N-STGAT：Spatio-Temporal Graph Neural Network Based Network Intrusion Detection for Near-earth Remote Sensing

Yalu Wang , Jie Li , Wei Zhao * , Zhijie Han , Hang Zhao , Lei Wang , Xin He

Posted Date: 22 May 2023

doi: 10.20944/preprints202305.1455.v1

Keywords: near-earth remote sensing; network intrusion; temporal features; spatio-temporal graph attention network

Article

N-STGAT：Spatio-Temporal Graph Neural Network Based Network Intrusion Detection for Near-Earth Remote Sensing

Yalu Wang [1], Jie li [2], Wei Zhao [3,*], Zhijie Han [4], Hang Zhao [5], Lei Wang [6] and Xin He [7]

[1] School of Computer and Information Engineering, Henan University, Kaifeng 475004, China
[2] School of Intelligent Engineering, Zhengzhou University of Aeronautics, Zhengzhou 450046, China
[3] School of Miami, Henan University, Kaifeng 475004, China
[4] International Joint Laboratory of Intelligent Network Theory and Key technology of Henan, Henan University, Kaifeng 475004, China
[5] State Key Laboratory of Crop Stress Adaptation and Improvement, Henan University, Kaifeng 475004, China
[6] College of agriculture, Henan University, Kaifeng 475004, China
[7] School of Software, Henan University, Kaifeng 475004, China
* Correspondence: henuzhao@vip.henu.edu.cn

Abstract: With the rapid development of Internet of Things (IoT)-based near-earth remote sensing technology, the problem of network intrusion for near-earth remote sensing systems has become more complex and large-scale. Therefore, it is essential to seek an intelligent, automated, and robust network intrusion detection method. In recent years, network intrusion detection methods based on graph neural networks (GNNs) have been proposed. However, there are still some practical issues with these methods. For example, they have not taken into consideration the characteristics of near-earth remote sensing systems, the state of the nodes, and the temporal features. Therefore, this article analyzes the characteristics of existing near-earth remote sensing systems and proposes a spatio-temporal graph attention network (N-STGAT) that considers the state of nodes, and applies them to the network intrusion detection of near-earth remote sensing systems and validates the effectiveness of the proposed method on the latest flow-based dataset.

Keywords: near-earth remote sensing; network intrusion; temporal features; spatio-temporal graph attention network

1. Introduction

Remote sensing technology has undergone rapid development in recent years. High-resolution satellite remote sensing technology has provided convenience for meteorology, terrain surveying, military, agriculture, and other fields [1,2]. However, satellite remote sensing has limitations in some remote sensing fields that require ultra-high precision and multimodal information due to distance and equipment constraints [3]. To address these issues, the development of Internet of Things (IoT)-based near-earth remote sensing technology has made significant progress. Near-earth remote sensing technology plays a crucial role in agriculture analysis, mining, water monitoring [4], and other fields. The near-earth remote sensing technology consists of a vertical structure as shown in Figure 1, which includes a large number of IoT devices [5], such as weather balloons, airplanes, drones, and sensors. These devices are connected to the base station via an IoT network and collect various remote sensing information, providing hardware support for subsequent data analysis. Figure 2 shows the deployment of near-earth remote sensing devices for monitoring crops in farmland.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

Figure 1. Schematic diagram of near-ground remote sensing system based on the Internet of Things.

The deployment of a large number of Internet of Things (IoT) devices is required for the near-ground remote sensing system, and most of these devices are outdoors, exposed to the elements, and unsupervised, which makes them vulnerable to physical or network-based intrusions. Once a remote sensing device is compromised, it may send incorrect data or attack other devices, causing problems for the near-earth remote sensing system. Therefore, it is essential to deploy a network intrusion detection system specifically designed for the near-ground remote sensing system.

Figure 2. Example of near-ground remote sensing technology in agriculture.

A network intrusion detection system is a system that can quickly respond when an IoT device is invaded, and its core is a network intrusion detection method. There are mainly two types of network intrusion detection methods: feature-based detection and machine learning-based detection [6]. Feature-based detection methods require a pre-set of attack features, and the captured data packets are compared with these features. The accuracy of this detection method depends entirely on the set of features, and this method cannot respond well to new attacks. With the development and application of machine learning in recent years, machine learning-based detection methods have been developed. However, these methods directly use data flow information for identification [7-11]. Although these methods can effectively solve the problem of feature dependence and have good detection effects on new attacks, the identification accuracy is not high, and they are difficult to apply to practical networks. Recently, researchers have studied the newer subfield of machine learning, graph neural networks (GNNs), and proposed intrusion detection methods based on GNNs [6,12,13].

 doi:10.20944/preprints202305.1455.v1

However, these methods have low identification accuracy in multi-classification tasks and cannot be effectively applied to near-Earth remote sensing systems.

Due to the fact that meteorological balloons, aircraft, drones, and other devices have their own operating systems, the near-Earth remote sensing system composed of these devices can provide more information. Therefore, this article considers the characteristics of the near-Earth remote sensing system and proposes a spatio-temporal graph attention network (N-STGAT) that considers the node status, applying spatio-temporal graph neural networks [14] to network intrusion detection in the near-Earth remote sensing system. To the best of our knowledge, this is the first time that spatio-temporal graph neural networks have been applied to network intrusion detection in the near-Earth remote sensing system.

The contributions of this article are as follows:

Expanding the latest IoT network intrusion detection dataset by incorporating the status of the node, which better reflects the situation of the near-Earth remote sensing system and enables better evaluation of the proposed method.

The article proposes for the first time the application of spatio-temporal graph neural networks to network intrusion detection in near-Earth remote sensing systems, and further improves the method to better align with the characteristics of near-Earth remote sensing systems, providing a new solution for network intrusion detection.

The proposed method is validated on the extended dataset and compared with some of the recent effective IoT network intrusion detection methods. The results show that the proposed method outperforms other methods.

The other chapters of this article are as follows: Chapter 2 introduces the research contents of various scholars in the field of network intrusion detection methods. Chapter 3 first briefly introduces some information about GNN and LSTM, and then explains how to extend the latest network intrusion detection dataset. Chapter 4 provides a detailed explanation of the proposed network intrusion detection system. Chapter 5 describes the experimental methods and results. Chapter 6 concludes the article.

2. Related Work

In recent years, many researchers have developed machine learning-based methods for network intrusion detection. However, most of these methods directly use data flow information for identification.

Casas et al. [15]. proposed an unsupervised network intrusion detection system called UNIDS, which employs a subspace clustering-based and multi-evidence accumulation-based unsupervised outlier detection method to detect unknown network attacks. It does not require the use of any features, labeled traffic, or training to detect various types of network attacks, such as DoS/DDoS, probing attacks, worm propagation, buffer overflow, illegal access to network resources, etc. The proposed method's effectiveness was demonstrated through experiments using KDD99 and real traffic data from two operational networks. However, the KDD99 dataset is quite old and may not reflect the attack characteristics present in modern IoT networks.

In [16], authors proposed a hybrid anomaly mitigation framework for IoT using fog computing to ensure faster and accurate anomaly detection. The framework employs signature- and anomaly-based detection methodologies for its two modules, respectively. The BoT-IoT dataset was used for experimentation and the effectiveness of the proposed method was ultimately validated. Results showed accuracy of 99% in binary and multi-class classification problems, with at least 97% average recall, average precision, and average F1 score.

The two methods mentioned above use traditional machine learning techniques for network intrusion detection, such as decision trees, MDAE, LSTM, random forests, and XGBoost. These methods directly train on the dataset without considering the graph topology information, resulting in each training data being relatively singular and unable to fully explore the information in each data. Therefore, these methods have limited capability in detecting complex network attacks, such as

 doi:10.20944/preprints202305.1455.v1

Botnet attacks [17], distributed port scans [18] or DNS Amplification attacks [19], which require a more global network view and traffic.

Leichtnam et al. [20]. introduced a unique graph representation called security objects' graphs that linked events of different kinds and allowed for a rich description of the analyzed activities. They proposed an unsupervised learning approach based on auto-encoders to detect anomalies in these graphs. The authors hypothesized that auto-encoders could build a relevant model of the normal situation based on the rich vision provided by security objects' graphs. To validate their approach, they applied it to the CICIDS2017 dataset and showed that their unsupervised method performed as well as, or even better than, many supervised approaches.

Due to the good performance of GNN on graph-structured data and the fact that a network is a natural graph, in recent years, some scholars have applied GNN to intrusion detection systems.

Wai Weng Lo et al. [6]. proposed the E-GraphSAGE architecture based on GraphSAGE, which allows capturing the edge features and topology information of the graph for network intrusion detection in the Internet of Things. Experiments on four latest NIDS benchmark datasets show that the method has achieved very good effects on binary classification problems, but not on multi-classification problems. When constructing the graph, this method directly uses the ip address and port of the network flow as nodes and the network flow as edges. When this method uses GraphSAGE for neighborhood aggregation, it may cause some neighborhoods to be aggregated multiple times and the whole neighborhood to be aggregated, which will affect the identification accuracy.

Cheng Q et al. [12]. proposed an Alert-GCN framework that uses graph convolutional networks (GCN) to associate alerts coming from the same attack for the prediction of the alert system. Then adopted the DARPA99 data set for experiments, achieving good results in the experiment. However, this method used a custom similarity method to define the edges in the graph when constructing an alarm-related graph, which will cause the neighbors of any node in the graph to be highly similar to themselves. This article recognizes that the graph constructed by this method will cause overfitting in the graph neural network.

Evan Caville et al. [13]. proposed an intrusion and anomaly detection method called Anomal-E based on E-GraphSAGE, which utilized edge features and graph topology in a self-supervised process. Then, they validated the proposed method using the NF-UNSW-NB15-v2 and NF-CSE-CIC-IDS2018-v2 datasets. Ultimately, they achieved results with at least 97.5% accuracy and at least 92% F1-Score. Additionally, the article was the first to apply a self-supervised graph neural network solution to network intrusion detection.

Although the above method applies graph neural networks to network intrusion detection, it does not take into considering the characteristics of the remote sensing system, node status, and temporal information. Therefore, this article combines GAT and LSTM to better utilize the spatial information of the graph and the temporal information of the data flow, increasing the reliability of network intrusion detection.

3. Background

This article argues that when identifying data flows, the state of the nodes should be considered simultaneously. When a network attack occurs, the state of the attacking node is significantly different from that of a normal node. Therefore, considering the state of the nodes is very beneficial for identifying attacks. In addition, identifying attacks should also consider the topological relationship between data flows and the temporal information of data flows.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

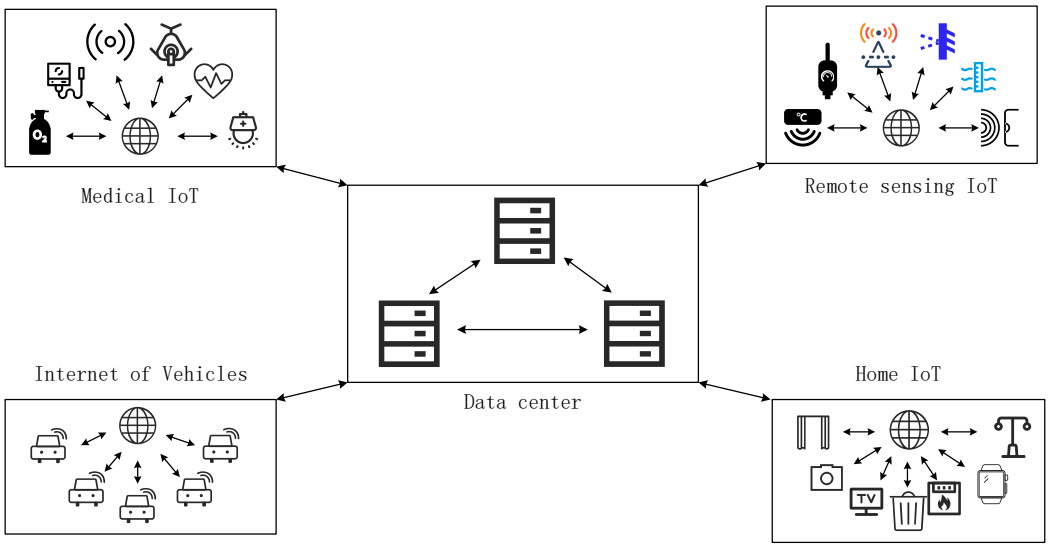


Figure 3. Characteristics of IoT structure.

Intrusion detection systems are generally deployed at the entry point in order to detect network intrusions. However, with the development of the IoT, more and more networks are structured as shown in Figure 3, which this article refers to as an "IoT domain". Such IoT domains are increasingly prevalent in industrial IoT systems, such as those used by companies like Apple, Huawei, and Xiaomi. Most of the devices in an IoT domain are monitored by a central server, which can obtain information about the status of the nodes. This situation is similar to that of a ground-based remote sensing system based on the IoT. As the situation where certain nodes within a domain are controlled to attack other nodes becomes more and more common, intrusion detection systems need to be deployed at the exit points of the domain to detect attacks on nodes.

3.1. GNN and LSTM

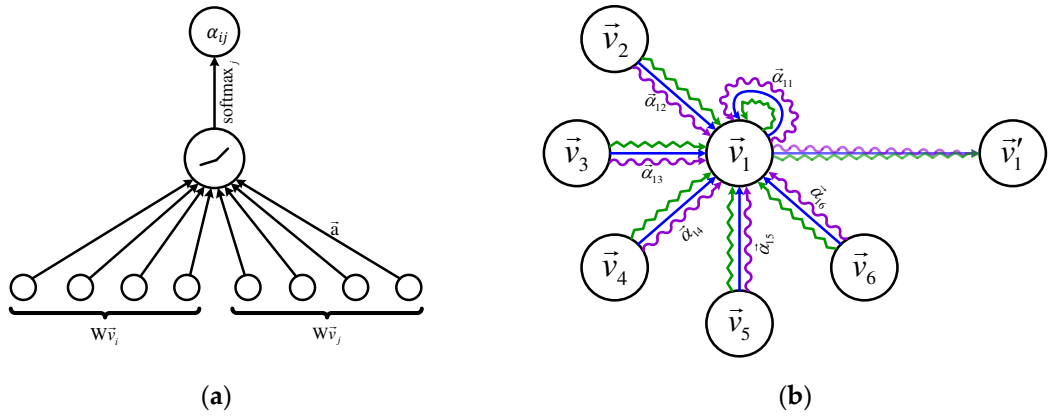
3.1.1. GNN

GNNs are a new type of neural network specifically designed to handle the properties of graphs. In recent years, GNNs have been widely applied in graph processing [21], networking [22], intelligent transportation [23], recommendation systems [24], and distributed computing [25], among other fields. A key feature of GNNs is that they can process non-Euclidean data and utilize topological structure data through message passing. When identifying nodes in a graph, GNNs can aggregate the data features of neighboring nodes, allowing them to influence each other [26]. This process is called embeddings [27]. There are many different types of GNNs, among which Graph Attention Network (GAT) [28], Graph Convolutional Network (GCN) [29], and GraphSAGE [30] are particularly popular among researchers.

GAT (Graph Attention Networks) is a type of graph neural network that incorporates attention mechanism to aggregate neighborhood nodes by computing attention coefficients. This enables GAT to capture the influence of different nodes within the neighborhood. GAT has two core components: computing the attention coefficients α_{ij} and the hidden layer features $\vec{v}_i'$ of nodes.

GAT is a type of graph neural network that incorporates attention mechanism to aggregate neighborhood nodes by computing attention coefficients. This enables GAT to capture the influence of different nodes within the neighborhood. GAT has two core components: computing the attention coefficients α_{ij} and the hidden layer features $\vec{v}_i'$ of nodes.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1



(**a**) (**b**)

Figure 4. (**a**) The calculation process of attention coefficient; (**b**) The calculation process of hidden layer features.

The calculation process for the attention coefficients α_{ij} between neighboring nodes is shown in Figure 4(a). The original GAT article mentions two types of neighborhoods: node neighbors (nodes directly connected to the current node) and full neighborhood (all nodes). In this article, we use node neighbors as the neighborhood to prevent the aggregation of a large amount of neighborhood node information that may cause the features to become blurry. Using node neighbors can also reduce time and space complexity. The calculation formula is:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{a}^T\left[W\vec{v}_i || W\vec{v}_j\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}(\vec{a}^T[W\vec{v}_i || W\vec{v}_k])\right)} \tag{1}$$

a is a single-layer feedforward neural network, which is parameterized by a weight vector $\vec{a} \in \mathbb{R}^{2F'}$, $W \in \mathbb{R}^{2F'\times F}$ is a shared linear transformation matrix, LeakyReLU is a nonlinear function, and $\mathcal{N}_i$ is the neighbor node of node i. The equation finally uses a $softmax$ function for normalization.

The calculation process of hidden layer features $\vec{v}_i'$ is shown in Figure 4(b). The calculation formula is:

$$\vec{v}_i' = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k W^k \vec{v}_j\right) \tag{2}$$

σ is a nonlinear function. Since GAT uses a multi-head attention mechanism, K represents the number of attention heads.

3.1.2. LSTM

LSTM is a special type of recurrent neural network (RNN) architecture used for processing and predicting sequence data [31]. It uses gate mechanisms to better capture long-term dependencies in time series data, making it perform better in handling such data. An LSTM network consists of a series of LSTM cells, each of which contains an input gate, a forget gate, and an output gate. The input gate and forget gate are used to control the flow of information, with the input gate determining which information should be updated in the memory cell and the forget gate deciding which information should be discarded from the memory cell. The output gate determines how much influence the memory cell should have on the current output. The core cell unit of LSTM is shown in Figure 5.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

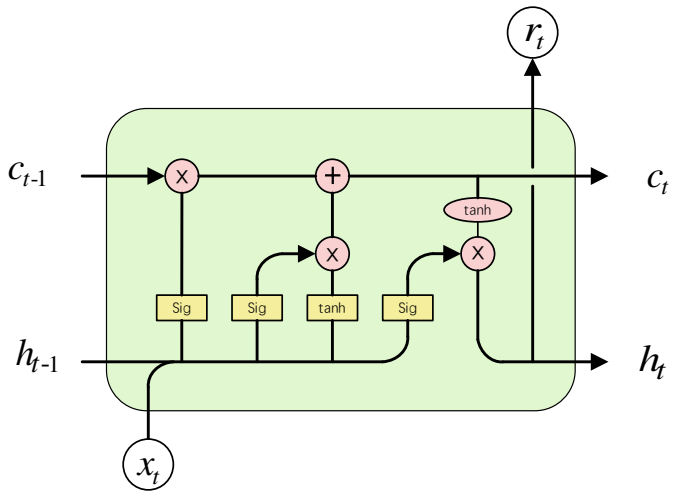


Figure 5. The core cell unit of LSTM.

3.2. Datasets

In order to describe the proposed method more clearly, it is necessary to first introduce the datasets used in this article. The datasets used in this article are NF-BoT-IoT-v2 and NF-ToN-IoT-v2[32], and their descriptions are shown in the following table:

Table 1. Basic information of the datasets.

Dataset	Release year	No. Classes	No. features	No. data	Benign ratio
NF-BoT-IoT-v2	2021	5	43	37,763,497	0.0 to 10.0
NF-ToN-IoT-v2	2021	10	43	16,940,496	3.6 to 6.4

These two datasets are improved based on the original datasets BoT-IoT 错误!未找到引用源。 and ToN-IoT [34], where the network data is integrated in a stream fashion to make a complete network behavior.

The features in the aforementioned datasets are referred to as flow features, denoted as FI in this article. As the article requires the node status information, 14 additional features were added to these two datasets based on the original datasets, including:

Table 2. Details of the newly added features.

Feature	Description
TIMESTAMP	The timestamp when the data flow is sent.
PROCESS_LOAD	1-minute load average
PROCESS_ID	Idle CPU percentage.
PROCESS_HI	Hard interrupt CPU percentage
PROCESS_US	User-space CPU percentage,
PROCESS_SY	Kernel-space CPU percentage
MEMORY_USED	Memory used ratio
MEMORY_BUFFER	Memory cache ratio
MEMORY_NETWORK	Memory used by network module ratio
NET_PAKAGES	Number of packets sent
NET_ BANDWIDTH_OUT	Network egress bandwidth
NET_TCP_CONNECTIONS	Number of TCP connections
DISK_READ	Disk read speed
DISK_WRITE	Disk write speed

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

This article refers to the features in Table 2 as node features, denoted as NI. With the above fields, the dataset is expanCded into two parts, with a total of 57 fields.

4. Method Description

4.1. Problem Definition

Suppose there are n nodes in a near-earth sensing system, defined as $N_1, N_2, \ldots\ldots, N_n$. The data flows sent by each node are defined in chronological order as v_i^k $(i \in [1,n], k \in [1,\infty))$, where k is the sequential number of the data flow.. The purpose of this article is to identify the attack type r_i^k of the data flow v_i^k sent by node N_i, as shown in Figure 6.

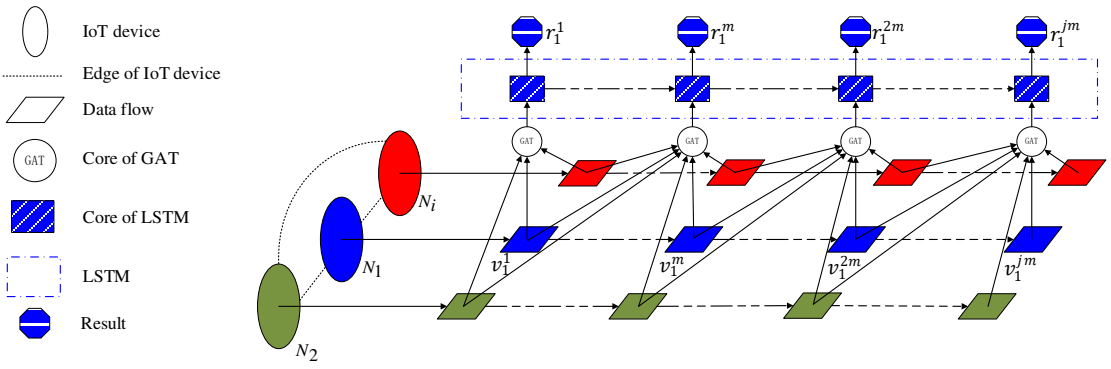


Figure 6. Problem Definition Structure.

To solve the aforementioned issues, this article proposes the N-STGAT network intrusion detection system. As the goal of this article is to solve network intrusion detection problems in time series, a spatiotemporal graph neural network combining GAT and LSTM is used, which has good performance in identifying data flows with spatiotemporal sequences.

Figure 7 illustrates the system flow of N-STGAT. First, the extended dataset is preprocessed, and then $\mathcal{G}$ for training and $\mathcal{G}_{test}$ for testing are generated based on the graph construction rules. Then, the $\mathcal{G}$ is fed into the training process of N-STGAT to obtain a trained model. Finally, the $\mathcal{G}_{test}$ is input into the model for data flow classification.

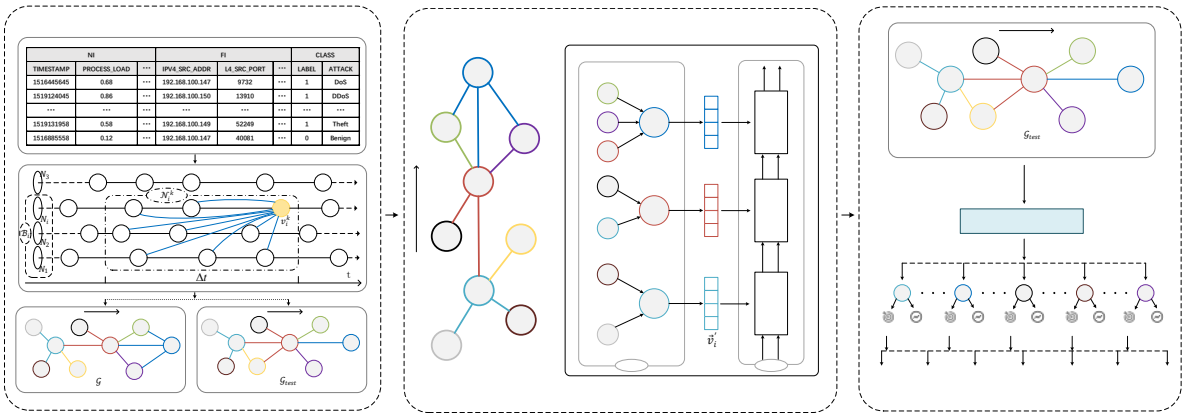


Figure 7. The workflow of the proposed N-STGAT intrusion detection system. First, the dataset is preprocessed, and a graph is constructed for training and testing based on time-axis relationships and similarity rules (left). N-STGAT is used to train the model on the training graph, and the trained model is output (middle). Finally, the generated testing graph is input into the trained model for intrusion detection classification (right).

4.2. Pre-processing and Graph Construction

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

To input the training data into GAT, the dataset needs to be transformed into a graph structure. Before constructing the graph, the dataset needs to be formatted to seamlessly apply to GAT. The flow-based dataset is collected through network monitoring tools, which capture and save packets that pass through nodes, switches, and routers. Then, the data packets are aggregated into flows using analysis tools. These datasets contain a wealth of important network information, including source and destination IP addresses, source and destination ports, byte counts, and other useful packet information. However, there are many data formats in the dataset that cannot be directly used for training, such as enumeration type, null value, and invalid value. Figure 8 shows the data processing process.

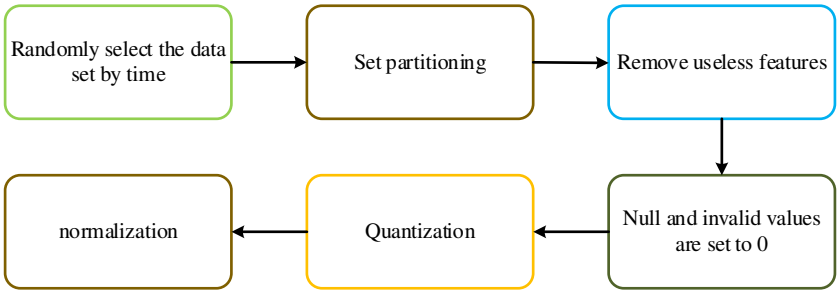


Figure 8. Data preprocessing process.

The dataset is grouped by the source IP address and each group is sorted by the time sequence of the data flows. In this way, the dataset is composed of thousands of chains of data flows. The data from time t to $(t + 7t_s/10)$ is selected as the training dataset, and the data from t_{test} to $(t_{test} + 3t_s/10)$ is selected as the testing dataset, as shown in Figure 6. Useless features such as IP addresses and ports are removed. There are also some invalid and empty values in the dataset, which are set to 0. Since the values of some fields in the dataset have a large range, quantification is needed. Finally, the remaining fields are normalized using L2 normalization.

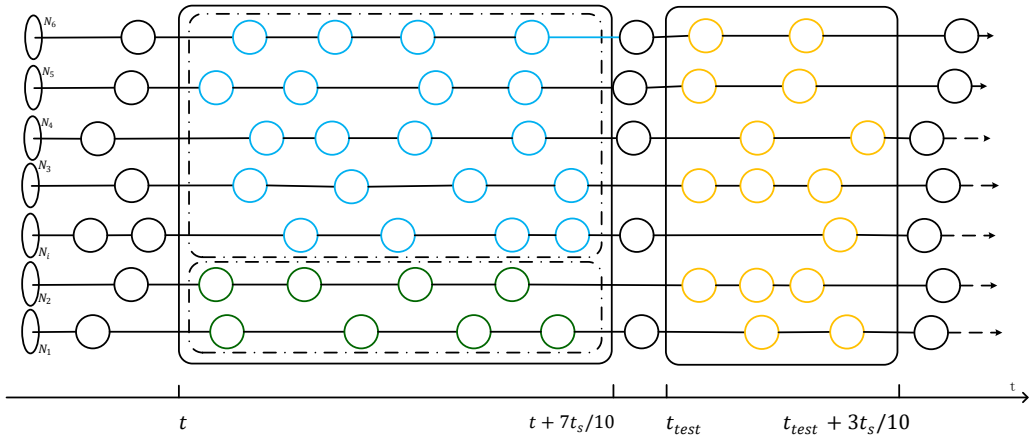


Figure 9. Random selection of datasets.

Once the datasets for training and testing are selected, it is necessary to build graphs based on specific rules. Graph construction is done to find neighbors for nodes, so knowing the neighbor-finding process for one node is enough to construct the entire graph. The process of finding neighbors for nodes can be divided into two parts: finding neighborhood $\mathcal{B}_i$ for node N_i and finding neighborhood $\mathcal{N}_i^k$ for node v_i^k.

$\mathcal{B}_i$ is obtained using the cosine similarity with the NI field of the dataset as shown in formula (3). The method is as follows: assuming the time when v_i^k exists is t_i^k, and the latest node before time t_i^k for any N_j is v_j^{lt}, calculate the cosine similarity $\cos(\theta)_{ij}^{klt}$ between v_i^k and v_j^{lt}, if $\cos(\theta)_{ij}^{klt}$ is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

greater than 0.7, then add N_j to $\mathcal{B}_i$. Here, x_i represents the i-th feature value in v_i^k's NI field, and y_i represents the i-th feature value in v_j^{lt}'s NI field.

$$\cos(\theta) = \frac{\sum_{i=1}^{n}(x_i \times y_i)}{\sqrt{\sum_{i=1}^{n}(x_i)^2} \times \sqrt{\sum_{i=1}^{n}(y_i)^2}} \tag{3}$$

In $\mathcal{B}_i$, all nodes v_j^l (including v_i^k) within $(t_i^k - \Delta t)$ to t_i^k are neighbors of v_i^k, and these nodes form the neighborhood $\mathcal{N}_i^k$ of v_i^k. Then, connect A and C to form an edge E, which is a directed edge from v_i^k to v_j^l. In this article, Δt is set to 3s when constructing the graph using the method described above, resulting in the graph structure shown in formula (4).

$$\begin{gathered} V = v_i^k \quad i \in [1, n], k \in [1, \infty) \\ E = e_{ij}^{kl} \quad l \in \left[t_i^k - \Delta t, t_i^k\right], j \in \mathcal{N}_i^k \\ \mathcal{G} \leftarrow V, E \end{gathered} \tag{4}$$

4.3. N-STGAT Training

Because the dataset contains two labels, "Label"and "Attack". The "Label"indicates whether the data flow is an attack or not, while the "Attack"label defines the type of the data flow. Therefore, the labels can be used for binary and multi-class classification training.

N-STGAT is described in Algorithm 1, and the pseudocode shows the process of integrating GAT and LSTM. In lines 3 to 6, GAT is used to calculate the hidden layer feature $\vec{v}_i^{k\prime}$. Then, in line 7, $\vec{v}_i^{k\prime}$ is assigned to x_i^k to participate in the LSTM calculation process. Lines 8 to 13 represent the LSTM calculation process. Finally, in line 16, the output C of LSTM is fed into a fully connected layer to obtain the final recognition result r_i^k.

	Algorithm 1: Pseudocode of the N-STGAT algorithm.
	Input: $\mathcal{G}(V, E)$ node features $\vec{v}_i^k$; GAT weight matrices W; non-linearity σ; LSTM weight matrices $W_f, W_l, W_C, W_o, b_f, b_l, b_C, b_o$; LSTM initialization C^0, h^0
	Output: node features r_i^k;
1	**for** i=1 to n do
2	**for** $k = 1$ to length(N_i) do
3	**for** $j = 1$ to length($\mathcal{N}_i^k$) do
4	$\alpha_i^{kj} = \frac{\cos(\theta)_i^{kj}(\exp(\text{LeakyReLU}(\vec{a}^T[W\vec{v}_i^k \| W\vec{v}^j])))}{\sum_{s\in\mathcal{N}_i}\cos(\theta)_i^{ks}(\exp(\text{LeakyReLU}(\vec{a}^T[W\vec{v}_i^k \| W\vec{v}^s])))}$
5	**end**
6	$\vec{v}_i^{k\prime} = \sigma(\sum_{s\in\mathcal{N}_i}\alpha_i^{ks}\mathrm{W}\vec{v}^s)$
7	$x_i^k \leftarrow \vec{v}_i^{k\prime}$
8	$f_i^k = \sigma(W_f \cdot [h_i^{k-1}, x_i^k] + b_f)$
9	$l_i^k = \sigma(W_l \cdot [h_i^{k-1}, x_i^k] + b_l)$
10	$\tilde{C}_i^k = tanh(W_C \cdot [h_i^{k-1}, x_i^k] + b_C)$
11	$C_i^k = f_i^k * C_i^{k-1} + l_i^k * \tilde{C}_i^k$
12	$o_i^k = \sigma(W_o \cdot [h_i^{k-1}, x_i^k] + b_o)$
13	$h_i^k = o_i^k * \tanh(C_i^k)$
14	**end**
15	**end**
16	$r_i^k = FC(h_i^k)$ //FC is fully connected layers

The algorithm uses only one layer of GAT model instead of multiple layers because multiple layers can make the features blur and not conducive to recognition. Specifically, we add the cosine

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

similarity $\cos(\theta)_i^{kj}$ between two nodes as a constraint to the calculation of attention coefficients in the attention coefficient calculation process in the 4th line, so that the graph information constructed in 4.2 can be better expressed.

Table 3. Hyperparameter values used in N-STGAT.

Hyperparameter	Values
No. Layers	1
No. Hidden	256
No. K	1
Learning Rate	$2e^{-3}$
Activation Func.	ReLU
Loss Func.	Cross-Entropy
Optimiser	Adam

For each dataset mentioned in the experiments, we conducted experiments with the hyperparameters shown in Table 3 to obtain the best training model. As mentioned above, we used a single-layer GAT model instead of a multi-layer one. To better represent node features, we expanded the original 53 features to 256 hidden layer features. GAT supports multi-head attention, but in this case, we only used single-head attention since we introduced cosine similarity as a constraint for attention coefficient calculation. We used ReLU as the activation function and did not use dropout rate. Cross-Entropy function was used to calculate the training loss, and Adam was used for backpropagation optimization with a learning rate set to 0.002.

4.4. N-STGAT Detection

In the previous section, the N-STGAT model $\mathcal{M}$ was obtained through model training. In the detection task, the test graph $\mathcal{G}_{test}$ is input into the model $\mathcal{M}$. The 16th line of Algorithm 1 uses a fully connected layer to recognize the specific classification. Whether it is a binary or multi-classification problem, the softmax function is used to calculate the classification in the final layer of the fully connected layer to obtain the exact classification result.

5. Experimental Evaluation

5.1. Evaluation Metrics

To evaluate the performance of N-STGAT, experiments were conducted using the datasets mentioned in Section 3.2. Due to the large size of these datasets, 10% of the datasets were selected using the method described in Section 4.2 for experimentation, with 70% of the experimental dataset used for training and 30% used for testing. Since only a portion of the dataset was used for experimentation, experiments were conducted multiple times for each algorithm and each dataset

In the experiments, the proposed method was compared with graph neural network-based methods (GAT, E-ResGAT, Anomal-E) and traditional machine learning methods (SVM, Random Forest). Therefore, evaluation metrics as shown in Table 4 were used to assess the performance of all methods. TP, FP, FN, and TN represent True Positive, False Positive, True Negative, and True Negative in the Confusion Matrix, respectively. These evaluation metrics provide a comprehensive comparison of the superiority of the proposed method.

Table 4. Performance comparison metrics for experiments.

Metric	Definition
Recall	$\frac{TP}{TP+FN}$
Precision	$\frac{TP}{TP+FP}$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

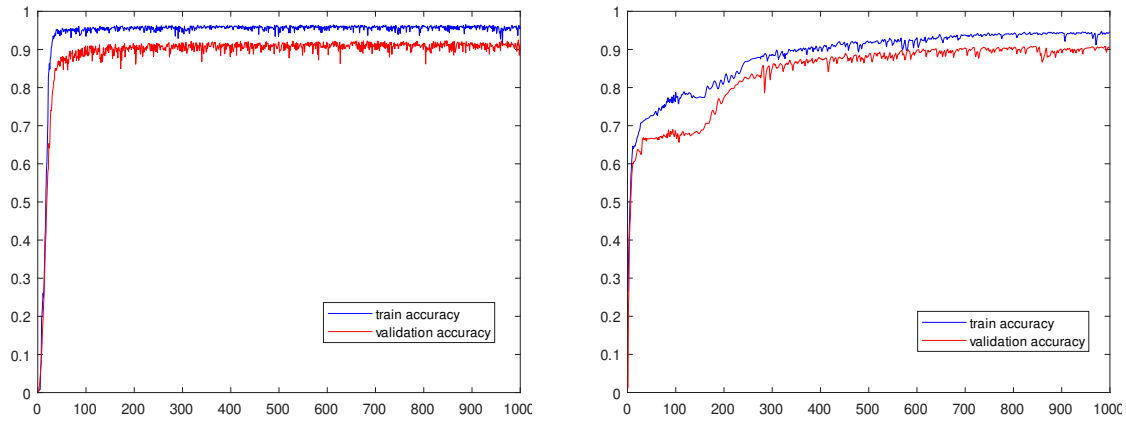
F1-Score	$2 \times \frac{Recall \times Precision}{Recall + Precision}$
Accuracy	$\frac{TP + TN}{TP + FP + FN + TN}$

5.2. Result

In the experiments, the accuracy and loss changes during the training process were first recorded, and then the recognition results for both binary and multi-class classification were evaluated based on the performance metrics in Section 5.1.

5.2.1. Loss and Accuracy Comparison in Training

The training accuracy changes for multi-class classification on different datasets are shown in Figure 10. The training accuracy of dataset NF-BoT-IoT-v2 quickly reached 90% before 20 epochs and remained stable above 95% after 100 epochs. Its validation accuracy was lower than the training accuracy throughout the entire training process, but also stabilized above 92% in the end. The training accuracy of dataset NF-ToN-IoT-v2 was slightly lower than that of dataset NF-BoT-IoT-v2, with continuous growth before 600 epochs and stability above 93% after 700 epochs. Its validation accuracy was also lower than the training accuracy throughout the entire training process, but stabilized above 90% in the end.

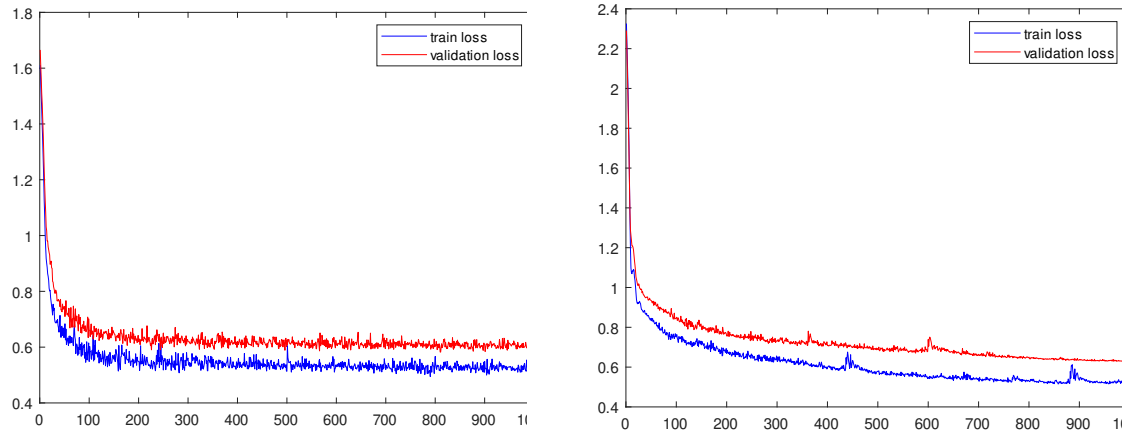


(**a**) (**b**)

Figure 10. The accuracy changes on different datasets. (**a**) Dataset NF-BoT-IoT-v2. (**b**) Dataset NF-ToN-IoT-v2.

The cross-entropy loss during the training process for multi-classification on different datasets is shown in Figure 11. For the dataset NF-BoT-IoT-v2, the training loss rapidly decreased to 0.6 before 100 epochs and stabilized around 0.4 after 200 epochs. The validation loss was higher than the training loss throughout the training process, but also stabilized at around 0.7 in the end. For the dataset NF-ToN-IoT-v2, the training loss was slightly worse than that of NF-BoT-IoT-v2, and it continued to decrease before 600 epochs and then stabilized at 0.4 after 700 epochs. The validation loss was higher than the training loss throughout the training process but also stabilized at 0.6 in the end.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1



(**a**) (**b**)

Figure 11. The loss changes on different datasets. (**a**) Dataset NF-BoT-IoT-v2. (**b**) Dataset NF-ToN-IoT-v2.

5.2.2. Binary Classification Results

The results of the binary classification experiment using the "Label" label of the dataset is shown in Table 5. In the binary classification task, the proposed method in this article outperforms the other 5 methods in terms of recall and accuracy, indicating that the proposed method has a better recognition accuracy. Additionally, the proposed method also performs well in precision, with results only 0.4% lower than E-GraphSAG in the NF-ToN-IoT-v2 dataset but higher than the other 5 methods. In terms of F1-score, the proposed method outperforms the other 5 methods, indicating better stability of the proposed method.

Table 5. Results of each method on binary classification.

DataSet	Algorithm	Recall	Precision	F1-Score	Accuracy
NF-BoT-IoT-v2	SVM	0.8485	0.9367	0.8904	0.8299
	Random Forest	0.8212	0.9151	0.8656	0.7923
	GAT	0.9013	0.9633	0.9313	0.8917
	E-GraphSAG	0.9615	0.9825	0.9719	0.9547
	Anomal-E	0.9412	0.9859	0.963	0.9412
	N-STGAT	**0.9812**	**0.9927**	**0.9869**	**0.9788**
NF-ToN-IoT-v2	SVM	0.7689	0.8991	0.8289	0.7415
	Random Forest	0.7368	0.9307	0.8224	0.7409
	GAT	0.8746	0.9724	0.9209	0.8776
	E-GraphSAG	0.9578	**0.9867**	0.972	0.9551
	Anomal-E	0.9461	0.9846	0.965	0.9441
	N-STGAT	**0.9755**	0.9827	**0.9791**	**0.9661**

5.2.3. Multiclass Classification Results

The results of the multi-class classification experiment using the "Attack" label of the dataset are shown in Table 6 and Table 7. It can be seen from the results that the proposed method has superiority in multi-class problems, with high recognition rates for each category of different datasets. On the NF-BoT-IoT-v2 dataset, the proposed method has a 3.53% - 20.49% improvement in Weighted Recall and 8.03% - 23.16% improvement in Weighted F1-Score compared to other methods. On the NF-ToN-

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 May 2023 doi:10.20944/preprints202305.1455.v1

IoT-v2 dataset, the proposed method has a 4.05% - 18.69% improvement in Weighted Recall and 7.17% - 17.78% improvement in Weighted F1-Score compared to other methods.

Table 6. Performance results for each algorithm over multiple classifications.

DataSet	Algorithm	Weighted Recall	Weighted F1-Score
NF-BoT-IoT-v2	SVM	0.7101	0.6948
	Random Forest	0.7719	0.7492
	GAT	0.7227	0.7021
	E-GraphSAG	0.8797	0.8461
	Anomal-E	0.865	0.8016
	N-STGAT	**0.915**	**0.9264**
NF-ToN-IoT-v2	SVM	0.7195	0.7514
	Random Forest	0.7786	0.7364
	GAT	0.7699	0.8163
	E-GraphSAG	0.8533	0.8204
	Anomal-E	0.8659	0.8425
	N-STGAT	**0.9064**	**0.9142**

Table 7. Recall results per class on the tow datasets.

Dataset	Algorithm	Per class Recall									
		Benign	**RN**	**DDos**	**Dos**	**Theft**					
NF-BoT-IoT-v2	SVM	0.7154	0.6148	0.8412	0.8649	0.7225					
	Random Forest	0.8205	0.8415	0.7451	0.5748	0.8148					
	GAT	0.8952	0.6715	0.8216	0.7469	0.7149					
	E-GraphSAG	0.8756	0.8912	0.82465	0.9051	0.8694					
	Anomal-E	0.9049	0.8648	0.7903	**0.9417**	0.8795					
	N-STGAT	**0.9506**	**0.9241**	**0.8786**	0.9207	**0.9513**					
		Benign	RN	DDos	Dos	Backdoor	Injection	MITM	Password	Scanning	XSS
NF-ToN-IoT-v2	SVM	0.7147	0.5792	0.8106	0.7129	0.6129	0.6792	0.8059	0.7138	0.846	0.7816
	Random Forest	0.8703	0.7126	0.6109	0.5498	0.7159	0.8619	0.7469	0.8759	0.7482	0.6874
	GAT	0.8761	0.7454	0.8418	0.7923	0.8219	0.7619	0.8242	0.6958	0.8716	0.9015
	E-GraphSAG	0.9418	0.8819	**0.9112**	0.8109	0.8846	0.7805	0.8759	0.8904	0.8513	0.8927
	Anomal-E	0.8042	**0.9229**	0.8496	0.8217	0.9036	**0.9158**	0.8176	0.9013	0.7219	0.9014
	N-STGAT	**0.9712**	0.9013	0.9013	**0.8735**	**0.9213**	0.9016	**0.9254**	**0.9186**	**0.8619**	**0.9208**

6. Conclusions

This article proposes a spatio-temporal graph attention network (N-STGAT) that considers node states and applies it to network intrusion detection in near-Earth remote sensing systems. A method is proposed for constructing graphs based on node state and the temporal characteristics of data flow, where data flow are viewed as nodes in the graph, and edges between nodes are constructed based on cosine similarity and time series features. A spatio-temporal graph neural network combining GAT and LSTM is applied to intrusion detection systems. Finally, experiments are conducted using the latest flow-based network intrusion detection dataset, and the proposed method is compared with existing methods to demonstrate its superiority and feasibility.

Author Contributions: Conceptualization, Yalu Wang and Jie Li; Data curation, Hang Zhao; Formal analysis, Zhijie Han; Investigation, Zhijie Han; Methodology, Yalu Wang and Wei Zhao; Project administration, Jie Li; Resources, Yalu Wang; Software, Zhijie Han; Supervision, Xin He; Validation, Yalu Wang, Jie Li and Wei Zhao; Writing – original draft, Yalu Wang; Writing – review & editing, Wei Zhao, Lei Wang and Xin He. All authors have read and agreed to the published version of the manuscript.

Funding: Please add: This research received no external funding.

Data Availability Statement: This article references four datasets: NF-BoT-IoT-v2, NF-ToN-IoT-v2, BoT-IoT, and ToN-IoT. The datasets NF-BoT-IoT-v2 and NF-ToN-IoT-v2 can be found at the following link: https://staff.itee.uq.edu.au/marius/NIDS_datasets/. The link for the BoT-IoT dataset is: https://research.unsw.edu.au/projects/bot-iot-dataset. The link for the ToN_IoT dataset is: https://research.unsw.edu.au/projects/toniot-datasets.

Acknowledgments: This work was supported by the Key Science and Technology Project of Henan Province (201300210400).

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

References

1. D. Wang, J. Zhang, B. Du, G. -S. Xia and D. Tao, "An Empirical Study of Remote Sensing Pretraining," in IEEE Transactions on Geoscience and Remote Sensing, vol. 61, pp. 1-20, 2023, Art no. 5608020, doi: 10.1109/TGRS.2022.3176603.
2. Goswami, A.; Sharma, D.; Mathuku, H.; Gangadharan, S.M.P.; Yadav, C.S.; Sahu, S.K.; Pradhan, M.K.; Singh, J.; Imran, H. Change Detection in Remote Sensing Image Data Comparing Algebraic and Machine Learning Methods. Electronics 2022, 11, 431. https://doi.org/10.3390/electronics11030431
3. Sun X, Zhang Y, Shi K, et al. Monitoring water quality using proximal remote sensing technology[J]. Science of The Total Environment, 2022, 803: 149805. https://doi.org/10.1016/j.scitotenv.2021.149805
4. Chen J, Chen S, Fu R, et al. Remote sensing big data for water environment monitoring: current status, challenges, and future prospects[J]. Earth's Future, 2022, 10(2): e2021EF002289. https://doi.org/10.1029/2021EF002289
5. Li J, Hong D, Gao L, et al. Deep learning in multimodal remote sensing data fusion: A comprehensive review[J]. International Journal of Applied Earth Observation and Geoinformation, 2022, 112: 102926. https://doi.org/10.1016/j.jag.2022.102926
6. W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher and M. Portmann, "E-GraphSAGE: A Graph Neural Network based Intrusion Detection System for IoT," NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 2022, pp. 1-9, doi: 10.1109/NOMS54207.2022.9789878.
7. H. He, X. Sun, H. He, G. Zhao, L. He and J. Ren, "A Novel Multimodal-Sequential Approach Based on Multi-View Features for Network Intrusion Detection," in IEEE Access, vol. 7, pp. 183207-183221, 2019, doi: 10.1109/ACCESS.2019.2959131.
8. Lawal, M.A.; Shaikh, R.A.; Hassan, S.R. An Anomaly Mitigation Framework for IoT Using Fog Computing. Electronics 2020, 9, 1565. https://doi.org/10.3390/electronics9101565
9. Sarhan, M., Layeghy, S., Moustafa, N., Portmann, M. (2021). NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. In: Deze, Z., Huang, H., Hou, R., Rho, S., Chilamkurti, N. (eds) Big Data Technologies and Applications. BDTA WiCON 2020 2020. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 371. Springer, Cham. https://doi.org/10.1007/978-3-030-72802-1_9
10. Kumar P, Gupta G P, Tripathi R. An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks[J]. Computer Communications, 2021, 166: 110-124. https://doi.org/10.1016/j.comcom.2020.12.003
11. Churcher, A.; Ullah, R.; Ahmad, J.; ur Rehman, S.; Masood, F.; Gogate, M.; Alqahtani, F.; Nour, B.; Buchanan, W.J. An Experimental Analysis of Attack Classification Using Machine Learning in IoT Networks. Sensors 2021, 21, 446. https://doi.org/10.3390/s21020446
12. Q. Cheng, C. Wu and S. Zhou, "Discovering Attack Scenarios via Intrusion Alert Correlation Using Graph Convolutional Networks," in IEEE Communications Letters, vol. 25, no. 5, pp. 1564-1567, May 2021, doi: 10.1109/LCOMM.2020.3048995.
13. Caville E, Lo W W, Layeghy S, et al. Anomal-E: A self-supervised network intrusion detection system based on graph neural networks[J]. Knowledge-Based Systems, 2022, 258: 110030. https://doi.org/10.1016/j.knosys.2022.110030
14. Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, Zhaoqi Wang; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 6272-6281. https://openaccess.thecvf.com/content_ICCV_2019/html/Huang_STGAT_Modeling_Spatial-Temporal_Interactions_for_Human_Trajectory_Prediction_ICCV_2019_paper.html

15. Casas P, Mazel J, Owezarski P. Unsupervised network intrusion detection systems: Detecting the unknown without knowledge[J]. Computer Communications, 2012, 35(7): 772-783. https://doi.org/10.1016/j.comcom.2012.01.016
16. Lawal, M.A.; Shaikh, R.A.; Hassan, S.R. An Anomaly Mitigation Framework for IoT Using Fog Computing. Electronics 2020, 9, 1565. https://doi.org/10.3390/electronics9101565
17. G. Vormayr, T. Zseby and J. Fabini, "Botnet Communication Patterns," in IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2768-2796, Fourthquarter 2017, doi: 10.1109/COMST.2017.2749442.
18. Monowar H. Bhuyan, D.K. Bhattacharyya, J.K. Kalita, Surveying Port Scans and Their Detection Methodologies, The Computer Journal, Volume 54, Issue 10, October 2011, Pages 1565–1581, https://doi.org/10.1093/comjnl/bxr035
19. Kambourakis, G., Moschos, T., Geneiatakis, D., Gritzalis, S. (2008). Detecting DNS Amplification Attacks. In: Lopez, J., Hämmerli, B.M. (eds) Critical Information Infrastructures Security. CRITIS 2007. Lecture Notes in Computer Science, vol 5141. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-89173-4_16
20. Leichtnam, L., Totel, E., Prigent, N., Mé, L. (2020). Sec2graph: Network Attack Detection Based on Novelty Detection on Graph Structured Data. In: Maurice, C., Bilge, L., Stringhini, G., Neves, N. (eds) Detection of Intrusions and Malware, and Vulnerability Assessment. DIMVA 2020. Lecture Notes in Computer Science(), vol 12223. Springer, Cham. https://doi.org/10.1007/978-3-030-52683-2_12
21. Hao J, Liu J, Pereira E, et al. Uncertainty-guided graph attention network for parapneumonic effusion diagnosis[J]. Medical Image Analysis, 2022, 75: 102217. https://doi.org/10.1016/j.media.2021.102217
22. Jiang W. Graph-based deep learning for communication networks: A survey[J]. Computer Communications, 2022, 185: 40-54. https://doi.org/10.1016/j.comcom.2021.12.015
23. Jiang W, Luo J. Graph neural network for traffic forecasting: A survey[J]. Expert Systems with Applications, 2022: 117921. https://doi.org/10.1016/j.eswa.2022.117921
24. Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20). Association for Computing Machinery, New York, NY, USA, 639–648. https://doi.org/10.1145/3397271.3401063
25. P. Sun, Z. Guo, J. Wang, J. Li, J. Lan, Y. Hu, Deepweave: Accelerating job completion time with deep reinforcementlearning-based coflow scheduling, International Joint Conferences on Artificial Intelligence (2021) 3314–3320. https://www.ijcai.org/proceedings/2020/0458.pdf
26. Xu K, Hu W, Leskovec J, et al. How powerful are graph neural networks?[J]. arXiv preprint arXiv:1810.00826, 2018. https://doi.org/10.48550/arXiv.1810.00826
27. H. Cai, V. W. Zheng and K. C. -C. Chang, "A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications," in IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 9, pp. 1616-1637, 1 Sept. 2018, doi: 10.1109/TKDE.2018.2807452.
28. Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[J]. arXiv preprint arXiv:1710.10903, 2017. https://doi.org/10.48550/arXiv.1710.10903
29. Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016. https://doi.org/10.48550/arXiv.1609.02907
30. Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs[J]. Advances in neural information processing systems, 2017, 30. https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf
31. Smagulova, K., James, A.P. A survey on LSTM memristive neural network architectures and applications. Eur. Phys. J. Spec. Top. 228, 2313–2324 (2019). https://doi.org/10.1140/epjst/e2019-900046-x
32. Sarhan, M., Layeghy, S. & Portmann, M. Towards a Standard Feature Set for Network Intrusion Detection System Datasets. Mobile Netw Appl 27, 357–370 (2022). https://doi.org/10.1007/s11036-021-01843-0
33. Koroniotis N, Moustafa N, Sitnikova E, et al. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset[J]. Future Generation Computer Systems, 2019, 100: 779-796. https://doi.org/10.1016/j.future.2019.05.041
34. Moustafa N. A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets[J]. Sustainable Cities and Society, 2021, 72: 102994. https://doi.org/10.1016/j.scs.2021.102994