Research Article

MAE-CAD: An IP-Based Core Network Asset Discovery Technology Based on Multiple Autoencoders

Yijing Wang1,2, Peng Chang1, Han Wang1, Yong Ding1, and Rui Sun1

1Institute of Information Engineering, Chinese Academy of Sciences, Beijing 1000864, China
2School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100021, China

Correspondence should be addressed to Yijing Wang; wangyijing@iie.ac.cn

Received 17 December 2021; Accepted 27 January 2022; Published 12 March 2022

Academic Editor: Zhili Zhou

With the continuous development of the network, the number of network assets continues to increase. Despite the convenience diversified network assets bring, it also poses new challenges to IP-based network asset management. Traditional asset discovery technologies mainly analyze network traffic, and detect relevant information (operating system, running software, etc.) of IP-based assets through methods such as active discovery, passive discovery, and discovery methods based on cyberspace search engines. These methods assign the same weight to all network IP-based network assets, and it is difficult to effectively analyze diversified network assets. In this paper, we propose the concept of IP-based core network assets, and collect the data of the relevant network assets based on this concept. Then, we construct a dataset and establish feature engineering for data preprocessing. As there is currently no relevant IP-based core network asset detection method, we propose an IP-based core network asset discovery technology based on pretraining of multiple autoencoders, MAE-CAD. The results show that our method can achieve 95.74% in Acc and 95.04% in F1 in the experimental environment (Acc = 98.11% and F1 = 97.16% in the actual network environment because of duplicate samples). In addition, MAE-CAD has excellent robustness. In an environment where the proportion of data is extremely unbalanced, when the IP-based core network asset data in the training set only accounts for 1/200 (0.5%), MAE-CAD can still obtain 92.91% in Acc and 91.57% in F1.

1. Introduction

With the development of network technology, more and more social activities rely on the network. According to China Internet Network Information Center (CNNIC) statistics, as of July 2021, China has 1.011 billion network users, and the number of various network assets continues to increase [1]. Many diversified network assets provide various services for people’s production and life, which brings great convenience for people, and at the same time poses higher requirements for cyber security [2–4]. People are more concerned about network security than ever before, such as privacy security, property security, etc. Before formulating various targeted protection measures for network assets, how to effectively identify the diversified network assets in the network, especially the network assets with high priority and high weight, is very important. Therefore, comprehensive network asset discovery is crucial for network managers to effectively manage assets, and it is also valuable for threat analysis and the establishment of a more complete network asset protection mechanism.

Traditional network asset discovery refers to the discovery of the operating system version, software version, printer model, and other asset information behind the IP or domain name through network data such as traffic and company registration information to provide information for other subsequent activities [5–7]. There are three main types of network asset discovery technologies: active discovery [8], passive discovery [7] and discovery based on network security search engines [9]. However, with the continuous development of networks, such as the further improvement of 5G and IPv6 technologies, the scale of information is also increasing, and network assets are constantly increasing. These assets often have different
importance under different conditions. When the information security management department conducts asset discovery, sometimes it only needs to detect specific core asset information. The traditional asset discovery technologies are basically used to discover asset information such as operating system version and software version in the Internet, which have certain limitations in diversified scenes and cannot meet personalized asset discovery.

In this paper, we divide the network assets in units of IP. In different circumstances, the assets behind different IPs have different priorities and importance. In addition, after artificially giving a definition of the importance of IP-based network assets, we refer to the network assets with high-priority and high-importance as IP-based core network assets. In contrast, low-priority and low-importance network assets are IP-based noncore network assets. Generally, we need to discover other IP-based core network assets in a region after given some network asset samples. A typical scene is that an asset management unit in a region needs to find out the online network assets in the region to facilitate subsequent operations. For instance, managers need to find out certain specific network assets within their regions and determine their effective value and operating status to protect and update them in time. This requires an effective and different method from traditional asset discovery techniques. However, there is currently no relevant detection technology.

With the current network environment as the background premise, we conduct research on IP-based core network asset discovery under the premise of establishing network asset specifications. We cooperate with network service providers to collect information about IP-based network assets in the Internet, and provide an offline network asset profiling technology based on multiple autoencoders, MAE-CAD.

The contributions of this paper are as follows:

1. We propose the definition of IP-based core network assets in specific scenes. And, based on this concept, we take the IP-based network assets of the government agencies as an example to collect a related IP-based core network asset dataset. In this dataset, the IP-based core network asset data scale reaches 100,000, and the IP-based noncore network asset data scale reaches 1 million. Under the premise of privacy protection and security, we desensitize the data and disclose the data source to support research in related fields.

2. We conduct feature engineering for IP-based network asset feature information to perform feature preprocessing. Methods such as data similarity fitting, feature merging, and missing feature complement are used for feature unification. Then, the features are vectorized and normalized, and the attributes of different feature spaces are transferred to a continuous space suitable for feature processing.

3. We propose an IP-based core network asset discovery technology based on neural network pre-training, MAE-CAD, which can automatically further extract and perform abstract fitting from the features constructed in feature engineering. First, multiple unbalanced AutoEncoders (AEs) [10] are used to construct a neural network, where the encoder has more neurons. This neural network is pretrained using unsupervised techniques with unlabeled network asset data. The training method is similar to Stacked AutoEncoder (SAE) [11, 12], the hidden output of the previous AE is the input of the next AE. Then, a neural network detector is composed of encoders in multiple AEs and a softmax classification layer, and it fine-tunes parameters based on the labeled data.

4. Based on the MAE-CAD, we implement a prototype system and verify it in experiments. Experiments show that our method has excellent performance. Compared with other methods (Bayes, SVM, etc.), MAE-CAD can obtain Acc = 94.49% in accuracy and F1 = 93.71% in F1 value. In addition, in the circumstance of data imbalance, that is, the ratio of IP-based core network asset data and IP-based noncore network asset data in the training set is 1:200, MAE-CAD can still obtain 92.91% in Acc and 91.57% in F1. This shows that MAE-CAD has excellent robustness. In the actual network environment, MAE-CAD can even reach Acc = 98.41% and F1 = 97.62% due to repeated feature samples in IP network asset data.

The remainder of this paper is organized as follows. Section 2 introduces the related work. In Section 3, we analyze the features of IP-based core network assets. Subsequently, Section 4 is the methodology and we introduce the composition of MAE-CAD. In Section 5, we evaluate our method and show the relevant experimental results. Finally, we discuss and summarize in Section 6 and Section 7, respectively.

2. Related Work

In this paper, the IP-based core network asset discovery, our study belongs to the field of asset discovery, and the technology used is related to neural networks. At present, the mainstream research methods for asset discovery are active discovery, passive discovery, and discovery based on network security search engines. We, respectively, introduce three aspects of related work.

2.1. Active Discovery Technology. Active discovery technology of network assets can be divided into discovery technologies based on network layer, transport layer, and application layer. The technology based on the network layer mainly uses the characteristics of the Internet Control Message Protocol (ICMP) to detect the information of the target host [13]. For instance, Arkin et al. [8] conduct a coarse-grained detection of the operating system based on the difference in the default TTL value restored by different operating systems for ICMP messages. The technology based on the transport layer mainly uses the Transmission Control Protocol (TCP) [14]. Based on its reliable transmission,
connection-oriented, byte stream, and other characteristics, TCP can be used for the identification of asset information such as the open port of the target host and the port service operating system. For instance, some fields in the TCP options can be used as the discrimination basis to identify the operating system, because the default values of the fields in the TCP corresponding to different operating systems are different [15]. The technology based on the application layer mainly uses Hypertext Transfer Protocol (HTTP) or Hypertext Transfer Protocol over SecureSocket layer (HTTPS) [5,6]. The explorer actively constructs a specific request packet, and analyzes the required network asset information based on the content of the returned response packet. For instance, the server field in the response packet may contain the current host’s operating system and system service information, and the HTML source code and image file data of the body in the packet may contain information such as the web service and device type used by the target host.

Vermeer et al. [5] construct a framework to systematize asset discovery technology. They extract asset discovery techniques from the latest academic literature in the field of security and networking, and put them into a systematic framework. This provides researchers and practitioners with the opportunity to discover and identify more assets than traditional technologies.

In general, the explorer can use the active discovery technology of network assets to perform flexible detection according to the needs, but this requires the explorer to actively interact with the target object, so the speed is slow. And, the target object may be aware of it, and then take measures to prohibit detection or return an error message.

2.2. Passive Discovery Technology. The technology of passive discovery of network assets is to deploy detectors at network exit nodes to passively collect traffic data flowing through Refs. [16–18]. Then, analyze certain fields in the traffic data to obtain the network asset information of the target object. After collecting traffic data, the analysis method is similar to active discovery, and there are also three technologies based on network layer, transport layer, and application layer. However, as the proportion of encrypted traffic (such as HTTPS traffic) continues to increase, the efficiency of passive discovery technology at the application layer continues to decline. Because the traffic collected by passive discovery technology is encrypted at the application layer, it is difficult for the explorer to obtain effective information.

The decentralized nature of traditional Industrial Control System (ICS) traffic and the lack of traditional network equipment capabilities make it difficult for standard passive discovery techniques to detect ICS. Wedgbury et al. [19] introduce an overview and understanding of passive ICS discovery, and provide an experimental result to demonstrate the performance of existing passive asset discovery tools in an ICS environment where port mirroring technology is not universally supported. Mavrikas et al. [7] proposed two tools to realize automatic and completely passive discovery of the host and its operating system (OS), and proved that machine learning can improve detection performance in passive OS detection. The test results performed on real ICS network data show the effectiveness of the proposed method.

In general, the passive discovery technology of network assets discovers and analyzes various types of network asset information in the same way as active discovery. It also discovers varied network asset information by analyzing the fields in the network layer, transport layer, and application layer protocol response packets. Passive discovery has the characteristics of fast speed and wide range, and the explorer does not interact with the target object during the discovery process, there is no risk of exposure, and it does not affect the normal work of the target object. However, for specific network assets, passive discovery technology has the disadvantage of low effectiveness. Especially as encrypted traffic continues to replace plaintext traffic in the network, the practicality of passive asset detection technology at the application layer is also declining.

2.3. Discovery Technology Based on Cyberspace Search Engines. The asset discovery technology based on cyberspace search engines mainly uses search engines to explore equipment and service information in the entire cyberspace [20,21]. Cyberspace search engines are different from traditional search engines such as Google, Baidu, Bing and other comprehensive information search engines. They are mainly aimed at detecting online IP-based devices on the Internet, such as cameras, routers, printers, and even Industrial control equipment and nuclear industry equipment in the industrial control network, such as Shodan [22] and ZoomEye [23].

In general, the asset discovery technology based on cyberspace search engines can discover more detailed network asset information that cannot be discovered through active and passive discovery, and the detection speed is faster. However, the disadvantage is that it is impossible to detect the network asset information in the enterprise LAN, and it is difficult to detect specific network assets, and cannot meet the personalized detection requirements. For instance, the IP-based core network assets in this paper cannot be directly detected by cyberspace search engines because of the need to formulate specific detection strategies.

2.4. Analysis and Summary. The current asset discovery technologies mainly focus on identifying asset information such as operating systems and software versions in the network, using online identification based on network traffic, which can achieve excellent results in the corresponding asset discovery scenes. Besides various solutions for asset management using existing asset discovery technologies, excellent results can also be achieved in specific scenes. However, these methods have certain limitations in the diversified asset identification and core asset management scenes within the enterprise, and it is difficult to perform flexible and personalized IP-based network asset identification, such as the IP-based core asset identification scene in this article. Therefore, we need to find a detection technology that can effectively discover IP-based
core network assets after core assets are defined in a specific scene.

3. Analysis of the Features of IP-Based Core Network Assets

In this section, we analyze the features of IP-based core network assets in a specific scene and introduce the data preprocessing method.

3.1. Scene Analysis. We focus on the identification and discovery of IP-based core network assets. In this paper, we refer to all devices that can be controlled online based on IP as network assets, and divide these network assets into IP units. In a region, there are often a variety of different network assets. These assets can be divided into different asset groups from different perspectives, such as classification according to traffic, function, and service life. According to different regulations, we can flexibly assign different priorities and importance to various network assets, so that some of them are called IP-based core network assets.

In order to better manage network assets and formulate targeted asset protection measures, managers need to discover IP-based core assets in relevant regions. As shown in Figure 1, we show the conceptual diagram of the distribution of IP-based core network assets and IP-based noncore network assets, in which the core network assets are marked by the red grid. There are different network assets in cyberspace. These network assets can be divided into core assets and noncore assets under the artificial definition. We need to identify the core assets among them.

Specifically, we set up such a scene in this article: IP-based government agency websites and important facilities are regarded as IP-based core network assets, and then it is necessary to find an effective detection method to find relevant core network assets in the cyberspace. Based on this concept, we collect some corresponding core asset and noncore asset data in the Internet. Based on this dataset, we propose a detection technology based on neural network pretraining to detect IP-based core network assets.

3.2. Network Asset Data Features. In this paper, we use IP-based related filing information features for data analysis. Table 1 shows the relevant IP-based network asset feature information, including data features such as IP terminal allocation methods, access gateways, and operators. We believe that these features can effectively characterize whether the network asset behind this IP are core network asset, and the similarity in features also represents the similarity of the asset behind it. For instance, if a network asset and a core network asset have the same first-level filing unit, it means that it is probably also a core network asset.

3.3. Vectorization of Features. When IP features are used in neural network detection, they need to be vectorized first to convert raw features into digital features, and map the feature attributes of different discrete spaces into a unified continuous numerical space.

3.3.1. Combination of Feature Values. In this paper, we combine the features with attributes by similarity to reduce the sparseness of data features. For instance, the access method (access_method) may have multiple values. We take it as a set. If two IPs are not empty in the feature set of this feature, they will be set to the same feature value. Others such as the e-mail service port (e-mail_port), the open port (open_port) all adopt this method. This feature processing method can effectively reduce the sparsity of the feature space, so that the IP feature attributes of the same core network assets have a higher degree of similarity, and the detection difficulty is reduced.

3.3.2. Vectorization of Text Features. Many features in feature composition of the IP-based network asset are textual information, such as the end use unit (end_use_unit) and the first-level filing unit (first_level_filing_unit), etc. In the actual network environment, many text feature information is not completely consistent, but they have the same meaning. For instance, in the feature of the country where the IP is located (country_ip), “China” and “the People’s Republic of China” express the same meaning. Therefore, in this paper, we adopt the method of text escaping and similarity calculation to deal with this type of feature, and replace the text with the same meaning with one feature value. After the feature values are merged, each text feature is converted into digital information through category coding.

3.3.3. Normalization of Digital Features. In the feature composition of the IP-based network asset, after text features are converted to digital features, the initial vectorized features are formed by combining with the digital features (is_authority_server, is_recursive, etc.). Different digital features have different scopes and scales, and exist in different feature spaces. If the neural network is used directly to detect these features, it will fail due to the high range of feature values. Therefore, we need to normalize each feature. MinMax [24] can reduce the data to [0,1] while maintaining the original distribution relationship between features. In this paper, we use MinMax for normalization for each feature, as shown below:

$$x = \frac{x - \min x_i}{\max x_i - \min x_i}$$

(1)

3.4. Missing Features. In the actual network, the IP feature information is not complete. The same is true for various attributes in the IP attribute dataset collected in this paper, and there is a phenomenon of missing features. We believe that when characterizing whether the network asset behind an IP is the core network asset, not all features have the same weight, but different features have different characterization strengths. Therefore, we divide the relevant features in Table 1 into noncore features and core features based on
Figure 1: The conceptual diagram of the distribution of IP-based core network assets and IP-based noncore network assets.

Table 1: Feature composition of the IP-based network asset.

| Index | Attribute               | Interpretation                                                                 |
|-------|-------------------------|-------------------------------------------------------------------------------|
| 0     | end_distribution_method | The end distribution method (reserved, self-use, distribution and use, self-contained) |
| 1     | end_use_unit            | The end use unit of the IP                                                    |
| 2     | end_access_unit         | The end access unit of the IP                                                 |
| 3     | first_level_filing_unit | The first-level filing unit of the IP                                         |
| 4     | usage_method            | The usage method of the IP (dynamic, dynamic fixed network, dynamic mobile network, static, unused, unknown) |
| 5     | ag_of_ip                | The access gateway of the IP                                                  |
| 6     | prov_ag                 | The province where the access gateway is located                             |
| 7     | city_ag                 | The city where the access gateway is located                                  |
| 8     | county_ag               | The county where the access gateway is located                                |
| 9     | detail_addr_ag          | The detailed address of the access gateway                                    |
| 10    | country_ip              | The country where the IP is located                                           |
| 11    | prov_ip                 | The province where the IP is located                                          |
| 12    | city_ip                 | The city where the IP is located                                              |
| 13    | county_ip               | The county where the IP is located                                            |
| 14    | detail_addr_ip          | The detailed address of the IP                                                |
| 15    | as_num                  | The AS number of the IP                                                       |
| 16    | Operators               | The operators of the IP                                                       |
| 17    | is_routable             | Whether the IP is routable                                                    |
| 18    | prefix_bgp              | The BGP [22] prefix of the IP                                                 |
| 19    | prefix_bgp_as           | The BGP prefix of the AS number                                               |
| 20    | Longitude               | The longitude of the IP                                                       |
| 21    | Latitude                | The latitude of the IP                                                        |
| 22    | addr_type               | The address type of the IP (network address, end user address, service user address) |
| 23    | access_method           | The access method of the IP (dedicated line access, broadband access, mobile network access) |
| 24    | net_loc                 | The network location of the IP (core network, backbone network, metropolitan area network, access network, customer premises network) |
| 25    | cloud_serv              | The cloud service used by the IP                                              |
| 26    | Website                 | The website of the IP                                                         |
| 27    | is_authority_server     | Is it an authoritative server                                                  |
| 28    | is_recursive            | Is it a recursive server                                                      |
| 29    | is_recursive_req        | Is it a recursive request server                                               |
| 30    | is_idc                  | Is it Internet data center                                                    |
experience and experimental analysis, as shown in Table 2. Noncore features can assist the detector to determine whether it is an IP-based core network asset, but it does not play a decisive role. Core features have greater weight and have greater instructions for IP asset identification. In the collection of IP-based network asset data in this paper, we replace with 0 when noncore features are missing, and discard relevant samples when core features are missing.

### 4. Methodology

#### 4.1. Problem definition

In this paper, we identify IP-based core network assets. In order to generalize the detection problem, so as to cover similar detection scenes, we formalized the corresponding detection process as follows:

There is network asset feature \( X = R^d \), \( d \) means the feature space, \( Y = \{ \text{core network asset, non – network asset} \} \) means the asset category, i.e., the label space. There is a network asset dataset \( D = \{ (x_i, y_i) | i = 1, 2, \ldots, N; x_i \in X; y_i \in Y \} \). Our task is to find a detection function \( f \) that maps \( X \) from the feature space to the label space \( Y \). For a network asset sample \( x, f(x) \in Y \) is given as the label of the sample as to whether it is a core network asset.

#### 4.2. MAE-CAD

**4.2.1. Overview**

In this paper, we propose a detection method based on pretraining of multiple autoencoders, MAE-CAD, for the discovery of IP-based core network assets. First, build multiple unbalanced AutoEncoders (AEs) for unsupervised pretraining, mainly based on the idea of layer-by-layer pretraining of neural networks based on Stacked Auto Encoder (SAE). The encoder output in the previous AE is the encoder input of the next AE. After pretraining, encoders in AEs and a softmax classification layer are combined as the classifier of MAE-CAD, and the labeled dataset is used for the fine-tuning of the classifier parameters. As shown in Figure 2, we show the pretraining and fine-tuning phase of MAE-CAD; the details are described later.

**4.2.2. MAE**

The multiple autoencoder (MAE) in MAE-CAD is improved based on the traditional AutoEncoder (AE) and Stacked AutoEncoder (SAE). Traditional AE is a neural network which is an unsupervised learning algorithm which uses back propagation to generate output value which is almost close to the input value, and the infrastructure is shown in Figure 3. Both encoder and decoder form a neural network. The encoder learns the function \( h \) and performs feature extraction and feature dimensionality reduction on the input data \( x \) to obtain the intermediate feature \( z \). The decoder learns the function \( h^{-1} \) to reconstruct the intermediate feature \( z \) and restore it to \( x' \) with the same dimension as \( x. x' \) must be as same as \( x \) as possible. Sparse autoencoder is a single-layer AE, and both the encoder and the decoder consist of only one layer of neurons [25]. Traditional stacked autoencoder (SAE) is a neural network consisting of several layers of sparse autoencoders where output of each hidden layer is connected to the input of the successive hidden layer. The learned data from the previous layer is used as an input for the next layer and this continues until the training is completed [12]. A SAE selects one layer of layer-by-layer pretraining at a time, gradually extracts features, and then performs tasks such as classification.

In this paper, we use multiple AEs to form a pretrained neural network, MAE. Compared with single AE, MAE uses a stack of multiple AEs, and each AE is unbalanced. The encoder inside has more neurons and network layers than the decoder. Compared with SAE, MAE can be considered as a complex SAE, and MAE uses a similar method to SAE for pretraining. MAE is composed of unbalanced AEs instead of sparse autoencoders, i.e., the encoder in each AE is multi-layer and the decoder is single-layer, which can make the encoder in each AE have stronger feature extraction capabilities. Traditional SAE only pretrains a single layer of neural network layer-by-layer. Here, we explore the use of multiple layers for pretraining, so that the model can fit more complex functions, to extract more complete feature information from the input data.

**4.2.3. Pretraining**

As shown in the pretraining phase (a) in Figure 2, MAE takes the preprocessed IP-based network asset data features as input in the first AE \( 1 \). After AE \( 1 \) pretraining converges; its encoder output is used as the input of the next AE \( 2 \). By analogy, MAE gradually learns the characteristics of the input data, and performs feature abstraction and dimensionality reduction. The ability of MAE lies in its ability to gradually learn multiple expressions of the original data. Each AE is based on the intermediate features of the previous AE, which can extract more abstract and complex features. In addition, in this paper, AE uses the Mean Square Error (MSE) [10] loss function to calculate the
difference, and performs gradient backhaul to continuously update the parameters of the encoder and decoder in each AE. 

The $MSE$ is shown in Equation (2); the $x$ is the input of the encoder, the $x'_{ij}$ is the output of the decoder, and $d$ is the feature space dimension.

$$
MSE(x_i) = \frac{1}{d} \sum_{j=1}^{d} x_{ij}^2 - x'_{ij}^2; \quad i = 1, 2, \ldots, N. 
$$ (2)

We believe that the pretrained network fits the structure of the training data to a certain extent. This makes the initial value of the entire network in a suitable state, which is convenient for accelerating the iterative convergence in the fine-tuning phase and further improving performance.

4.2.4. Fine-Tuning. As shown in the fine-tuning stage (b) in Figure 2, after training the AEs structure in the MAE above, we discard the decoding process, connect the encoders of each AE, and then connect a softmax classification layer to form a classifier. At this time, the parameters of the pretrained classifier have been adjusted to a state suitable for processing data, and it has certain dimensionality reduction and feature extraction capabilities. Then, we use the labeled data as the input of the classifier, perform supervised learning, and fine-tune the model parameters, which can further improve the detection ability of the classifier in a specific direction. Since it is a 2-classification, we use the Binary Cross Entropy (BCE) [26] loss to calculate the

| Core feature composition of the IP-based network asset. |
|----------------------------------|
| end_distribution_method, end_use_unit, end_access_unit, first_level_filing_unit, usage_method, ag_of_ip, as_num, is_routable, prefix_bgp, net_loc, is_authority_server, is_recursive, is_recursive_req, opera_sys, open_port, e-mail_port, e-mail_cat, online_rate |

Figure 2: The overall detection process of MAE-CAD. (a) the pretraining phase. (b) the fine-tuning phase.

Figure 3: The infrastructure of traditional AE.
gradient and fine-tune the classifier parameters, as shown in Equation (3).

\[
BCE(x_i) = y_i \ln(y_i) + (1 - y_i) \ln(1 - y_i); \quad \hat{y}_i = \text{Classifier}(x_i); \quad i = 1, 2, \ldots, N.
\]  

4.3. Algorithm. MAE-CAD consists of the phases of building MAE, unsupervised pretraining, and supervised fine-tuning with IP-based network asset data. In this paper, combined with the data preprocessing method, we show the overall training and detection process of MAE-CAD in the form of an algorithm, as shown in Algorithm 1.

The input is the labeled IP-based network asset dataset \( D = \{(x_i, y_i) | i = 1, 2, \ldots, N; x_i \in X; y_i \in Y\} \), the unlabeled auxiliary IP-based network asset dataset \( D' = \{x_i | i = 1, 2, \ldots, N'; x_i \in X\} \), and the IP-based network asset dataset to be detected \( S' = \{x_i | i = 1, 2, \ldots, N''; x_i \in X\} \), and various hyperparameter sets \( P \) (number of neurons, number of trainings, etc.) required by MAE-CAD. The output is the labeled dataset \( S = \{(x_i, y_i) | i = 1, 2, \ldots, N''; x_i \in X; y_i \in Y\} \) that gives results for each sample in \( S' \).

The algorithm is divided into three stages. In the data preprocessing phase, the data are processed into a format suitable for neural network according to the method in Section 3. In the model construction phase, a neural network MSE composed of multiple AEs is established. In the pretraining phase, unsupervised pretraining is performed on the MAE and the model parameters are adjusted. In the fine-tuning phase, encoders in the MAE are extracted, and a softmax layer is added to form the classifier, and supervised data are used for fine-tuning. In the detection phase, the network asset data to be detected is judged and the corresponding result is given.

5. Experiment and Evaluation

5.1. Dataset. Data are the most important resource for practical research. Limited by permissions and privacy and security protection, there is no relevant dataset on IP-based network assets in the current network environment. Therefore, in order to be more complete and more authentic, we have invested a lot of work in collecting and labeling IP-based network asset data. In this paper, we cooperate with relevant security units to collect and clean up IP-based network asset datasets from the Internet. And, with government agency websites and important facilities as core network assets, data feature analyses are carried out. Due to security and privacy protection, these data have been desensitized to ensure that relevant information will not cause privacy leakage to others while supporting research.

Specifically, the data set consists of IP-based core network asset information, IP-based noncore network asset information data, and a large amount of unlabeled information, as shown in Table 3. Among them, there are 98,901 samples of core asset data. There are 1 million samples of noncore asset data, and the remaining unlabeled data reaches 10 million. The data labeling method is that the relevant service provider performs labeling through internal whitelisting and partial manual verification. Based on this approach, we annotate part of the core network asset data and noncore network asset data, hoping to conduct effective semi-supervised learning based on a small number of labeled data and a large unlabeled data.

5.2. Evaluation Metrics and Environmental Configuration.

In this section, we evaluate MAE-CAD. First, introduce the configuration of related experiments. Then compare MAE-CAD with other methods, and finally, make a self-comparison.

5.2.1. Evaluation Metrics. In this paper, we use IP-based core network asset data as positive samples, and IP-based noncore network assets as negative samples. \( TP \) represents the number of positive samples detected as positive samples, and \( FP \) represents the number of negative samples detected as positive samples. \( TN \) represents the number of negative samples detected as negative samples, and \( FN \) represents the number of positive samples detected as negative samples. Based on these data, we use the accuracy \( Acc \), the precision \( P \), and the recall \( R \) as the basic detection metrics, as shown in Equation 4. In addition, \( F1 \) is also used to verify the comprehensive performance of the model in detecting network assets (see Equation 5). Because the same experiment will be repeated \( k \) times and will produce multiple indicators of the same type, here we calculate the macro average to get the result.

\[
\text{Acc} = \frac{1}{k} \sum_{j=1}^{k} \frac{TP_j + TN_j}{TP_j + TN_j + FP_j + FN_j},
\]

\[
P = \frac{1}{k} \sum_{j=1}^{k} \frac{TP_j}{TP_j + FP_j},
\]

\[
R = \frac{1}{k} \sum_{j=1}^{k} \frac{TP_j}{TP_j + FN_j},
\]

\[
F1 = \frac{1}{k} \sum_{j=1}^{k} \frac{2 \times P_j \times R_j}{P_j + R_j}.
\]

5.2.2. Environmental Configuration. The system environment is Ubuntu16.04 LTS. The hardware facilities are 64-core CPU and 64G memory. Pytorch is used to implement the MAE-CAD.
Based on experience and previous experimental results, we set some hyperparameters for the model and related experiments. For the model structure, MAE-CAD uses the Dense layer as the basic network layer to construct a classifier structure of 40-1024-512-256-128-32-2. Among them, 40-1024-512-256 belongs to the encoder of AE 1 in MAE (the corresponding decoder structure is 256-40), and 128-32 belongs to the encoder of the AE 2 in MAE (the corresponding decoder structure is 32-256), 32-2 belongs to the softmax classification layer. The activation functions of the pretraining phase and the fine-tuning phase are both LeakyReLU (negative_slope = 0.2) [27], and the optimization functions are both Adam [28]. The learning rate of Adam in the pretraining phase is 1e-3, and in the fine-tuning phase is 0.5 × 1e-3. The dropout (drop_rate = 0.3) [29] is also used in the fine-tuning phase. For the basic hyperparameters of the experiment, the number of repetitions of the same experiment k = 5. For the data configuration, by default, the positive/negative ratio of data samples in the training set and test set are roughly 1:1 after deduplication. Both the training set and the test set are randomly selected from the raw dataset, and there is no intersection between the two to ensure the effectiveness of the detection.

5.3. Comparison with Other Methods. As there is no relevant research on IP-based core network asset discovery before, we chose classic machine learning methods (Bayes [30], Logistic Regression (LR) [31], Decision Tree (DT) [32], Support Vector Machine (SVM) [33]) for comparison, and these methods are widely used in network asset discovery. For instance, Mavrakis et al. [7] verify that the use of machine learning in the passive discovery of ICS network assets can produce promising performance improvements.

The experimental results are shown in Table 4. Compared with the traditional methods, the MAE-CAD method has better performance, and is optimized in 3 of the 4 metrics. Although SVM has the highest recall R of 98.06%, its accuracy Acc is only 84.21%, which means that it has identified many IP-based noncore network assets as core network assets, increasing the false negatives. In contrast, MAE-CAD has a better trade-off between false positives and false negatives, and can get 94.55% in Acc and 95.54% in R. Therefore, MAE-CAD has the best overall performance.

5.4. Self-Comparison of MAE-CAD

5.4.1. Efficiency of Pretraining. In this paper, we verify the performance improvement brought by pretraining in MAE-CAD. We set the neural network layer structure of this MAE-CAD classifier as 40-1024-512-256-128-32-2, and the specific MAE structure is set as shown in Table 5. We perform pretraining based on MAEs with different numbers of AEs to explore its impact on the final detection.

Figure 4 shows the convergence results of the accuracy in the training set during the fine-tuning phase under the different conditions, and MAE-CAD can reach convergence under these conditions. In general, the use of pretraining can effectively improve the detection performance of the model, and the best convergence result can be obtained when MAE-CAD has 2 AEs in MAE for pretraining.

Table 6 shows the various metrics of MAE-CAD with different AEs in the test set. Similarly, the pretrained MAE-
CAD with 2 AEs has the best comprehensive detection performance, and its Acc and F1 can reach 95.74% and 95.04%, respectively.

However, it should also be noted that there is a situation when the number of AEs in MAE increases during pre-training, the convergence effect of the combined classifier and the final detection result will decrease, even lower than MAE-CAD without pretraining. For instance, in the structure 40-1024-512-256-128-32-2, when the number of AEs in the MAE reaches 3 and 5 (this is equivalent to pretraining the MAE layer-by-layer), the convergence and detection effect of MAE-CAD is lower than no pretraining. This is because the ability of MAE in MAE-CAD to extract features of IP-based network asset data still has certain limitations. When the number of AEs is too large and MAE needs to stack pretrained deeper networks, the model is easy to overfit and the parameters are adjusted to a fixed range, which makes it difficult for the classifier to adapt to labeled data in the fine-tuning phase. However, we believe that adding pretraining and constructing MAE with a suitable structure can improve the classifier’s ability to discover IP-based core network assets.

### 5.4.2. Robustness of MAE-CAD

In the actual network environment, the core network assets in a region often account for a small proportion, and other noncore network assets account for a large proportion, and the data of the two are extremely unbalanced.

Therefore, whether an asset discovery method still has good detection performance in an unbalanced experimental environment is one of the key factors for its application in the actual network environment. In order to measure the detection performance of MAE-CAD in different

| Methods     | Acc (%) | P (%)   | R (%)   | F1 (%) |
|-------------|---------|---------|---------|--------|
| Bayes       | 61.81   | 53.48   | 81.24   | 64.50  |
| LR          | 92.51   | 87.32   | 96.46   | 91.66  |
| DT          | 92.30   | 86.25   | 97.52   | 91.54  |
| SVM         | 91.32   | 84.21   | 98.06   | 90.61  |
| MAE-CAD     | 95.74   | 94.55   | 95.54   | 95.04  |

| Pre-training method | No pre-training | Pre-training with 1 AE | Pre-training with 2 AEs | Pre-training with 3 AEs | Pre-training with 5 AEs |
|---------------------|-----------------|------------------------|-------------------------|-------------------------|-------------------------|
| Acc (%)             | 95.45           | 95.51                  | 95.74                   | 94.31                   | 94.26                   |
| P (%)               | 93.56           | 93.31                  | 94.55                   | 90.70                   | 91.09                   |
| R (%)               | 95.95           | 96.39                  | 95.54                   | 96.58                   | 95.94                   |
| F1 (%)              | 94.74           | 94.83                  | 95.04                   | 93.55                   | 93.45                   |
environments, here we design training scenes of core-network-assets: non-core-network-assets with different proportions. MAE-CAD is trained separately based on these training sets of different proportions, and then tested on the same test set to test the robustness of the model.

The recall convergence process of MAE-CAD with different ratio of core-network-assets (positive samples): noncore-network-assets (negative samples) in the training set during the fine-tuning phase is shown in Figure 5. When the number of positive samples in the training set, that is, the number of the IP-based core network asset data, decreases to the model’s ability to detect positive samples continues to decline, and the degree of volatility continues to increase. When it drops to 1/200, the recall R can only converge to around 90%. This is because the positive samples in the training set continue to decrease, and the severe data imbalance causes the model to fit more features of the negative samples during the training process, so it is easier to judge an unknown sample as a negative sample (noncore network asset). This leads to a decline in recall rates.

However, MAE-CAD has powerful feature extraction capabilities. Even when the ratio of positive and negative samples in the training set, that is, the ratio of IP-based core network assets to nonco network assets reaches 1:200, MAE-CAD can still achieve convergence. The experimental detection results of MAE-CAD trained with the training set of different proportions are shown in Table 7. MAE-CAD has excellent performance under different conditions, and it can still get the excellent performance of $\text{Acc} = 92.91\%$, $F1 = 91.57\%$, $R = 90.20\%$ at the ratio of 1:200.

6. Discussion

For the selection of hyperparameters, due to too the many choices of hyperparameters, the search space is huge. In this paper, a series of hyperparameters are set based on experience and previous experimental basis. For hyperparameters with multiple optional values, we use grid search to determine the optimal feature combination.

For the problem of offline identification, there are obstacles to the collection of IP-based core network asset data. The distribution of these data in the cyberspace is very sparse and complex, and it is difficult to use automated technical means to collect and clean up diverse IP network asset information. Therefore, these data are currently cleaned and identified offline.

For the detection in the actual network environment, because many IP-based network assets in the actual network environment have the same features, MAE-CAD will detect many identical IP feature information. Under this premise, MAE-CAD can reach $\text{Acc} = 98.11\%$, $P = 96.92\%$, $R = 97.40\%$, $F1 = 97.16\%$.

For the division of IP-based core network assets, we regard IPs of government agency websites and important facilities as the core network assets. But the definition of core assets is flexible and has different concepts in different scenes. In addition, apart from the division of core assets and noncore assets, different IP-based network assets have multiple levels. In a more complex scene, network assets cannot simply be classified into two categories. Therefore, the research of network asset discovery based on multiple levels is also our next research work.

7. Conclusion

Focusing on the identification of IP-based assets in the Internet, we propose the concept of IP-based core network assets. We collect data about relevant network assets based on this concept, and establish feature engineering to perform data preprocessing. In addition, we propose an IP-based core network asset discovery technology based on pretrained multiple autoencoders, MAE-CAD. Compared with traditional machine learning methods, our method can obtain 95.74% in $\text{Acc}$ and 95.04% in $F1$. In the actual network environment, due to the duplication of asset characteristics, MAE-CAD can reach $\text{Acc} = 98.11\%$ and $F1 = 97.16\%$. More importantly, under the condition of unbalanced data (core network assets: noncore network assets=1:200 in the training set), 92.91% in $\text{Acc}$ and 91.57% in $F1$ can still be
obtained. This shows that MAE-CAD has excellent robustness.

We will further expand the dataset, establish a more diversified network asset information database, and develop a multi-level classification plan for network assets. Then, we will also study online detection solutions to achieve real-time monitoring.

Data Availability
The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Acknowledgments
This study was supported by the National Natural Science Foundation of China (Grant No. U1736218).

References
[1] The 48th Statistical Report on Internet Development in China, “China Internet Network Information Center,” http://www.cnnic.net.cn/hlwfyj/hlwzxbg/hlwtybg/202109/P020210915523670981527.
[2] D. Bhamare, M. Zolanvari, A. Erbad, R. Jain, K. Khan, and N. Meskin, "Cybersecurity for industrial control systems: a survey," Computers & Security, vol. 89, Article ID 101677, 2020.
[3] F. Nejatabkhal, Y. W. Li, H. Liang, and R. R. Arabhi, "Cybersecurity of smart microgrids: a survey," Energies, vol. 14, no. 1, p. 27, 2021.
[4] S. N. Matheu, J. L. Hernández-Ramos, A. F. Skarmeta, and G. Baldini, "A survey of cybersecurity certification for the Internet of Things," ACM Computing Surveys, vol. 53, no. 6, pp. 1–36, 2020.
[5] M. Vermeer, J. West, A. Cuevas, and S. Niu, "SoK: a framework for asset discovery: systematizing advances in network measurements for protecting organizations," in Proceedings of the 6th IEEE European Symposium on Security & Privacy (EuroS&P), Vienna, Austria, September 2021.
[6] E. G. Ganzy, “Scalable asset discovery,” Vulnerability Scanning, and Penetration Testing for Remote Sites and Wireless Spectrums Utilizing an Embedded Linux Plug-PwniPlug and the Raspberry Pi B+ as a Sample Pen Test, No. KSC-E-DAA-TN19263, https://ntrs.nasa.gov/citations/2015000388, 2014.
[7] C. Mavrakis, “Passive asset discovery and operating system fingerprinting in industrial control system networks,” 2015, https://pure.tue.nl/ws/files/46916656/840171-1.%20pdf.
[8] O. Arkin, “A remote active OS fingerprinting tool using ICMP,” login: The Magazine of USENIX and Sage, vol. 27, no. 2, pp. 14–19, 2002.
[9] R. Li, M. Shen, H. Yu, C. Li, P. Duan, and L. Zhu, "A survey on cyberspace search engines, ccs," in Proceedings of the China Cyber Security Annual Conference, pp. 206–214, Springer, Beijing, China, August 2020.
[10] H. Marmolin, “Subjective MSE measures,” IEEE transactions on systems, man, and cybernetics, vol. 16, no. 3, pp. 486–489, 1986.
[11] T. P. Oliveira, J. S. Barbar, and A. S. Soares, “Multilayer perceptron and stacked autoencoder for Internet traffic prediction, aise,” in Proceedings of the IFIP International conference on network and parallel computing, pp. 61–71, Springer, Berlin, Germany, September 2014.
[12] G. Liu, H. Bao, and B. Han, "A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis," Mathematical Problems in Engineering, vol. 2018, Article ID 5105709, 10 pages, 2018.
[13] V. Mohan, Y. R. J. Reddy, and K. Kalpana, “Active and passive network measurements: a survey,” International Journal of Computer Science and Information Technologies, vol. 2, no. 4, pp. 1372–1385, 2011.
[14] R. A. Forouzan, TCP/IP Protocol Suite, McGraw-Hill Higher Education, New York, NY, USA, 2002.
[15] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Can we classify an iot device using tcp port scan?" in Proceedings of the 2018 IEEE International Conference on Information and Automation for Sustainability (ICIA&S), pp. 1–4, IEEE, Colombo, Sri Lanka, December 2018.
[16] D. Montigny-Leboeuf and F. Massicotte, Passive Network Discovery for Real Time Situation Awareness, Communications Research Center Ottawa, Ontario, Canada, 2004.
[17] D. Montigny-Leboeuf, A Multi-Packet Signature Approach to Passive Operating System Detection, Defence Research and Development Canadaottawa, Ontario, Canada, 2005.
[18] F. Klang. Network Device Enumeration and Identification Using Passive Asset Detection, M.S. thesis, Linköping University, Linköping, Sweden, 2021.
[19] A. Wedgbury and K. Jones, "Automated asset discovery in industrial control systems-exploring the problem," in Proceedings of the 3rd International Symposium for ICS & SCADA Cyber Security Research 2015 (ICS-CSR 2015), vol. 3, pp. 73–83, Ingolstadt, Germany, September 2015.
[20] R. Zheng, H. Ma, Q. Wang, J. Fu, and Z. Jiang, "Assessing the security of campus networks: the case of seven universities," Sensors, vol. 21, no. 1, p. 306, 2021.
[21] M. Zhang, Y. Chen, H. Chen, and Y. Zhao, "Design and implementation of a high performance network scanning system for vwidgets hosts," in Proceedings of the 2016 International Conference on Communications, Information Management and Network Security, pp. 119–122, Atlantis Press, Shanghai, China, September 2016.
[22] J. Matherly, Complete Guide to Shodan, Shodan, LLC, Pflugerville, TX, USA, 2015.
[23] R. Bates and H. Istance, “Zooming interfaces! Enhancing the performance of eye controlled pointing devices,” in Proceedings of the fifth international ACM conference on Assistive technologies, pp. 119–126, New York, NY, USA, July 2002.
[24] G. Tzortzis and A. Likas, "The MinMax k-Means clustering algorithm," Pattern Recognition, vol. 47, no. 7, pp. 2505–2516, 2014.
[25] N. Zeng, H. Zhang, B. Song, W. Liu, Y. Li, and A. M. Dobaie, "Facial expression recognition via learning deep sparse autoencoders," Neurocomputing, vol. 273, pp. 643–649, 2018.
[26] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," Annals of Operations Research, vol. 134, no. 1, pp. 19–67, 2005.
[27] J. Xu, Z. Li, and B. Du, "Reluplex made more practical: leaky ReLU," in Proceedings of the 2020 IEEE Symposium on Computers and Communications (ISCC), pp. 1–7, IEEE, Rennes, France, June 2020.
[29] N. Srivastava, G. Hinton, and A. Krizhevsky, “Dropout: a simple way to prevent neural networks from overfitting,” *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[30] I. Rish, “An empirical study of the naive Bayes classifier,” in *Proceedings of the IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, pp. 41–46, Seattle, WA, USA, August 2001.

[31] R. E. Wright, “Logistic regression,” in *Reading and Understanding Multivariate Statistics*, L. G. Grimm and P. R. Yarnold, Eds., pp. 217–244, American Psychological Association, Washington, DC, USA, 1995.

[32] A. J. Myles, R. N. Feudale, Y. Liu, N. A. Woody, and S. D. Brown, “An introduction to decision tree modeling,” *Journal of Chemometrics*, vol. 18, no. 6, pp. 275–285, 2004.

[33] W. S. Noble, “What is a support vector machine?” *Nature Biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.