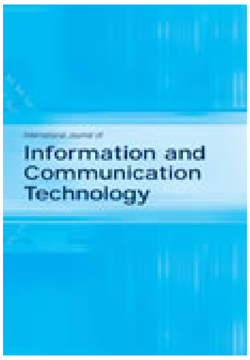


International Journal of Information and Communication Technology

ISSN online: 1741-8070 - ISSN print: 1466-6642
https://www.inderscience.com/ijict

Mobile cellular network security vulnerability detection using machine learning

Gongping Chen, Hong Wang, Chuanqi Zhang

DOI: 10.1504/IJICT.2023.10054797

Article History:

Received:	10 October 2022
Last revised:	21 December 2022
Accepted:	04 January 2023
Published online:	03 April 2023

Mobile cellular network security vulnerability detection using machine learning

Gongping Chen*, Hong Wang and Chuanqi Zhang

Lu'an Vocational Technical College,
No. 1, Zhengyang Road, Lu'an City,
Anhui Province, 237158, China
Email: chgp@lvtc.edu.cn
Email: wh@lvtc.edu.cn
Email: zhangcq@lvtc.edu.cn
*Corresponding author

Abstract: Due to the low monitoring accuracy and duration of the traditional cellular mobile network security infringement monitoring system, a computerised cellular mobile network intelligent blank monitoring system is proposed. It connects the blank detection module to the scanner according to the data attributes to scan the blanks in the mobile cellular network. During the tracking of cyberspace signals, the data space of the system session is controlled. Mobile cells of cellular networks introduce machine intelligence data processing learning algorithms hidden in the data. Experimental results show that ML-based cellular mobile network vulnerability detection (VD-MCN) can effectively improve system control accuracy and cellular network security space control efficiency. However, there are still some things that are ignored to improve the development efficiency of MCN, and developers often only care about themselves. Whether the corresponding functions can be realised in the process of code reuse, or there is lack of understanding, inspection and testing of the reuse code, the integration of these, can achieve our expected results.

Keywords: machine learning; ML; wireless communication; network security vulnerability; intelligent monitoring, mobile cellular network; MCN.

Reference to this paper should be made as follows: Chen, G., Wang, H. and Zhang, C. (2023) 'Mobile cellular network security vulnerability detection using machine learning', *Int. J. Information and Communication Technology*, Vol. 22, No. 3, pp.327–341.

Biographical notes: Gongping Chen received his MS degree from Hefei University of Technology in 2012. Currently, he is an Associate Professor at Lu'an Vocation Technology College. His research interests include personalised recommendation and network technology.

Hong Wang received her MS degree from Anhui University in 2011. Currently, she is an Associate Professor at Lu'an Vocation Technology College. Her research interests include personalised recommendation and database technology.

Chuanqi Zhang received his MS degree from Anhui University of Technology in 2017. Currently, he is a Lecturer at Lu'an Vocation Technology College. His research interests include network fundamentals and network security.

1 Introduction

In order to improve the development efficiency of MCN and reduce development costs, code reuse has been widely praised by developers, among which the prevalence of open source MCN is the most typical. The data shows that in 2018 the download of toolkits on major platforms increased to varying degrees. In 2018, npm alone had an incredible 317 billion downloads (Wu et al., 2022; Al Galil et al., 2021).

To put it simply, the full name of MCN is multi-channel network, that is, multi-channel network, a product form of multi-channel network, and a new economic operation mode of internet celebrities. This model unites professional content production (PGC) of different types and content, and with the strong support of capital, it guarantees the continuous output of content, so as to finally realise the stable realisation of business. The concept of MCN was first born abroad, and has gradually grown in China after localisation and improvement in recent years. MCN is equivalent to three bridges between content creators and platforms, between content creators and customers, and between customers and platforms. MCN provides traffic support for creators on the platform, helps creators and platforms solve commercial monetisation problems, and helps brand customers promote commercial transformation to achieve a triple win-win situation. The emergence of MCN has provided convenience for brand commercial promotion. More and more brands are more inclined to MCN one-stop marketing and promotion solutions. You have found the only organisation in China that can provide ‘Zhihu one-stop solution’. We are minimalist technology, ‘the only four licenses in China’ Zhihu MCN organisation in the consumer field, covering ‘beauty’, ‘digital’, ‘entertainment’, ‘finance’ and other categories, exclusive signing Zhihu head KOL With more than 6 million fans, it is the only organisation in China that has Zhihu official MCN certification, Zhihu content service provider, and Zhihu effect core agent. We are the domestic company with the ‘widest range of services’ in the Zhihu sector, and are considered by the industry as one of the institutions that ‘know how to play Zhihu best’. Our main business is [brand name operation], [social marketing case], [KOL brokerage], [information stream delivery], etc. to solve various demands of the brand in an all-round way.

However, while the widespread use of open source MCNs has brought great convenience to developers, it has also brought many problems:

1 MCN vulnerabilities are increasing day by day. In 2018, the number of npm exploits increased by 47%, with RHEL, Debian, Ubuntu increasing by more than 4x. The number of vulnerabilities in MCNs increased by 88% year-on-year in two years (Al-Azab et al., 2022).

2 MCN vulnerabilities are not paid attention to and repaired. The survey showed that more than 37% of open source developers did not implement any type of security testing during continuous integration (CI); more than 50% of developers lacked effective measures to test the security of Docker images during development. The median time from when a vulnerability was added to an open source MCN package to when it was fixed was more than 2 years.

3 The use of open source MCNs introduces more vulnerabilities. 78% of vulnerabilities stem from direct or indirect references to open source MCNs.

On the one hand, the level of developers is uneven, and the huge amount of code brings a severe test to developers' testing and maintenance. Most developers lack the complete ability to find and avoid security loopholes in the code; On the other hand, in order to improve the development efficiency of MCNs, developers often only care about whether they can implement corresponding functions in the process of code reuse, but lack the understanding, inspection and testing of the reused code. As a result, the security vulnerabilities exposed in MCNs are increasing day by day.

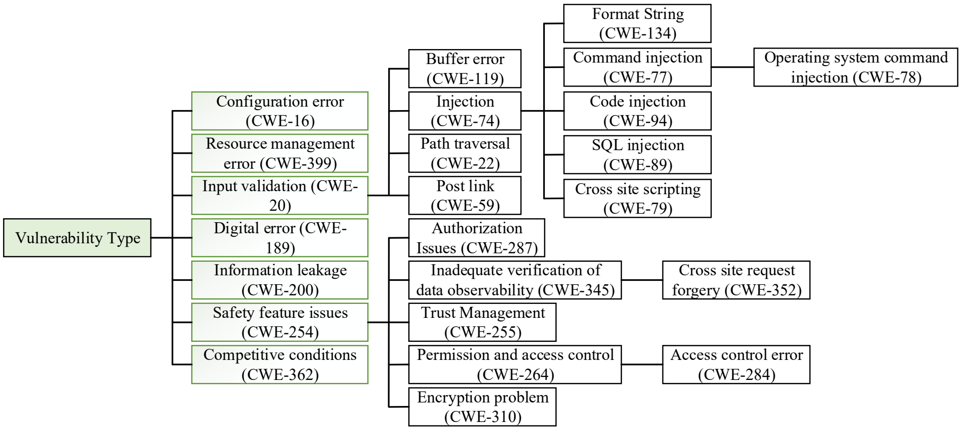
The MCN vulnerability is a security hole implanted during the development of the MCN due to the negligence of developers or the limitations of the programming language itself. This flaw allows attackers to perform malicious actions, including exposing sensitive information, controlling program operations, and compromising computer systems (Al-Mekhlafi et al., 2020). MCN vulnerabilities generally do not affect the operation of programs under normal logic, so it is often difficult to find them. However, once used by intentional people, they can weaken the security of the MCN system and can cause significant damage to businesses and individuals. At the same time, vulnerabilities in some basic and popular open source MCNs (Heartbleed, ShellShock, etc.) also threaten the security of thousands of companies and their customers around the world, causing immeasurable economic losses directly or indirectly. The ransomware attack 'WannaCry' that broke out in May 2017 attacked more than 200,000 computer devices in more than 150 countries by exploiting several vulnerabilities in Microsoft systems, and eventually caused economic losses of nearly 4 billion US dollars. Therefore, how to quickly and accurately detect the vulnerabilities in the code has become a crucial topic in the MCN industry and computer security field.

2 Related work

2.1 Vulnerability classification

First, using the development concepts view of the CWE list, this paper classifies common vulnerabilities as shown in Figure 1 from a broader perspective.

Figure 1 Vulnerability classification (see online version for colours)


Vulnerability Type
Configuration error (CWE-16)
Resource management error (CWE-399)
Input validation (CWE-20)
Digital error (CWE-189)
Information leakage (CWE-200)
Safety feature issues (CWE-254)
Competitive conditions (CWE-362)
Buffer error (CWE-119)
Injection (CWE-74)
Path traversal (CWE-22)
Post link (CWE-59)
Format String (CWE-134)
Command injection (CWE-77)
Operating system command injection (CWE-78)
Code injection (CWE-94)
SQL injection (CWE-89)
Cross site scripting (CWE-79)
Authorization Issues (CWE-287)
Inadequate verification of data observability (CWE-345)
Cross site request forgery (CWE-352)
Trust Management (CWE-255)
Permission and access control (CWE-264)
Access control error (CWE-284)
Encryption problem (CWE-310)

2.2 Vulnerability detection using ML

At present, the vulnerability detection methods using ML mainly involve the first two, and according to whether they need manual help to define their features, this paper divides them into two categories (Liu et al., 2019; Alsubari et al., 2022).

2.2.1 Vulnerability detection using traditional ML

Traditional ML methods still rely on domain experts to manually define some attributes and features, and then classify these features through ML algorithms such as SVM, RF, and DT. Vulnerability detection methods using traditional ML for specific vulnerability types and methods independent of vulnerability types according to the scope of applicable vulnerabilities (Al-Garadi et al., 2020; Trinh et al., 2019).

The premise of methods targeting specific vulnerability types is to classify vulnerabilities into different types with the help of domain knowledge of domain experts (such as vulnerability principles), and then learn vulnerability patterns corresponding to different types of vulnerabilities through ML techniques (Hossain et al., 2019; Uprety and Rawat, 2020). In Park et al. (2020), it proposed a method called Chucky to help programmers speed up manual review of source code. The method firstly marks the source of pollution of the source code, and identifies the abnormal or missing condition related to the safety-critical object using the mark, so that the input validation vulnerability in the C program can be detected.

Besides, there are some detection methods for other vulnerability types, such as format string, information disclosure (Cvitić et al., 2021), etc. The types of vulnerabilities that can be detected by detection methods for specific types of vulnerabilities are very clear. There is a one-to-one correspondence between detection methods and detection types. The participation of domain experts is to help identify the characteristics of the corresponding types of vulnerabilities and improve the detection effect.

Vulnerability type-independent methods do not stick to some specific vulnerability types, but use ML technology to learn patterns and rules that can characterise vulnerabilities for various types of vulnerabilities (Liu et al., 2018). Vulnerability detection methods using traditional ML rely on the domain knowledge of domain experts, and need to manually define some attributes and features, and then use ML methods to classify using these features (Qin et al., 2020). However, the granularity of the code used for training is too coarse, so that although the ML model can more accurately classify the presence or absence of vulnerabilities, it cannot locate the specific location of the vulnerability.

2.2.2 Vulnerability detection using deep learning

Such methods no longer rely on domain experts to manually discover and define features, but use deep neural networks to autonomously learn and generate corresponding vulnerability patterns (Challita et al., 2019; DeAlmeida et al., 2021). Extract a higher level and generalisable function representation from the abstract syntax tree (AST), use this function representation as the training set to train a vulnerability classification model on the bidirectional LSTM neural network, and apply it to the vulnerability detection task in cross project scenarios (Islam et al., 2019; Zhang et al., 2018). A novel numerical vector is calculated using the control flow chart of each binary function, and then the similarity can be effectively detected by measuring the distance between the embedding of two functions (Adi et al., 2020; Afzal and Murugesan, 2022). Use the neural network to learn the patterns in the input file from the past fuzzy exploration, and perform fuzzy mutation according to the past mutation and the corresponding code coverage information to guide future fuzzy exploration (Ashraf et al., 2020; da Costa et al., 2019). It is the first work to use deep learning to detect vulnerabilities at the slicing level (more sophisticated than the function level), so that vulnerabilities can be located (Karimipour et al., 2019; Waheed et al., 2020). It is a framework using deep learning, which uses syntax, semantics and vector-based representations to detect various types of vulnerabilities at the slice level.

The vulnerability detection method using deep learning largely gets rid of dependence on manual work, and instead generates corresponding vulnerability patterns for various types of vulnerabilities through autonomous learning (Cui et al., 2018). Because the powerful learning ability of the machine can learn some hidden information that cannot be found by human, the detection effect of this kind of methods is generally improved compared with the original methods. However, as this method is still in its infancy, there are still many shortcomings: the granularity of detection mostly stays at the function level, which cannot achieve accurate vulnerability location. There is no large-scale annotation dataset corresponding to various vulnerability types, and the annotation results mostly rely on other static analysis tools. The applied depth learning model is limited, and the detection effect of other depth learning models cannot be verified.

To sum up, vulnerability detection methods using code similarity, rules, symbol execution and fuzzy testing are relatively mature in research, but due to the inherent limitations of the methods, they have their inevitable drawbacks. Although it has achieved no inferior performance to traditional methods relying on its powerful learning ability. As mentioned in the previous section, the security vulnerabilities exposed in MCNs are increasing day by day, which has brought severe challenges to both the MCN industry and the computer security field. Therefore, how to combine and apply the emerging ML technology to relieve the security pressure of MCN industry is a subject of great research and practical significance.

3 Methods

3.1 Hardware design

Unsecurity of MCN is one of the main reasons that communication network is attacked and invaded. It is very important to find the security vulnerabilities of MCN in time. The overall structure of the MCN is shown in Figure 2.

The MCN security vulnerability monitoring system mainly scans the MCN through ICMP echo and ICMP. It is the process of waiting for the host to respond after sending a request to the MCN and controlling it. The internal structure of the control system is complex, the implementation is delayed, and it is difficult to track the gap of MCN in time. The system has been improved as shown in Figure 3.

Figure 2 Overall architecture of MCN (see online version for colours)

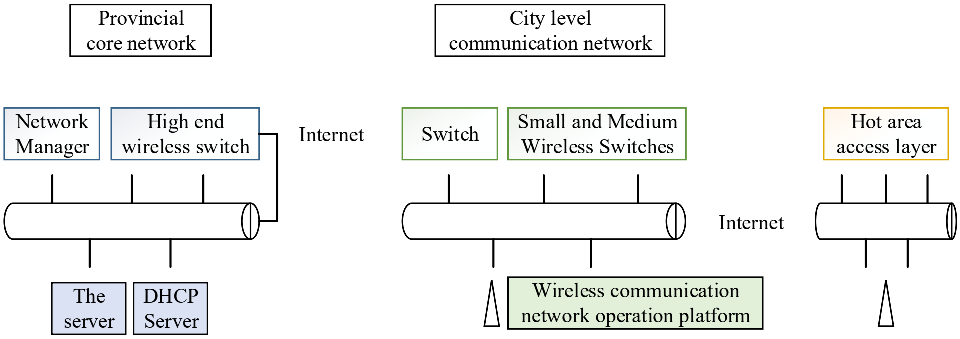


Figure 3 Overall framework of the monitoring system (see online version for colours)

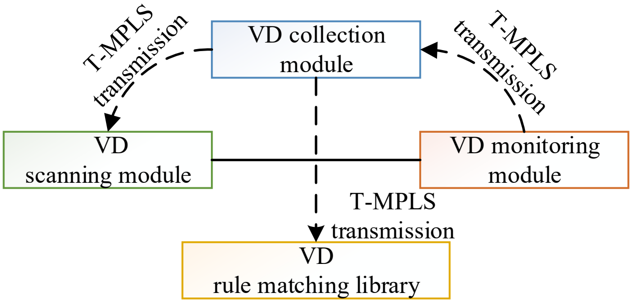
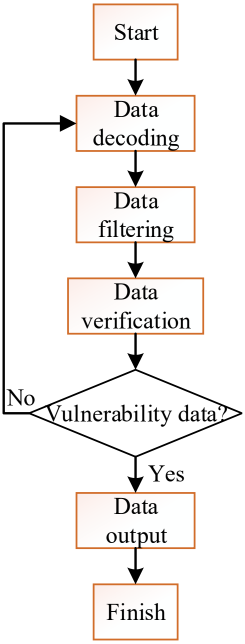


Figure 4 The intelligent monitoring process of system vulnerabilities (see online version for colours)



In Figure 3, the data acquired through acquisition, scanning and monitoring are transmitted through data transmission technology, which can directly extract blank spaces. The improved control system is easier to find loopholes and reduce system development costs. The MCN security vulnerability data monitoring process is shown in Figure 4.

3.2 Overview of monitoring model

PCA is a traditional and major dimensionality reduction method in the field of data analysis and dimensionality reduction today, that is, the orthogonal transformation is used to transform a linearly related variable into a linear but unrelated new variable or major component, so as to facilitate the development of new variables to represent the main characteristics of data analysis in smaller dimensions. PCA algorithm steps are as follows.

Assuming a high-dimensional sample set $D = \{x_1, x_2, \ldots, x_n\}$, the first step is to centre the samples in D:

$$x_i \leftarrow x_i - \frac{1}{m}\sum_{i=1}^{m} x_i \tag{1}$$

The second step is to count the covariance matrix XX^T of the samples after centring, and then decompose its eigenvalues.

The final conclusion is to obtain d low-dimensional orthonormal basis $v_1, v_2, \ldots, v_d$, and to minimise the squared sum of reconstruction deviations between the basis vector and the reconstructed high-dimensional data result X, the formula is as follows:

$$E = \sum_{i=1}^{n} \left\| x_i - \sum_{j=1}^{d} \left(x_i \cdot v_j \right) v_j \right\|^2 \tag{2}$$

PCA can reduce the sample dimensions of higher dimensional space through linear projection, and the calculation speed is fast, so it can solve the linear correlation well, but it cannot handle the samples of higher order correlation, and the data distribution must obey the Gaussian distribution, which has certain limitations.

Non-negative matrix factorisation (NMF) refers to the decomposition of a sample dataset into two matrices, which have low rank and non-negative characteristics. The formula is as follows:

$$V \approx W \times H \tag{3}$$

NMF can be converted into the following optimisation problem solving.

$$\min f_A(W, H) \tag{4}$$

where $f_A(W, H)$ represents the difference between A and W, H, f_A is the measurement standard, and GKLD and SED are two commonly used standards.

GKLD is defined as:

$$D\left((V \| WH) = \sum_{ij} V_{ij} \log \frac{V_{ij}}{(WH)_{ij}} ij - V_{ij} + (WH)_{ij} \right), s.t. W \geq 0, H \geq 0 \tag{5}$$

The iterative formula is as follows:

$$W_{ir} \leftarrow W_{ir} \frac{\sum_u H_{ru} V_{ru} / (WH)_{iu}}{\sum_v H_{rv}}; H_{ru} \leftarrow H_{ru} \frac{\sum_i W_{ir} V_{iu} / (WH)_{iu}}{\sum_k W_{kr}} \tag{6}$$

SED is defined as:

$$F = \|V - WH\|^2, s.t. W \geqslant 0, H \geqslant 0 \tag{7}$$

The iterative formula is as follows:

$$W_{ir} \leftarrow W_{ir} \frac{(VH^T)_{ir}}{(WHH^T)_{ir}}; H_{rj} \leftarrow H_{rj} \frac{(W^T V)_{rj}}{(W^T WH)_{rj}} \tag{8}$$

To make the application scope of NMF wider, the constraint conditions of non-negativity are relaxed, and a semi non-negative matrix factorisation (semi-NMF) is proposed. The approximate decomposition is as follows:

$$X^{\pm} \approx Z^{\pm} H^{+} \tag{9}$$

To solve the problem that the single-layer NMF lacks the ability to represent the model, a multi-layer non-negative matrix factorisation (MNMF) is proposed. In fact, this algorithm is a simple hierarchical multi sequence decomposition process. The formula for performing sequence decomposition process is as follows: First, use non-negative matrix decomposition (NMF). In the second step, a similar decomposition is obtained from the decomposition result of the first step; the third step is to repeat the decomposition process until the termination condition. Finally, a model with base matrix of $Z_1 Z_2 \ldots Z_L$ is established.

$$X \approx Z_1 H_1; H_1 \approx Z_2 H_2; \ldots; X \approx Z_1 Z_2 \ldots Z_L H_L \tag{10}$$

On the basis of semi-NMF, in order to solve this drawback of MNMF, a deep semi-NMF algorithm is proposed. Its main formula is as follows:

$$X^{\pm} \approx Z_1^{\pm} H_1; X^{\pm} \approx Z_1^{\pm} Z_2^{\pm} H_2^{+}; \ldots; X^{\pm} \approx Z_1^{\pm} Z_2^{\pm} \ldots Z_L^{\pm} H_L^{+} \tag{11}$$

Backward derivation yields $H_1^{+} \approx Z_1^{\pm} Z_2^{\pm} \ldots Z_{L-1}^{\pm} Z_L^{\pm} H_L^{+}$.

The steps of the training algorithm are roughly as follows: firstly initialise the data matrix $X \approx Z_1 H_1$ respectively, and then decompose the feature matrix $H_1 \approx Z_2 H_2$ until all layers are traversed. Then alternately optimise tow factors to reduce the reconstruction error, the objective function is as follows:

$$C = \frac{1}{2} \|X - Z_1 Z_2 \ldots Z_L H_L\|^2 \tag{12}$$

Compared with MNMF, the decomposition of each layer in deep semi-NMF is not independent, and the decomposition of each layer is realised alternately. Therefore the updated Z_L law is as follows:

$$Z_L = \left(\Psi^T \Psi\right)^{-1} \Psi^T X \tilde{H}_i^T \left(\tilde{H}_i \tilde{H}_i^T\right)^{-1} \tag{13}$$

Update H_i rules are as follows:

$$H_i = H_i \text{yuandian} \cdot \sqrt{\frac{[\Psi^T X]^+ + [\Psi^T \Psi]^- H_i}{[\Psi^T X]^- + [\Psi^T \Psi]^+ H_i}} \tag{14}$$

Deep semi-NMF can automatically learn latent hierarchical attributes and find the most suitable clustering data representations.

4 Experiments

4.1 Evaluation indicators

This paper is the same as the evaluation value indicators of other deep learning anomaly detection algorithms. Three evaluation indicators are mainly used to measure the performance of each anomaly detection algorithm, namely AUC, Precision and Recall. The statistical source is the classification confusion matrix of anomaly detection, as shown in Table 1.

Table 1 Anomaly detection confusion matrix

True category	*Predicted normal*	*Forecast anomaly*
Normal	TP	FN
Exception class	FP	TN

1 ROC-AUC

In the previous anomaly detection methods, ROC is a widely used evaluation index. ROC curve is a curve formed by comparing the true positive rate (abnormal points are identified as abnormal) and false positive rate (normal points are identified as abnormal) according to the correct label information and detection results of nodes. In this paper, the AUC is used to quantify the performance of different anomaly detection algorithms.

The number of sample pairs is $M \times N$, the calculation method of AUC is formula (15).

$$AUC = \frac{\sum I\left(P_{\text{Positive sample}}, P_{\text{Negative sample}}\right)}{M \times N} \tag{15}$$

2 Accuracy

The accuracy rate represents the proportion of the nodes that are actually abnormal among all the prediction results. The accuracy rate is calculated by formula (16).

$$precision = \frac{TN}{TN + FN} \tag{16}$$

3 Recall rate

The recall rate represents the proportion of the abnormal nodes successfully detected in all the actual abnormal nodes of the sample. The recall rate is calculated by formula (17).

$$recall = \frac{TN}{TN + FN} \tag{17}$$

4.2 Anomaly detection

This section compares the VD-MCN algorithm with the classic methods in the field of network anomaly detection. MCN is a multi-channel network service, the model originated from the mature net red economy operation abroad, the essence is a multi-channel network product form, the PGC (professional content production) content united, with the strong support of capital, to ensure the continuous output of content, so as to finally achieve stable commercial realisations. Through experiments on five experimental datasets, multiple evaluation indicators are used to analyse and compare the detection capabilities of each method. The AUC scores of each anomaly detection method in the Blog Catalog and Flickr datasets are shown in Table 2.

It can be seen from the comparison of AUC performance evaluation experiment results in Table 2 that the AUC score of VD-MCN is 48.95% higher than LOF, 70.83% higher than SCAN, and 31.62% higher than AMEN. These three methods are classical algorithms. Although they have good effects in detecting exceptions in some specific aspects of the network structure, they have great limitations on attribute network anomaly detection and are not suitable for complex situations where multiple exceptions exist simultaneously. LOF and SCAN can only consider network structure and node attribute exceptions respectively. If these two types of exceptions exist at the same time, the detection rate will be greatly reduced due to their own limitations. The attribute network has good expressiveness, and generally has both characteristics, so the two methods have poor anomaly detection capabilities. The main purpose of AMEN method is to detect the abnormal clustering of neighbour nodes.

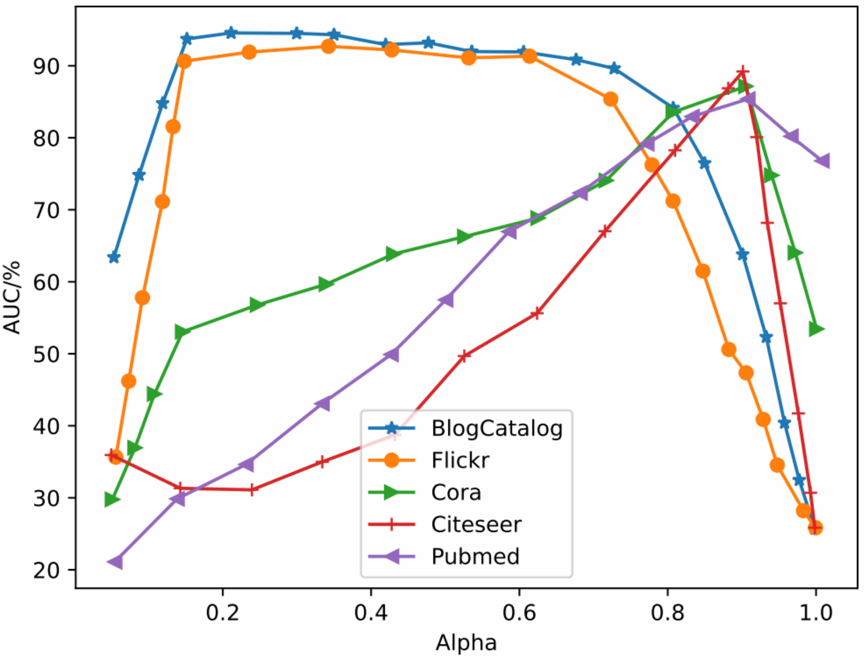
Table 2 AUC scores of anomaly detection methods in two datasets

Method	*Blog Catalog*	*Flickr*
LOF	49.17	48.83
SCAN	27.25	26.88
AMEN	66.46	60.45
Radar	71.02	72.88
Anomalous	72.83	71.57
Dominant	78.11	74.92
ADAN-CMA	84.73	83.86
ADAN-AMA	98.12	9.52

4.3 Parameter sensitivity

The experiment in this section analyses the influence of different parameter value selection on abnormal detection results in the process of feature extraction. Parameter α is used to measure the loss of structural anomaly and attribute anomaly, and parameter E is used to specify the dimension of feature extraction for each encoder. The AUC scores of each dataset under different parameters α are shown in Figure 5.

Figure 5 Alpha parameter sensitivity (see online version for colours)



Since the VD-MCN experiment has the best experimental effect, this section firstly conducts the anomaly detection experiment of parameter α on the five datasets respectively, and the value of parameter α is 0.0~0.1. It can be seen from the broken line in Figure 5 that parameter α has a significant impact on the detection effect. Among them, when the Blog Catalog dataset and Flickr dataset are in the interval [0.1, 0.6], the detection effect is better, and when the parameter α is greater than 0.7, the effect is significantly reduced. Combined with the datasets, it is found that the two datasets have a large number of nodes and dense connections between nodes, but the attribute dimension in each network is not high, so the abnormal attributes of attributes are more obvious, and more attention should be paid to extracting features.

In the three datasets of Cora, Citeseer, and Pubmed, the AUC keeps improving with the increase of parameter α, which shows that when minimising the reconstruction error, the reconstruction quality of the adjacency matrix should be improved. The analysis found that the connection relationship of nodes in these three datasets is simple, especially in the Citeseer dataset, the number of edges is less than the number of nodes, which means that structural anomalies in the network are easier to detect. In addition, the

AUC score drops significantly when parameter α is 0 or 1, indicating that neither only considering structure nor only considering attributes can effectively detect anomalies.

4.4 Time consuming

To verify the effectiveness of this method, the experiment compares the time of monitoring MCN security vulnerabilities according to the method, Al-Mekhlafi et al. (2020) and Liu et al. (2019), as shown in Figure 6.

Figure 6 Time-consuming comparison of vulnerability monitoring (see online version for colours)

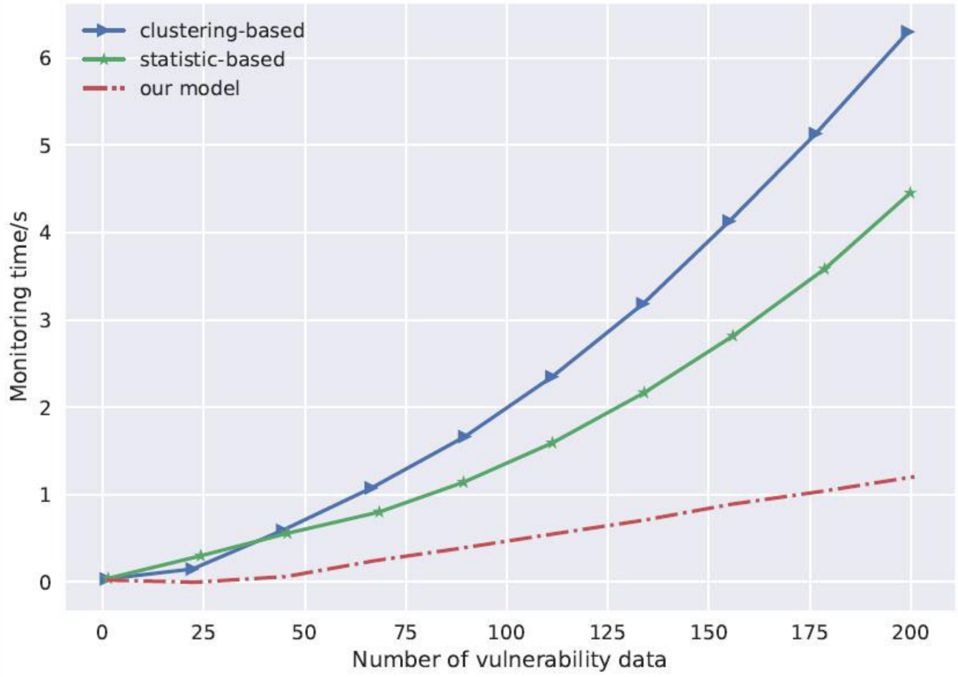


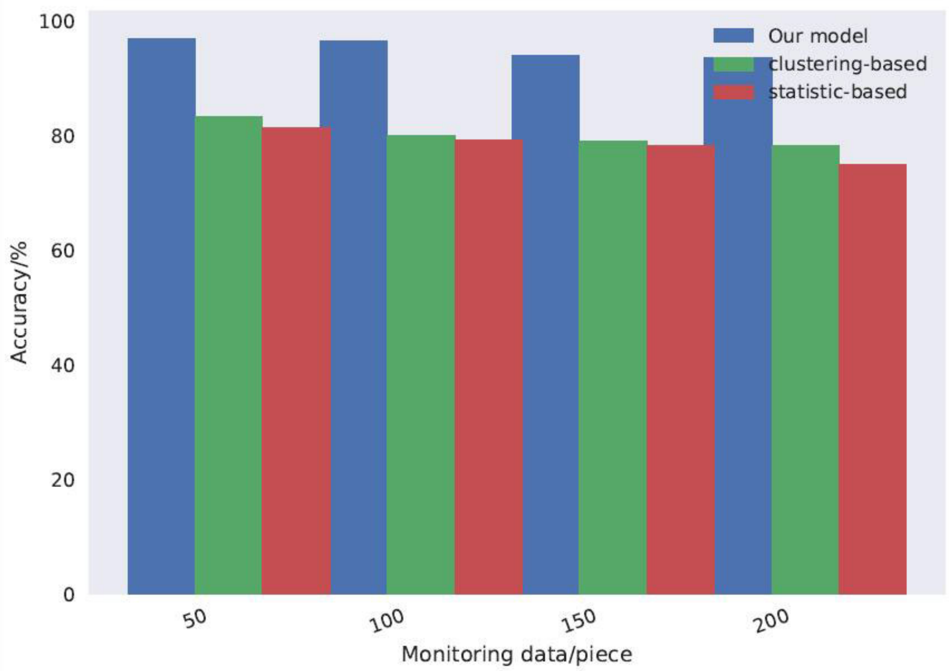
Figure 6 shows that under the same parameters, the space detection in this article takes relatively short time. When 200 data gaps are detected, the monitoring duration under this method is about 1.4 seconds, the monitoring duration under Al-Mekhlafi et al.'s (2020) method is about 6.3 seconds, and the monitoring duration under Liu et al.'s (2019) method is about 4.5 seconds. These data are collected again to track the difference in the MCN. In this paper, the MCN directional monitoring system can directly and effectively identify and capture the gaps in the MCN, which verifies the effectiveness and feasibility of this method.

4.5 Accuracy analysis

To further verify the effectiveness of the system, the experiment compares the gap data found when monitoring 200 radio communication data. The experimental results are shown in Figure 7.

The Figure 7 shows that with the increase of monitoring data, the monitoring accuracy of the three methods has a downward trend. It can be seen that when the monitoring data reaches 100, the accuracy of the gap data tracked by this method is 95%, the accuracy of the vulnerability data tracked by files is 85%, and the accuracy of the vulnerability data tracked by files is 82%. Although the monitoring accuracy has a downward trend, this method achieves more than 90% accuracy in monitoring the vulnerability data, which exceeds the monitoring accuracy of the other two methods, and verifies the reliability of this method.

Figure 7 Comparison of the accuracy of monitoring vulnerability data by different methods (see online version for colours)



5 Conclusions

Due to the deepening application of computer technology, the demand and scale of MCN is increasing, and the quantity and complexity of code is growing exponentially, which undoubtedly poses a great challenge to developers. On the one hand, the level of developers varies, and the huge amount of code poses a severe test for developers' testing and maintenance. Most developers lack the complete ability to find and avoid security vulnerabilities in the code; on the other hand, in order to improve the development efficiency of MCN, developers often only care about whether they can achieve the corresponding functions in the code reuse process, and lack the understanding, inspection and testing of the reused code. Therefore, how to quickly and accurately detect vulnerabilities in the code has become a key topic in the MCN industry and computer security field.

Acknowledgements

1 The Natural Science Research Project of Higher School of Anhui Province in 2021 under Grant No. KJ2021A1361.

2 The Second Batch of China University's Production, Research and Innovation Fund of 2020, 'New Generation of Information Technology Innovation Project', No. 2020ITA02018.

3 Provincial Quality Engineering Project of colleges and universities of Anhui Provincial Department of education in 2019 under Grant No. 2019xfzx04.

4 Provincial Quality Engineering Project of colleges and universities of Anhui Provincial Department of education in 2018 under Grant No. 2018yljc188.

5 Academic funding project for top talents in disciplines (majors) of Anhui Provincial Department of education in 2019 under Grant No. gxbjZD74.

References

Adi, E., Anwar, A., Baig, Z. and Zeadally, S. (2020) 'ML and data analytics for the IoT', *Neural Computing and Applications*, Vol. 32, No. 20, pp.16205–16233.

Afzal, R. and Murugesan, R.K. (2022) 'Rule-based anomaly detection model with stateful correlation enhancing mobile network security', *Intell. Autom. Soft Comput*, Vol. 31, No. 3, pp.1825–1841.

Al Galil, A., Mohammed, F., Zambare, S.P., Al-Mekhlafi, F.A. and Al-Keridis, L.A. (2021) 'Effect of dimethoate on the developmental rate of forensic importance Calliphoridae flies', *Saudi Journal of Biological Sciences*, Vol. 28, No. 2, pp.1267–1271.

Al-Azab, A.M., Zaituon, A.A., Al-Ghamdi, K.M. and Al-Galil, F.M.A. (2022) 'Surveillance of dengue fever vector Aedes aegypti in different areas in Jeddah city Saudi Arabia', *Adv. Anim. Vet. Sci*, Vol. 10, No. 2, pp.348–353.

Al-Garadi, M.A., Mohamed, A., Al-Ali, A.K., Du, X., Ali, I. and Guizani, M. (2020) 'A survey of machine and deep learning methods for internet of things (IoT) security', *IEEE Communications Surveys & Tutorials*, Vol. 22, No. 3, pp.1646–1685.

Al-Mekhlafi, F.A., Alajmi, R.A., Almusawi, Z., Al GAlil, F.M.A., Kaur, P., Al-Wadaan, M. and Al-Khalifa, M.S. (2020) 'A study of insect succession of forensic importance: Dipteran flies (diptera) in two different habitats of small rodents in Riyadh City, Saudi Arabia', *Journal of King Saud University-Science*, Vol. 32, No. 7, pp.3111–3118.

Alsubari, S.N., Deshmukh, S.N., Alqarni, A.A. and Alsharif, N.H.T. (2022) 'Data analytics for the identification of fake reviews using supervised learning', *CMC-Computers, Materials & Continua*, Vol. 70, No. 2, pp.3189–3204.

Ashraf, J., Bakhshi, A.D., Moustafa, N., Khurshid, H., Javed, A. and Beheshti, A. (2020) 'Novel deep learning-enabled LSTM autoencoder architecture for discovering anomalous events from intelligent transportation systems', *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 7, pp.4507–4518.

Challita, U., Ferdowsi, A., Chen, M. and Saad, W. (2019). ML for wireless connectivity and security of cellular-connected UAVs', *IEEE Wireless Communications*, Vol. 26, No. 1, pp.28–35.

Cui, L., Yang, S., Chen, F., Ming, Z., Lu, N. and Qin, J. (2018) 'A survey on application of ML for internet of things', *International Journal of ML and Cybernetics*, Vol. 9, No. 8, pp.1399–1417.

Cvitić, I., Peraković, D., Periša, M. and Gupta, B. (2021) 'Ensemble ML approach for classification of IoT devices in smart home', *International Journal of ML and Cybernetics*, Vol. 12, No. 11, pp.3179–3202.

da Costa, K.A., Papa, J.P., Lisboa, C.O., Munoz, R. and de Albuquerque, V.H.C. (2019) 'Internet of things: a survey on ML-based intrusion detection approaches', *Computer Networks*, Vol. 151, pp.147–157, DOI: 10.1016/j.comnet.2019.01.023.

DeAlmeida, J.M., Pontes, C.F., DaSilva, L.A., Both, C.B., Gondim, J.J., Ralha, C.G. and Marotta, M.A. (2021) 'Abnormal behavior detection based on traffic pattern categorization in mobile networks', *IEEE Transactions on Network and Service Management*, Vol. 18, No. 4, pp.4213–4224.

Hossain, E., Khan, I., Un-Noor, F., Sikander, S.S. and Sunny, M.S.H. (2019) 'Application of big data and ML in smart grid, and associated security concerns: a review', *IEEE Access*, Vol. 7, pp.13960–13988, DOI: 10.1109/ACCESS.2019.2894819.

Islam, S.N., Baig, Z. and Zeadally, S. (2019) 'Physical layer security for the smart grid: vulnerabilities, threats, and countermeasures', *IEEE Transactions on Industrial Informatics*, Vol. 15, No. 12, pp.6522–6530.

Karimipour, H., Dehghantanha, A., Parizi, R.M., Choo, K.K.R. and Leung, H. (2019) 'A deep and scalable unsupervised ML system for cyber-attack detection in large-scale smart grids', *IEEE Access*, Vol. 7, pp.80778–80788, DOI: 10.1109/ACCESS.2019.2920326.

Liu, C.H., Lin, Q. and Wen, S. (2018) 'Blockchain-enabled data collection and sharing for industrial IoT with deep reinforcement learning', *IEEE Transactions on Industrial Informatics*, Vol. 15, No. 6, pp.3516–3526.

Liu, Q., Liu, C., Wang, Y. et al. (2019) 'Integrating external dictionary knowledge in conference scenarios: the field of personalized machine translation method', *Journal of Chinese Informatics*, Vol. 33, No. 10, pp.31–37.

Park, S.T., Li, G. and Hong, J.C. (2020) 'A study on smart factory-based ambient intelligence context-aware intrusion detection system using ML', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 11, No. 4, pp.1405–1412.

Qin, Z., Cao, F., Yang, Y., Wang, S., Liu, Y., Tan, C. and Zhang, D. (2020) 'CellPred: a behavior-aware scheme for cellular data usage prediction', *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, Vol. 4, No. 1, pp.1–24.

Trinh, H.D., Zeydan, E., Giupponi, L. and Dini, P. (2019) 'Detecting mobile traffic anomalies through physical control channel fingerprinting: a deep semi-supervised approach', *IEEE Access*, Vol. 7, pp.152187–152201, DOI: 10.1109/ACCESS.2019.2947742.

Uprety, A. and Rawat, D.B. (2020) 'Reinforcement learning for IoT security: a comprehensive survey', *IEEE Internet of Things Journal*, Vol. 8, No. 11, pp.8693–8706.

Waheed, N., He, X., Ikram, M., Usman, M., Hashmi, S.S. and Usman, M. (2020) 'Security and privacy in IoT using ML and blockchain: threats and countermeasures', *ACM Computing Surveys (CSUR)*, Vol. 53, No. 6, pp.1–37.

Wu, D., Lei, Y., He, M., Zhang, C. and Ji, L. (2022) 'Deep reinforcement learning-based path control and optimization for unmanned ships', *Wireless Communications and Mobile Computing*, Vol. 2022, Article ID 7135043, 8pp, https://doi.org/10.1155/2022/7135043.

Zhang, D., Han, X. and Deng, C. (2018) 'Review on the research and practice of deep learning and reinforcement learning in smart grids', *CSEE Journal of Power and Energy Systems*, Vol. 4, No. 3, pp.362–370.