on network devices, such as active hosts, open ports, service software versions, and operating systems. The disclosure of such information may pose a significant threat to network security. For instance, scanning may reveal the specific version of service software running on a target host. By correlating this information with publicly available vulnerability databases, potential software security vulnerabilities can be identified. Therefore, our method is designed for open-world environments, dedicated to identifying and collecting scanning organizations across the Internet.

The proposed network scanning organization discovery method based on GCN is shown in Figure 1. The framework contains four modules: *Attribute Graph Construction*, *Feature Extraction*, *Graph Embedding*, and *Organization Discovery*. We build multiple honeypot servers to collect scanning traffic logs from the network. We construct a network scanning behavioral attribute graph based on traffic log data. In the feature extraction stage, the labels of nodes and edges are first determined, and the log content is analyzed to determine the attributes of nodes and edges. Then the textual, categorical, and continuous numerical attributes are processed using the Term Frequency-Inverse Document Frequency (TF-IDF), One-Hot encoding, and Normalization approaches, respectively. The feature matrix processed by Laplace smoothing is input into the GCN model along with the adjacency matrix. The embedding representations of each node are learned by graph convolution

operation using the feature vectors of nodes and edges and the adjacency matrix. These embedding representations can comprehensively reflect the feature information of nodes and their topology in the network. Finally, each node of the attribute graph is mapped into a low-dimensional vector space, and the output embedding matrix is clustered as input to K-means clustering and Spectral clustering to identify network scanning organizations.

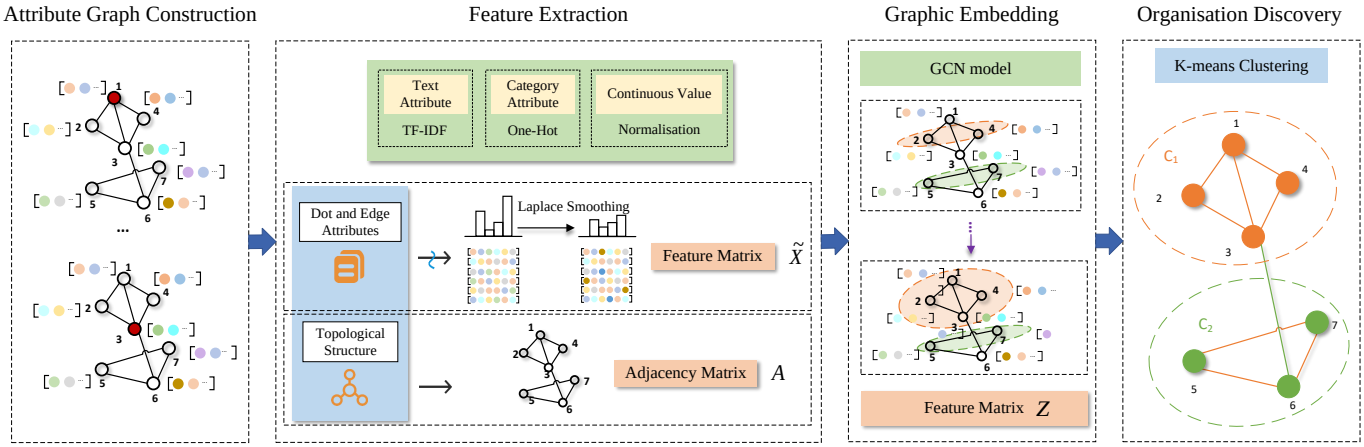


Figure 1. The framework of the proposed method.

3.1. Attribute Graph Construction

We construct attribute graphs to analyze the characteristics of network scanning behaviors, revealing interaction patterns and behavioral trends in cyberspace. We consider the hosts in the network as nodes (hosts) and the types of communication connections between hosts as edges (actions), which in turn allows us to extract and analyze features of the dynamic behavior of cyberspace. Node attributes include IP address (IP), port number (port), whois information (whois), country code (area code), city name (city), Internet Service Provider (isp), Autonomous System Number (asn), name of the organization to which the IP address belongs (org), province or state (p), latitude (lat) and longitude (lon). Side attributes include connection type (action), transport protocol, UTC, headers, headers_keys, headers_values, method, proto, URI, datagram information, uri, pack_datagram, and data_length. Examples of nodes in the attribute graph are shown in Table 1.

Table 1. Entities and descriptions in the network scanning attribute graph.

Entity Type	Entity Description and Example
Host Computer Equipment	Ip: IP Address, e.g., xxx.xxx.xxx.34 Port: Port Number, e.g., 42824 Whois: Information About The Domain Name Associated With The Source IP Address Areacode: The Code Of The Country Where The Source IP Address Is Located, e.g., US City: The Name Of The City Where The Source IP Address Is Located, e.g., San Francisco. Isp: Name Of The Internet Service Provider Of The Source IP Address, e.g., Enes Koken Asn: Autonomous System Number Assigned To Each ISP, e.g., 14061 Org: Name Of The Service Provider Or organization Managing The Source IP, e.g., DigitalOcean, LLC P: The Name Of The Province Or State Where The Source IP Address Is Located, e.g., California Lat: Latitude Where The Source IP Address Is Located, e.g., 37.775090 Lon: The Longitude Of The Source IP Address, e.g., −122.419640
Network Connection	Action: Type Of Request, e.g., Connect Transport Protocol: Type of transport protocol, e.g., TCP Utc: Universal Standard Time, e.g., 31 October 2023 11:57:00 p.m. Headers: Request Headers Stored As Key-Value Pairs Headers_keys: Keys For Request Headers Headers_values: Values For Request Headers Method: Request Method, Such As GET Proto: Protocol Type, e.g., HTTP/1.1 Uri: Uniform Resource Identifier Used To Indicate The Path To The Requested Resource, e.g., /manage/account/login Pack_datagram: Hexadecimal Representation Of The Packet Data_length: Length Of The Requested Data

Given an attribute graph $G = (V, E, X)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of vertices with a total of n nodes, $E = \{e_{11}, e_{12}, \ldots, e_{ij}, \ldots, e_{nn}\}$ is the set of edges with a total of m edges, $X = [x_1, x_2, \ldots, x_n]^T$ is the feature matrix of all nodes, $x_i \in R^d$ is the real-valued eigenvector of the node v_i, $x_k(v_i)$ is the k-th attribute of the node v_i, $dom(x_k)$ is the set of possible values of the k-th element of the attribute vector, i.e., the domain of the x_k. The topology of an attribute graph can be represented by an adjacency matrix $A = \{a_{ij}\} \in R_{n \times n}$. If $a_{ij} = 1$, it means that there is an edge from node v_i to node v_j.

3.2. Feature Extraction

The findings of many papers show that the 'topology' and 'feature information' of attribute graphs often provide complementary information, and that fusion of the two can improve the quality of feature representation. For example, feature information may help to solve the problem of missing or noisy attributes, while topology information can compensate for the structural sparsity of the network. Therefore, we need to extract features separately for attribute features and topology for subsequent feature embedding and representation. In order to accurately characterize the nodes and edges in a network, their attributes need to be refined. In this paper, three main feature processing methods are used: the TF-IDF, the One-Hot encoding, and the Normalization approach for textual, categorical, and continuous numerical attributes.

TF-IDF for textual attributes. TF-IDF is a feature vectorization method commonly used in text mining; this technique evaluates the importance of a word in a particular document, taking into account the frequency of distribution of the word in the entire document collection. TF-IDF consists of two parts: Term Frequency (TF) and Inverse Document Frequency (IDF), where TF indicates the proportion of the number of occurrences of a word in a document to the total number of words in the document, and IDF is used to measure the frequency of occurrences of a word in all the documents, thus determining its 'rarity'.

Assuming that there are k words in the document d_j, n_{kj} is the number of times the word t_k appears in the document d_j, and the sum of n_{kj} is the sum of the number of occurrences of all the words in the document. The formulae for the word frequency and the inverse document frequency, respectively, are as follows:

$$TF_{ij} = \frac{n_{ij}}{\sum_k n_{kj}} \tag{1}$$

$$IDF_i = lg \frac{|D|}{|\{j : t_i \in d_j\}|} \tag{2}$$

where $|D|$ denotes the total number of documents. The text data can be converted into numerical feature representations through TF-IDF, which can be used as input for subsequent machine learning. In this paper, we use TF-IDF to process the textual attributes of nodes and edges thereby effectively capturing the importance of keywords and providing rich semantic information for structural analysis of the network.

One-Hot for categorical attribute. One-Hot encoding is commonly used to deal with categorical attributes, which represent the mutual exclusivity between categories by converting each category attribute into a binary vector. Where only one position is a 1 and the rest of the positions are zeros. This approach is simple and intuitive and enables the model to understand that there is no numerical magnitude relationship between different categories, which can prevent the model from misinterpreting the category attributes. In this paper, the unique heat coding can help the model accurately distinguish the different category attributes of nodes and edges, and provide a basis for the subsequent cluster analysis.

Normalization for numerical attribute. Normalization approaches are commonly used to deal with continuous numerical attributes by scaling the attribute values to a specific range (e.g., between 0 and 1) or normalizing to a distribution with zero mean and unit variance. This approach can eliminate the effects of different magnitude attributes to a certain extent to ensure the stability and efficiency of model training. In this paper, the Normalization process ensures that continuous numerical attributes are well distributed in the feature matrix and improves the sensitivity of the algorithm to subtle differences in network structure.

The textual attributes processed by TF-IDF, the categorical attributes processed by solo thermal coding, and the continuous numerical attributes processed by Normalization are stitched into a feature matrix. This feature matrix provides a comprehensive representation of the nodes and edges in the network. Since the topology representation of the graph is also one of the core aspects of cyberspace mapping and its related research, it is directly related to the effectiveness and accuracy of the subsequent analysis and calculation. The adjacency matrix, as a classical method to represent the topology of a graph, has been widely adopted due to its intuition and ease of operation. Therefore, this paper adopts the adjacency matrix together with the feature matrix as the attribute representation of nodes.

Laplace feature smoothing. The basic assumption of graph learning is that the neighboring nodes on the graph should be similar and therefore the node features should be smooth on the graph flow shape. In order to obtain smoother signals and retain the low-frequency components while filtering out the high-frequency components while ensuring high computational efficiency, a Laplace smoothing filter is used in this paper. The Laplace smoothing filter is defined as follows:

$$H = I - KL = I - \frac{1}{\lambda_{max}} \tag{3}$$

where K is the real value, λ_{max} is the maximum eigenvalue and L is the symmetric normalized Laplace matrix. After stacking t Laplace smoothing filters, the filtered feature matrix is denoted by $\tilde{X} = H^t X$.

3.3. Graph Embedding

Research on graph embedding has emerged with the widespread use of graphs in a number of domains. These studies typically use graphs as input and incorporate auxiliary information to optimize the embedding process. In general, there are five different types of auxiliary information, which are labels, attributes, node features, information dissemination, and knowledge base. A label refers to the categorical marking of a node or edge, and nodes with different labels should be kept away from each other in the embedding space. Attributes can be classification labels or continuous values for nodes or edges. Node features consist mainly of text or image features, most of which are in the form of text. These textual features can be used directly as feature vectors for each node or in the form of documents. Features in the form of documents can be further processed by bag-of-words modeling, topic modeling, or treating 'words' as node types to extract feature vectors.

GCN, as an effective tool for graph embedding, has received much attention due to its excellent performance. GCN learns the feature representations of nodes by performing convolutional operations on graph-structured data and is able to efficiently capture the complex relationships between nodes and the rich information of node features. The core idea is to aggregate the features of each node with those of its neighboring nodes, so as to obtain a new feature representation of that node. The adjacency matrix and feature matrix are the important components of this algorithm, and both of them are used as inputs to gradually extract the high-level features of the graph data through the multilayer GCN

structure. In GCN models, preprocessing of the adjacency matrix is one of the key steps to improve the performance of the model. In order to increase the expressive power of the model, self-connections are usually added in and the following calculations are performed:

$$\tilde{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}} \tag{4}$$

Normalization is performed to balance the degree of influence of different nodes. GCN uses the normalized adjacency matrix and node feature matrix for feature aggregation to update the feature representation of each node. The feature aggregation process can be formalized as:

$$H(l+1) = \sigma(\tilde{A}H^{(l)}W^{(l)}) \tag{5}$$

where $H^{(l)}$ denotes the node identity matrix of the layer, $W^{(l)}$ is the weight matrix of the layer, and σ is the activation function. Through layer-by-layer aggregation, GCN is able to learn node embedding representations that contain both local graph structure and global graph information. After the multi-layer feature aggregation of the GCN model, the final generated embedding matrix contains the low-dimensional vector representation of each node in the graph. These embedding vectors reflect both the feature information of the node and the position information of the node in the graph. After embedding through the GCN graph, the feature representation is fed into the clustering algorithm to cluster out network scanning organizations, as shown in Figure 2. Notably, [1 0], [0 0], [0 1], [1 1] are the feature vectors of nodes. Node features primarily encompass textual or image features, represented in vector form, and are presented here merely as an illustrative example.

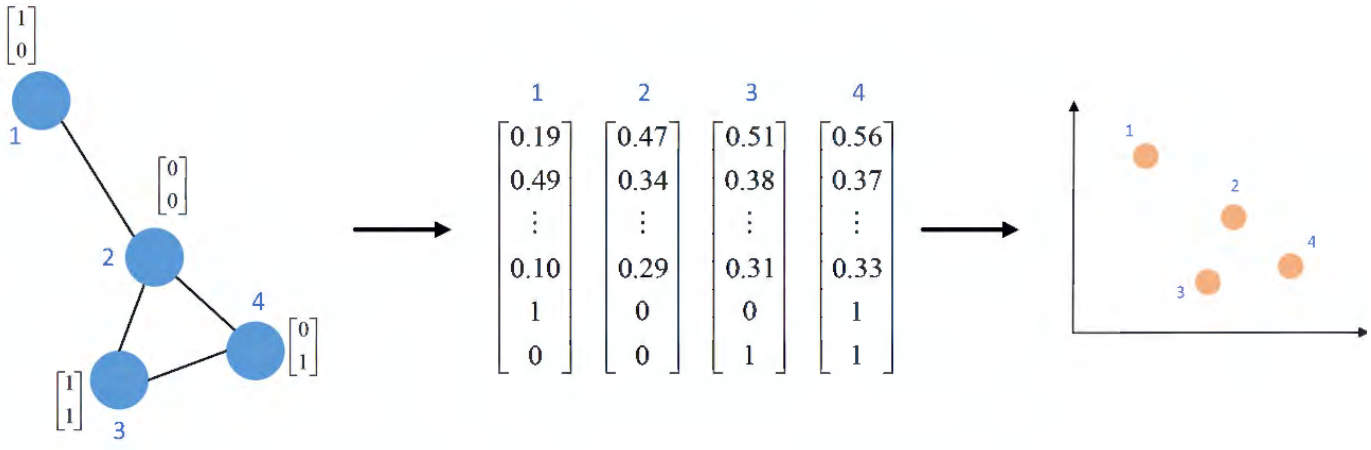


Figure 2. The process of graph embedding to obtain feature representations.

3.4. Organization Discovery

We use the embedding matrix generated by GCN for clustering analysis of network scanning organizations and design three organization discovery methods based on K-means, Spectral, and DBSCAN clustering methods, respectively.

GCN+K-means. K-means is an unsupervised learning clustering algorithm based on segmentation. The clustering algorithm divides the data points into distinct clusters, where each point in the graph belongs to the center of the cluster closest to itself. The K-means algorithm generally uses the Euclidean distance as a measure of the similarity between data points. It works by constantly updating the center of the cluster so that the similarity between the data points in the cluster and their corresponding centers is gradually reduced

by the sum of the squares of the intra-cluster errors. The Sum of Squared Errors (SSE) is calculated as follows:

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} |d(x, C_i)|^2 \tag{6}$$

SSE is used to measure the closeness of the clustering results—the smaller the SSE value is, the closer the clustering results are, i.e., the closer the data points are to the cluster centers they belong to. When SSE is stable or the objective function reaches convergence, the clustering process has reached a stable state, at which time the algorithm stops and outputs the final clustering results. The specific clustering process of K-means is to first randomly select k initial clustering centers $C_i (1 \leq i \leq k)$ and then calculate the Euclidean distance between each data point and these centers:

$$d(x, C_i) = \sqrt{\sum_{j=1}^{m} (x_j - C_{ij})^2} \tag{7}$$

where x is the data object, C_i is the i-th clustering centre, m is the dimension of the data object, x_j and C_{ij} are the j-th attribute values of x and C_i. Next, the center of each cluster is updated as the mean of all data points in that cluster, and then the distance between each data point and the updated center is recalculated and assigned again. The above process is repeated until the center of the cluster no longer changes significantly. In the final clustering result obtained, the data points are assigned to clusters and each data point has the minimum distance from the center of the cluster to which it belongs. The network scanning organization discovery algorithm based on GCN and K-means is shown as Algorithm 1.

Algorithm 1 The network scanning organization discovery algorithm based on GCN and K-means

Input:
Network Scanning Attribute Graph: G,
Number of Clusters: k.
Output:
Cluster Centres and Labels.
Process:

IF-IDF processing of text attributes in G.
One-Hot processing of categorical attributes in G.
Normalization processing of numerical attributes in G.
Graph embedding: $Z = GCN(G)$.
Initializing clustering centres: k nodes are randomly selected from Z as initial clustering centres.
for each z_i in Z **do**
    Calculate the distances of z_i from all cluster centres.
    Assign z_i to the nearest cluster.
    Update the clustering centre: The cluster centre vector is the average of all node vectors in the cluster.
end for

GCN+Spectral. Spectral clustering is an unsupervised clustering algorithm based on graph theory and spectral theory. Unlike traditional clustering algorithms (e.g., K-means), Spectral clustering is not constrained by the convex shape of the data or the cluster size and is therefore effective in dealing with non-convex shaped and complex structured datasets. The method achieves the clustering task by building a similarity graph structure between data objects and utilizing spectral analysis of the graph. The core idea is to consider the samples in a dataset as nodes in a graph, build the edges of the graph by the similarity

between the samples, and then use the spectral structure of the graph to group the data. Spectral clustering uses matrices that have been extensively studied in spectral graph theory, so-called graph Laplacian functions, of both "unnormalized" and "normalized" types. The non-normalized Laplace matrix is defined as:

$$L = D - W \tag{8}$$

where D is the degree matrix and W is the weight matrix. There are two general types of normalized Laplace matrices, both of which are closely related and are defined as follows:

$$L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \tag{9}$$

$$L_{rw} = D^{-1} L = I - D^{-1} W \tag{10}$$

L_{sym} is a symmetric matrix, L_{rw} is closely related to a randomized tour, and these two matrices are closely related to each other. The network scanning organization discovery algorithm based on GCN and Spectral is shown as Algorithm 2.

Algorithm 2 The network scanning organization discovery algorithm based on GCN and Spectral

Input:
Network Scanning Attribute Graph: G,
Number of Clusters: k.
Output:
Cluster Centres and Labels.
Process:

IF-IDF processing of text attributes in G.
One-Hot processing of categorical attributes in G.
Normalization processing of numerical attributes in G.
Graph embedding: $Z = GCN(G)$.
for each pair of nodes i, j in Z **do**
    Calculate the Euclidean distance S_{ij} between nodes i and j.
    Taking S_{ij} as the similarity.
end for
Constructing the similarity matrix S.
Calculate the degree matrix: $D_{ii} = \sum_i S_{ij}$.
Calculate the Laplace matrix: $L = D - W$.
Calculate the normalized Laplace matrix: $L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$.
Feature decomposition of L_{sym}.
Select the feature vectors corresponding to the smallest k non-zero eigenvalues to compose a new matrix U.
Input U into the K-means algorithm to cluster k clusters.

GCN+DBSCAN. The DBSCAN algorithm is a density-based clustering method whose core mechanism relies on the estimation of the minimum density level. The model defines two main parameters: the radius ϵ of the neighborhood to be used for any distance measure and the minimum number of neighbors $minPts$ located within that radius. If the number of neighbors of an object within the given radius exceeds the threshold, the object is identified as a core point. The network scanning organization discovery algorithm based on GCN and Spectral is shown in Algorithm 3.

Algorithm 3 The network scanning organization discovery algorithm based on GCN and DBSCAN

Input:
Network Scanning Attribute Graph: G,
Neighborhood Radius: ϵ,
Minimum Number of Neighbors: $minPts$.
Output:
Cluster Centres and Labels.
Process:

IF-IDF processing of text attributes in G.
One-Hot processing of categorical attributes in G.
Normalization processing of numerical attributes in G.
Graph embedding: $Z = GCN(G)$.
Initialize the unvisited points set N.
while N is not empty **do**
    Choose a point p from N.
    Calculate the neighborhood $N_\epsilon(p)$ of p.
    if $|N_\epsilon(p)| < minPts$ **then**
        Mark p as the noise point.
    else
        Create a new cluster C and add p to C.
        Add all points in $N_\epsilon(p)$ to the set S.
        while S is not empty **do**
            Choose a point q from S.
            if q is not visited **then**
                Remove q from N.
                Calculate the neighborhood $N_\epsilon(q)$.
                if $|N_\epsilon(p)| \geq minPts$ **then**
                    Add all points in $N_\epsilon(q)$ to the set S.
                end if
            end if
            if q is not part of any cluster **then**
                Add q to C.
            end if
        end while
    end if
end while

4. Evaluation

In this section, we conduct a systematic experiment to evaluate the effectiveness of the proposed network scanning organization method based on GCN. We evaluate the identification accuracy of the proposed method on a given dataset, which will be described in detail later.

4.1. Experiment Setup

Datasets. The dataset used in this paper is derived from traffic logs from honeypots deployed on several VPSs, with the contents of the logs stored in an Elasticsearch database. We deploy 50 honeypots across eight geographic regions to collect comprehensive cyber bot traffic for pattern analysis. Each honeypot runs on a virtual machine configured with two CPU cores and 2 GB RAM, with the deployment sharing 1 GB/s network bandwidth. Individual honeypots host three to seven services selected from a pool of 21 available protocols, creating varied service combinations such as HTTP, FTP, SSH, HTTP, MySQL, RTSP, and HTTP, Telnet, MongoDB. We identified these 19 authentic organizations (labels) by comparing data from online mapping platforms such as FOFA. There are “quake, censys, stretchoid, shadowserver, internet-census, binaryedge, shodan,

intrinsicsec, research-scanner, rapid7, recyber, criminalip, onyphe, internettl, zoomeye, ipip, fofa, internet-measurement, and cyber". The six organizations with the highest number of logs are: quake, censys, stretchoid, shadowserver, internet-census, binaryedge, with respective proportions of 25%, 23.5%, 15.7%, 8.4%, 6.2%, and 5.1%. We select 53,091 labeled traffic log data of these organizations as the training set and 56,930 log data as the test set to verify the effectiveness of the method in this paper.

Parameter Settings. The GCN we constructed comprises three neural network layers. The first layer is a graph convolutional layer with an output channel size of 16 and a ReLU activation function. The second layer is a Dropout layer with a rate of 0.5 to prevent the model from overfitting. The third layer is the graph convolutional layer, which serves as the model's output layer, with the output dimension corresponding to the number of classes. During the model training phase, we employed the Adam optimizer alongside the cross-entropy loss function.

Evaluation Metrics. We selected the following six evaluation metrics to evaluate the effectiveness of the method: Silhouette Index (SI), Calinski–Harabasz Index (CHI), Davies–Bouldin Index (DBI), Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI), where SI, CHI, and DBI are internal metrics that measure the closeness of a data point after it has been assigned to the cluster group to which it belongs. ACC, NMI, ARI, and external metrics, a measure of the clustering algorithm in the supervised case, comparing the clustering results with known results.

4.2. Evaluation Results

The methods compared in this paper are K-means, Spectral, and DBSCAN clustering algorithms. We processed the data in the following three ways:

- **non-Graph:** Instead of constructing a node attribute graph, the relevant attributes of the host devices and the relevant attributes of the network connections are spliced together, and the whole is used as a feature.
- **Graph-G:** Construct a node attribute graph with host devices as nodes and network connections as relationships.
- **Graph-LG:** Based on the construction of the node attribute graph, the feature matrix is smoothed using the Laplace smoothing technique.

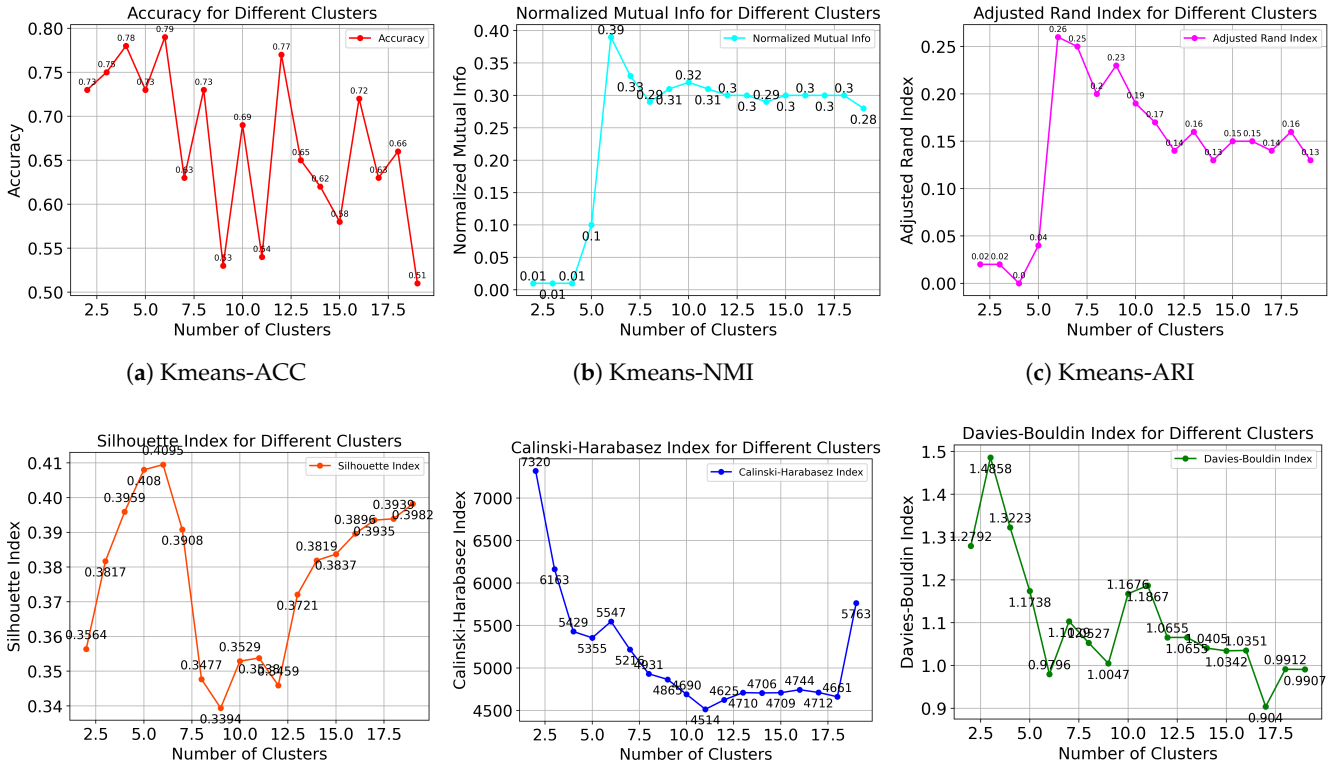
We used these three processing methods to conduct experiments with the three clustering algorithms. The results on the six evaluation metrics are shown in Table 2.

Table 2. Results of six evaluation metrics in organizational discovery based on clustering algorithm. Bold indicates the best result among the three clustering methods, while * denotes the best result for that metric among all methods.

Method		ACC	NMI	ARI	SI	CHI	DBI
non-Graph	**K-means**	**0.7913**	**0.3948**	**0.2639**	0.4095	**5546.8930**	0.9796
	Spectral	0.7104	0.3133	0.2461	**0.5213**	3920.2341	**0.8721**
	DBSCAN	0.4233	0.2015	0.1971	0.1358	379.3614	1.3172
Graph-G	**K-means**	**0.8012**	**0.5542**	0.4179	0.7167	**23,914.2313**	**0.4496**
	Spectral	0.7728	0.4854	**0.4253**	**0.7741**	9176.2561	0.6182
	DBSCAN	0.5144	0.2448	0.2175	0.3142	1052.6348	2.9178
Graph-LG	**K-means**	**0.8341** *	**0.6074** *	**0.5652** *	**0.7828** *	**31,034.5827** *	**0.4235** *
	Spectral	0.7543	0.5188	0.4732	0.7182	10,378.5347	0.6017
	DBSCAN	0.5214	0.3437	0.2831	0.2876	987.4192	3.7206

The experimental results show that the K-means algorithm has the best clustering effect, with results of 0.8341, 0.6074, 0.5652, 0.7828, 31034.5827, and 0.4235 on the six

metrics ACC, NMI, ARI, SI, CHI, and DBI, respectively. Meanwhile, we can see the best experimental results after constructing the attribute graph and smoothing it with Laplace features. The worst results are obtained when features are directly spliced together without constructing an attribute graph. This does not take into account the correlations between network scanning behaviors. The experimental results after smoothing with the Laplace filter are better than the unsmoothed ones, indicating the effectiveness of the feature smoothing technique for graph feature extraction and clustering. The ACC, NMI, and ARI results of the K-means clustering algorithm for different numbers of clusters are shown in Figure 3.



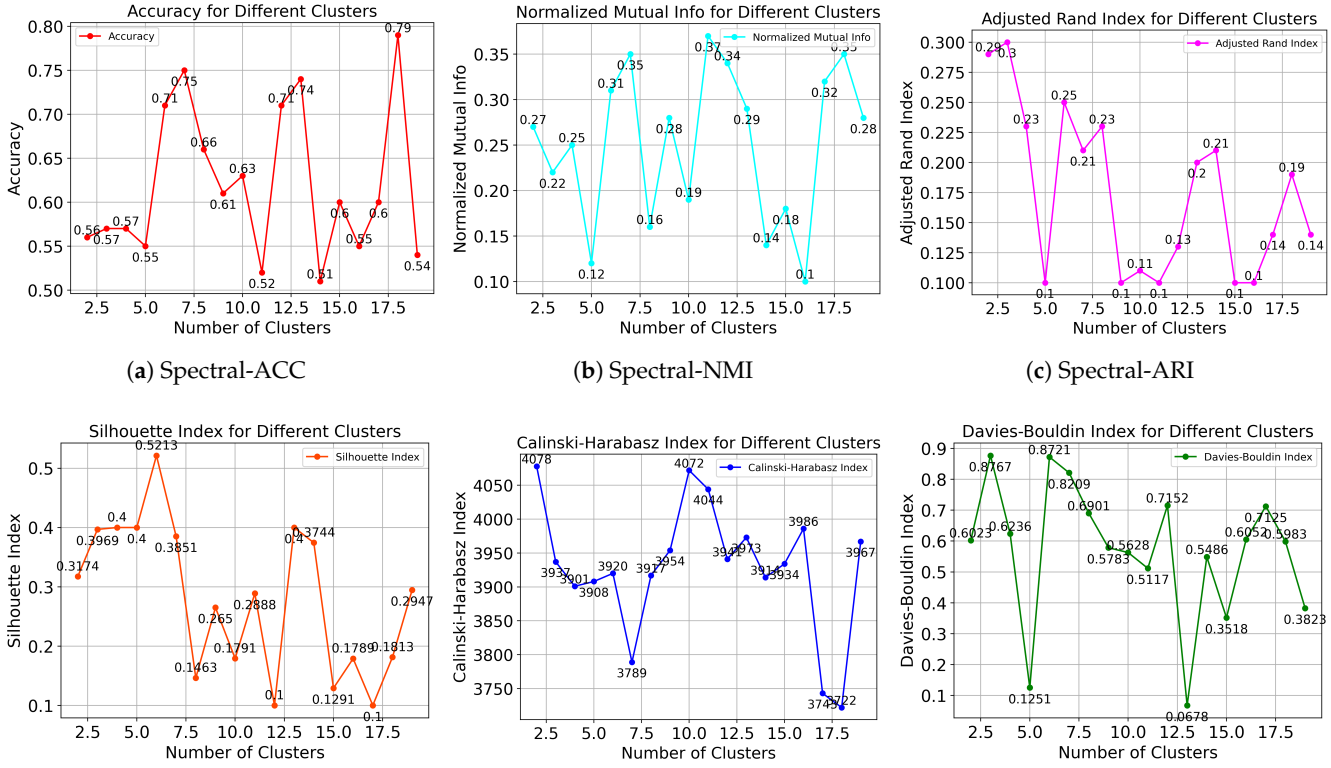
(**a**) Kmeans-ACC (**b**) Kmeans-NMI (**c**) Kmeans-ARI

(**d**) Kmeans-SI (**e**) Kmeans-CHI (**f**) Kmeans-DBI

Figure 3. Results for different number of clusters in K-means clustering.

It can be seen from the figure that the ACC, NMI, ARI, and SI values of K-means clustering are maximized and the DBI value is minimized at cluster number 6, which further proves that the K-means algorithm performs optimally. The ACC, NMI, and ARI results of the Spectral clustering algorithm for different numbers of clusters are shown in Figure 4. We visualized and analyzed the clustering results as shown in Figure 5.

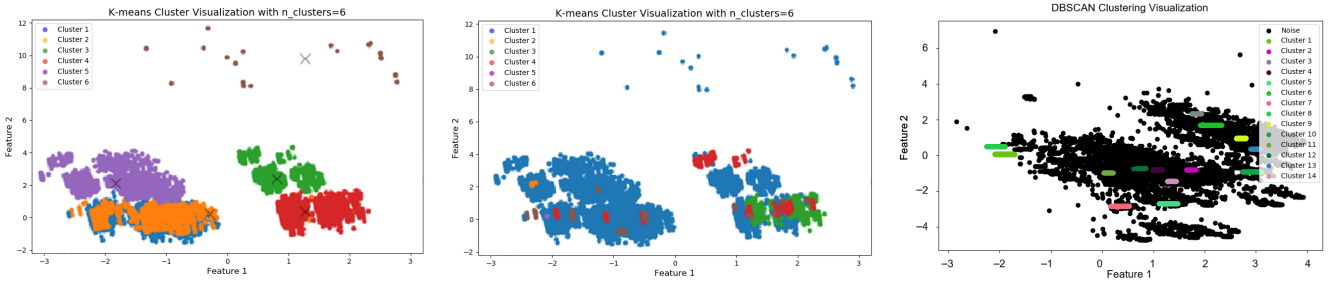
As can be seen from the figure, four clusters can be clearly distinguished in the K-means clustering effect plot, with a total of four organizations accounting for more than 10% of the total, namely quake (29.8%), censys (28%), stretchoid (18.6%) and shadoserver (10%). The two algorithms, K-means and DBSCAN, differ in the boundary distinctness and tightness of the clusters. Although DBSCAN's clusters present clearer and tighter boundaries in the graph, K-means exhibit higher values for evaluation metrics such as ACC, NMI, and ARI in comparison. This may be due to the fact that K-means is better adapted to the convex distribution or homogeneity of the data on the dataset used in this paper, and therefore the correlation metrics are all better than DBSCAN.



(**a**) Spectral-ACC (**b**) Spectral-NMI (**c**) Spectral-ARI

(**d**) Spectral-SI (**e**) Spectral-CHI (**f**) Spectral-DBI

Figure 4. Results for different number of clusters in Spectral clustering.



(**a**) K-means (**b**) Spectral (**c**) DBSCAN

Figure 5. Visualisation of clustering results.

5. Discussion

5.1. Limitations

We propose a network scanning organization discovery method based on GCN, which models the correlations between network scanning behaviors to identify organizations. Although the preliminary results of the paper confirm the effectiveness of the method, there are still some issues that need to be discussed. Firstly, in this paper, we use TF-IDF to vectorize the text features which may cause some limitations as it is difficult to adequately represent the semantics in the text. Second, although the K-means algorithm clusters well, this algorithm usually assumes that the clusters are roughly the same size and density, which may not be applicable to complex, dynamically changing scenarios.

5.2. Future Work

For the issue of representing textual attributes of nodes, we will consider adopting more advanced natural languages processing techniques such as Word2Vec, BERT, or word

embedding methods such as GloVe, combined with GCN models, so as to capture and utilize the deep semantic information in textual data more effectively in the subsequent research. For the dynamically changing scenarios of network scanning organizations, we will explore dynamic clustering algorithms to better adapt to the evolution of organizational structure over time and more accurately reflect the real-time state of cyberspace scanning organizations.

6. Conclusions

In this paper, we propose a network scanning organization discovery method, which constructs an attribute graph to analyze correlations between scanning behaviors. Unlike existing methods that mostly focus on identifying illegal scanning behaviors, we focus more on identifying the organizations behind these scanning behaviors. We construct an exhaustive dataset of network scanning behaviors, including 1,201,797 pieces of data. Through experimental analysis of this dataset, we can effectively identify the organization behind these network scanning behaviors. This is important for reducing the threat of attacks on network assets and maintaining network security.

Author Contributions: Conceptualization, P.X. and C.W.; methodology, L.D. and P.X.; software, L.D.; validation, C.W. and C.H.; formal analysis, C.H. and P.X.; investigation C.W. and P.X.; resources, J.W.; data curation, J.W.; writing—original draft preparation P.X.; writing—review and editing C.W. and L.D.; visualization, C.H.; supervision, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China (Grant No. 2022YFB3102902).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is restricted and not publicly available; it can be obtained by contacting the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Hao, H.; Xu, C.; Zhang, W.; Yang, S.; Muntean, G.M. Joint task offloading, resource allocation, and trajectory design for multi-uav cooperative edge computing with task priority. *IEEE Trans. Mob. Comput.* **2024**, *23*, 8649–8663. [CrossRef]
2. Hao, H.; Xu, C.; Zhang, W.; Yang, S.; Muntean, G.M. Task-Driven Priority-Aware Computation Offloading Using Deep Reinforcement Learning. *IEEE Trans. Wirel. Commun.* **2025**, *24*, 8114–8128. [CrossRef]
3. Camelo, M.; Soto, P.; Latré, S. A General Approach for Traffic Classification in Wireless Networks Using Deep Learning. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 5044–5063. [CrossRef]
4. Jenefa, A.; Sam, S.; Nair, V.; Thomas, B.G.; George, A.S.; Thomas, R.; Sunil, A.D. A Robust Deep Learning-based Approach for Network Traffic Classification using CNNs and RNNs. In Proceedings of the 2023 4th International Conference on Signal Processing and Communication (ICSPC), Coimbatore, India, 23–24 March 2023; pp. 106–110. [CrossRef]
5. Marchetta, P.; Pescapé, A. DRAGO: Detecting, quantifying and locating hidden routers in Traceroute IP paths. In Proceedings of the 2013 Proceedings IEEE INFOCOM, Turin, Italy, 14–19 April 2013; pp. 3237–3242. [CrossRef]
6. Sherry, J.; Katz-Bassett, E.; Pimenova, M.; Madhyastha, H.V.; Anderson, T.; Krishnamurthy, A. Resolving IP aliases with prespecified timestamps. In Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, Melbourne, Australia, 1–3 November 2010; IMC '10, pp. 172–178. [CrossRef]
7. Marchetta, P.; Persico, V.; Pescapè, A. Pythia: Yet another active probing technique for alias resolution. In Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies, Santa Barbara, CA, USA, 9–12 December 2013; CoNEXT '13, pp. 229–234. [CrossRef]
8. Yang, B.; Sun, S.; Li, J.; Lin, X.; Tian, Y. Traffic flow prediction using LSTM with feature enhancement. *Neurocomputing* **2019**, *332*, 320–327. [CrossRef]

9. Vikram, A.; Mohana. Anomaly detection in Network Traffic Using Unsupervised Machine learning Approach. In Proceedings of the 2020 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 10–12 June 2020; pp. 476–479. [CrossRef]
10. Marwah, M.; Arlitt, M. Deep Learning for Network Traffic Data. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 14–18 August 2022; KDD '22, pp. 4804–4805. [CrossRef]
11. Elsheikh, M.; Shalaby, M.; Sobh, M.A.; Bahaa-Eldin, A.M. Deep Learning Techniques for Intrusion Detection Systems: A Survey and Comparative Study. In Proceedings of the 2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), Cairo, Egypt, 27–28 September 2023; pp. 1–9. [CrossRef]
12. Liu, H.; Lang, B. Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. *Appl. Sci.* **2019**, *9*, 4396. [CrossRef]
13. Kurnala, V.; Naik, S.A.; Surapaneni, D.C.; Reddy, C.B. Hybrid Detection: Enhancing Network & Server Intrusion Detection Using Deep Learning. In Proceedings of the 2023 IEEE 5th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA), Hamburg, Germany, 7–8 October 2023; pp. 248–251. [CrossRef]
14. Ashiku, L.; Dagli, C. Network Intrusion Detection System using Deep Learning. *Procedia Comput. Sci.* **2021**, *185*, 239–247. [CrossRef]
15. He, P.; Zhu, J.; He, S.; Li, J.; Lyu, M.R. Towards Automated Log Parsing for Large-Scale Log Data Analysis. *IEEE Trans. Dependable Secur. Comput.* **2018**, *15*, 931–944. [CrossRef]
16. Landauer, M.; Skopik, F.; Wurzenberger, M.; Rauber, A. System Log Clustering Approaches for Cyber Security Applications: A Survey. *Comput. Secur.* **2020**, *92*, 101739. [CrossRef]
17. Zhong, M.; Zhou, Y.; Chen, G. A Security Log Analysis Scheme Using Deep Learning Algorithm for IDSs in Social Network. *Secur. Commun. Networks* **2021**, *2021*, 5542543. [CrossRef]
18. Ramachandran, S.; Agrahari, R.; Mudgal, P.; Bhilwaria, H.; Long, G.; Kumar, A. Automated Log Classification Using Deep Learning. *Procedia Comput. Sci.* **2023**, *218*, 1722–1732. [CrossRef]
19. Bhuyan, M.H.; Bhattacharyya, D.; Kalita, J. Surveying Port Scans and Their Detection Methodologies. *Comput. J.* **2011**, *54*, 1565–1581. [CrossRef]
20. Mirza, A. Port Scanning: Techniques, Tools and Detection. *engrXiv* 2023, *preprint*. [CrossRef] [PubMed]
21. Wang, Y.; Zhang, J. DeepPort: Detect Low Speed Port Scan Using Convolutional Neural Network. In Proceedings of the International Conference on Bio-Inspired Computing: Theories and Applications, Beijing, China, 2–4 November 2018; Qiao, J., Zhao, X., Pan, L., Zuo, X., Zhang, X., Zhang, Q., Huang, S., Eds.; Springer: Singapore, 2018; pp. 368–379.
22. Algaolahi, A.Q.M.; Hasan, A.A.; Sallam, A.; Sharaf, A.M.; Abdu, A.A.; Alqadi, A.A. Port-Scanning Attack Detection Using Supervised Machine Learning Classifiers. In Proceedings of the 2021 1st International Conference on Emerging Smart Technologies and Applications (eSmarTA), Sana'a, Yemen, 10–12 August 2021; pp. 1–5. [CrossRef]
23. Aksu, D.; Ali Aydin, M. Detecting Port Scan Attempts with Comparative Analysis of Deep Learning and Support Vector Machine Algorithms. In Proceedings of the 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), Ankara, Turkey, 3–4 December 2018; pp. 77–80. [CrossRef]
24. Sun, H.; He, F.; Huang, J.; Sun, Y.; Li, Y.; Wang, C.; He, L.; Sun, Z.; Jia, X. Network Embedding for Community Detection in Attributed Networks. *ACM Trans. Knowl. Discov. Data* **2020**, *14*, 1–25. [CrossRef]
25. Boden, B.; Ester, M.; Seidl, T. Density-Based Subspace Clustering in Heterogeneous Networks. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases Machine Learning and Knowledge Discovery in Databases, Nancy, France, 15–19 September 2014; Calders, T., Esposito, F., Hüllermeier, E., Meo, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 149–164.
26. Cui, G.; Zhou, J.; Yang, C.; Liu, Z. Adaptive Graph Encoder for Attributed Graph Embedding. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual, 6–10 July 2020; ACM: New York, NY, USA, 2020; KDD '20. [CrossRef]
27. Richter, P.; Berger, A. Scanning the Scanners: Sensing the Internet from a Massively Distributed Network Telescope. In Proceedings of the Internet Measurement Conference, Amsterdam, The Netherlands, 21–23 October 2019; IMC '19, pp. 144–157. [CrossRef]
28. Li, X.; Azad, B.A.; Rahmati, A.; Nikiforakis, N. Good Bot, Bad Bot: Characterizing Automated Browsing Activity. In Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 24–27 May 2021; pp. 1589–1605. [CrossRef]
29. Mazel, J.; Strullu, R. Identifying and characterizing ZMap scans: A cryptanalytic approach. *arXiv* **2019**, arXiv:1908.04193. [CrossRef]