Low-complexity Anomaly Detection Method based on Feature Importance using Shapley Value

Joohong Rheey and Hyunggon Park
Graduate Program in Smart Factory
Dept. Electronic and Electrical Engineering
Ewha Womans University, Seoul, Republic of Korea
joohong.rheey@ewhain.net, hyunggon.park@ewha.ac.kr

***Abstract*—The increasing popularity of Internet of Things (IoT) devices has brought significant security challenges to IoT networks. However, most deep learning-based anomaly detection solutions often require high computation performance so that it is difficult to be implanted on low-end IoT devices with limited power and memory capacity. In this paper, we propose a low-complexity network anomaly detection method based on feature selection using the Shapley value for the Isolation Forest algorithm. The proposed feature selection method using the Shapley value can reduce the dimension of input data, thereby improving the performance with reduced computational complexity. We provide simulation results to demonstrate the effectiveness of the proposed method. The results show that the proposed method based on Isolation Forest achieves comparable performance to the deep learning method based on neural networks while using fewer dimensions than the deep learning method.**

***Index Terms*—Anomaly detection, Dimensionality reduction, Low-complexity, Feature importance, Shapely value**

I. Introduction

As communication and network technologies based on vertical services are developed, the Internet of Things (IoT) technology has been rapidly growing. The increasing popularity of the IoT has posed a significant challenge to the security of IoT networks [1]. While IoT devices are often located in easily accessible areas, attackers can easily access them [2]. Therefore, network anomaly detection algorithms should be implemented in IoT devices.

Recently, there has been active research on network anomaly detection algorithms based on deep learning techniques, such as an autoencoder, to detect network intrusion attempts [3], [4]. Since such anomaly detection solutions based on deep learning techniques require significantly high computing powers, they cannot be adopted in low-end IoT devices such as smart light bulbs and sensors with limited hardware capabilities [5]. Therefore, developing low-complexity anomaly detection solutions that can operate efficiently on low-end IoT devices is crucial.

Isolation Forest algorithm [6] is a popular unsupervised machine learning algorithm used for anomaly detection. Unlike other anomaly detection algorithms based on neural network structure, the Isolation Forest approach does not rely on distance or density calculations to identify anomalies [7]. Therefore, it is able to rapidly distinguish exceptional data from normal data with low linear time complexity using isolation trees.

In order to improve the Isolation Forest algorithm, we reduce the dimension of the input data based on the feature selection using the Shapley value [8]. The feature selection can identify and select only the important features from the input data, thereby reducing its dimensionality. Reduced dimension of the input data can improve the performance of anomaly detection of the Isolation Forest algorithm with lower computational complexity [9].

The rest of the paper is organized as follows. In Section II, we propose the low-complexity algorithm including the feature selection process, and present the anomaly detection that improves the Isolation Forest algorithm. The simulation results are shown in Section III and the conclusions are drawn in Section IV.

II. Isolation Forest-based Low-complexity Anomaly Detection system

In this section, we present a low-complexity anomaly detection algorithm that improves the Isolation Forest algorithm by including only important features to detect anomalies. An overview of the proposed method is described in Fig. 1.

To determine the feature importance score, we employ the Shapley value, which is a solution concept in game theory that measures the average marginal contribution of each player to the coalition value [8]. By considering the features as players in a coalition cooperating for accurate predictions, the concept of the Shapley value is used for feature selection [10], [11].

Let $\mathbf{F} = \{\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_k\}$ be a set of features of dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ of n instances from a k-variate distribution. $S(\subseteq N \backslash \{i\})$ denotes a subset excluding i-th feature. An importance score based on Shapley value of i-th feature in the set N is expressed as

$$\phi_i = \sum_{S \subseteq N \backslash \{i\}} \frac{|S|!(|N| - |P| - 1)!}{|N|!} (v(S \cup \{i\}) - v(S)), \quad (1)$$

where $v(S)$ is the value of subset S, and $|\cdot|$ denotes the cardinality of a set.

The larger the value of ϕ, the more important feature is considered to be. In the proposed algorithm, features are selected based on the importance score computed by (1).

Let $m(\leq k)$ be the number of selected features, which can be set by considering the desired level of complexity. Only top m features with the highest importance score ϕ_i among k

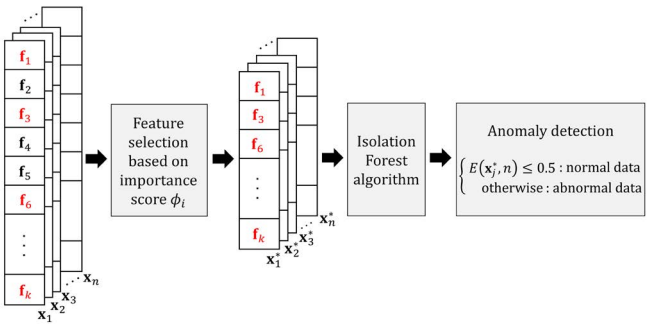

Fig. 1. Overview of the proposed anomaly detection method.

features can be chosen. Since the input data of the Isolation Forest algorithm is transformed from $\mathbf{x} \in \mathbb{R}^k$ to $\mathbf{x}^* \in \mathbb{R}^m$, the computational complexity required for building an isolation tree can be reduced.

The Isolation Forest algorithm builds an ensemble of isolation trees that have an equivalent structure to Binary Search Tree (BST). To construct an isolation tree for a given dataset $\mathbf{X}^* = \{\mathbf{x}_1^*, \mathbf{x}_2^*, \cdots, \mathbf{x}_n^*\}$, we define A as a node of the tree. A is either an external node with no child or an internal node with one test and exactly two child nodes (A_l, A_r), where A_l and A_r denote the left child node and the right child node. Constructing an isolation tree is a process of recursively dividing $\mathbf{X}$ by randomly selecting an attribute q and a split value p until the tree reaches a height limit, $|\mathbf{X}| = 1$, or all data in $\mathbf{X}$ have the same values. The test $q < p$ divides data points into A_l and A_r, where data points that satisfy the test are placed in the left subtree, and those that do not are placed in the right subtree. The resulting isolation tree is a proper binary tree, where each node has either zero or two child nodes. Assuming that all instances are distinct, each instance is isolated to an external node when the isolation tree is fully grown. For the fully grown isolation tree, the number of external nodes is n, the number of internal nodes is $n-1$, and the total number of nodes in the isolation tree is $2n-1$. Therefore, the memory requirement is bounded and grows linearly with n.

To detect anomalies, we compute an anomaly score E. As the tree of the Isolation Forest algorithm is trained with normal data, the abnormal data is closer to the root node of the tree, and the normal data is far from the root node [7]. Therefore, the anomaly score E is derived from the path length for each test instance. Path length $L(\mathbf{x}_j^*)$ is defined as the number of edges traversed by an instance $\mathbf{x}_j^*$ in an isolation tree from the root node to an external node. To rank the data points and detect anomalies, we can use their path lengths L or anomaly scores E. Since isolation trees have a similar structure to BST, the average path length for external node terminations can be estimated using the same method as an unsuccessful search in BST. We can use BST analysis to estimate the average path length of the isolation tree. As in [12], the average path length of unsuccessful searches in BST for the given dataset of n instances can be expressed as

$$b(n) = 2H(n-1) - (2(n-1)/n), \tag{2}$$

where $H(\cdot)$ is the harmonic number and it can be estimated by Euler's constant. We use $b(n)$ to normalize $L(\mathbf{x}_j^*)$, as it represents the average of $L(\mathbf{x}_j^*)$ given n. The anomaly score of the Isolation Forest algorithm is defined as

$$E(\mathbf{x}^*, n) = 2^{-\frac{\overline{L}(\mathbf{x}_j^*)}{b(n)}}, \tag{3}$$

where $\overline{L}(\mathbf{x}_j^*)$ is the average of $L(\mathbf{x}_j^*)$ from a collection of isolation trees, and $0 < E(\mathbf{x}_j^*, n) \le 1$ for $0 < L(\mathbf{x}_j^*) \le n-1$. To detect anomalies, we pass the test instances through isolation trees to obtain an anomaly score for each instance. For example, if an anomaly score $E(\mathbf{x}_j^*, n)$ of an instance $\mathbf{x}_j^*$ is smaller than 0.5, it is regarded as a normal data point.

III. Simulation

A. Simulation Setup

In the simulation, the NSL-KDD dataset [13] and the CIC-IDS2017 dataset [14] are used. These are popular real-world datasets collected in the LAN networks. The normal data are divided into a train set and a test set in an 8:2 ratio. The number of features of the NSL-KDD and the CIC-IDS2017 datasets is $k = 40$ and $k = 65$. We set the controllable parameter m to $1/10$ of k, i.e, $m = 4$ for the NSL-KDD dataset and $m = 6$ for the CIC-IDS2017 dataset.

B. Performance Analysis

We consider the input data with randomly sampled features to confirm the role of feature selection based on the importance score. The number of features in randomly sampled data sets is equal to m.

Fig. 2 and Fig. 3 show ROC (Receiver Operating Characteristic) curves. The results provided in Fig. 2 and Fig. 3 are obtained from 15 independent simulations using 15 random seeds. It is clearly observed that the proposed method achieves higher detection rates than the random sampling method. This is because the feature selection process using the Shapley value leads to selecting significant features rather than unnecessary features.

C. Performance Comparisons

We compare the performance of the proposed solution with that of the autoencoder which is the most widely used deep learning method for anomaly detection. In the autoencoder, an encoder learns a representation from the input data and a decoder tries to reconstruct from the representation to the output data with the goal of minimizing reconstruction error between the input and output data [15]. The autoencoder includes complex nonlinear computation through an activation function.

We consider two versions of the autoencoder with different levels of complexity.

- Stacked autoencoder (SAE): The autoencoder with a stacked structure that has four layers. For the NSL-KDD

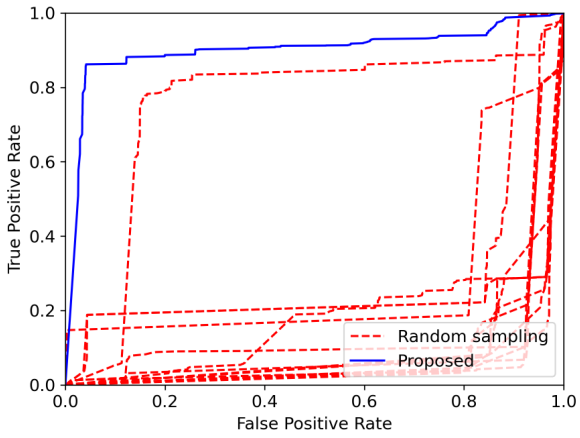

Fig. 2. ROC curves of the anomaly detection using Isolation Forest on the NSL-KDD dataset.

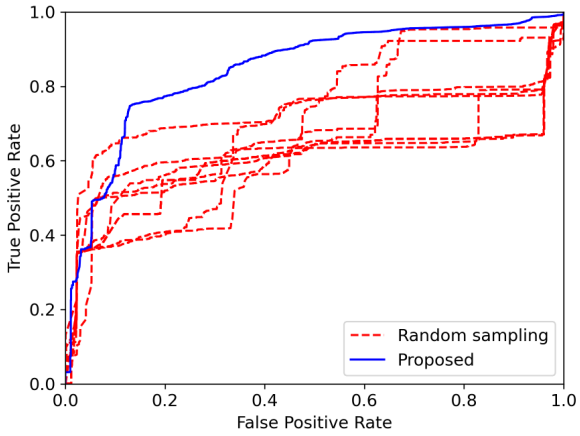

Fig. 3. ROC curves of the anomaly detection using Isolation Forest on the CIC-IDS2017 dataset.

dataset, the number of units in the first hidden layer and the second layer are 20 and 10. For the CIC-IDS2017 dataset, the number of units in the first hidden layer and the second layer are 40 and 20. The Stacked autoencoder uses the original input data with k dimensions.

- Basic autoencoder (AE): The autoencoder with a simple structure that has an input layer with m units, one hidden layer with two units, and an output layer with m units. The dimension of input data of the basic autoencoder is the same as the proposed method.

The activation function and loss function of the autoencoder are ELU and MSE (Mean Squared Error), respectively. For computing the anomaly score of the autoencoder, MAE (Mean Absolute Error) is used.

Table I presents the results of anomaly detection using four performance metrics. Across all metrics and datasets, the performance of the proposed method is slightly inferior to that of the stacked autoencoder while the inference time of the proposed method is much shorter than that of others. Notably, in terms of accuracy, the proposed approach achieves 96% and 94% compared to the stacked autoencoder. Furthermore, the proposed method outperforms the basic autoencoder on the CIC-IDS2017 dataset. It is noteworthy that these results are achieved using an Isolation Forest-based anomaly detection method with low computational complexity, using only approximately $1/10$ of the data compared to the stacked autoencoder. Hence, the feasibility of the proposed method for implementation on low-end devices is demonstrated.

TABLE I
PERFORMANCE COMPARISONS

NSL-KDD dataset					
	Accuracy	Precision	Recall	F1-score	Inference time
Proposed	**0.92**	**0.71**	**0.83**	**0.76**	**0.53 sec**
SAE	0.96	0.98	0.97	0.98	2.41 sec
AE	0.92	0.96	0.94	0.95	1.63 sec

CIC-IDS2017 dataset					
	Accuracy	Precision	Recall	F1-score	Inference time
Proposed	**0.80**	**0.70**	**0.84**	**0.76**	**6.09 sec**
SAE	0.85	0.95	0.75	0.84	22.06 sec
AE	0.62	0.65	0.60	0.62	13.71 sec

IV. Conclusion

In this paper, we propose a low-complexity anomaly detection method that includes a feature selection process. The proposed approach overcomes the limitations of conventional anomaly detection methods, which require high computational complexity. By calculating the Shapley value of each feature, we obtain an importance score and reduce the dimension of the input data. Only the selected features with high importance scores are used as input for the Isolation Forest algorithm. The simulation results confirm that the proposed method improves scalability and efficiency without sacrificing the critical capability of the model.

Acknowledgment

This work was supported in part by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2021-0-00739, Development of Distributed/Cooperative AI based 5G+ Network Data Analytics Functions and Control Technology), and in part by the Korea Foundation for Women In Science, Engineering and Technology (WISET) grant funded by the Ministry of Science and ICT (MSIT) under the team research program for female engineering students.

References

[1] K. Lee, B. Kim, and J. Cho, "Design and implementation of security system for providing secure boot and firmware update in low-end iot device," *Journal of Korean Institute of Information Scientists and Engineers*, vol. 45, no. 4, pp. 321–331, 2018.

[2] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "Iot security: ongoing challenges and research opportunities," in *2014 IEEE 7th international conference on service-oriented computing and applications*, 2014, pp. 230–234.

[3] W. T. Lunardi, M. A. Lopez, and J.-P. Giacalone, "Arcade: Adversarially regularized convolutional autoencoder for network anomaly detection," *IEEE Transactions on Network and Service Management*, 2022.
[4] X. Xing, X. Jin, H. Elahi, H. Jiang, and G. Wang, "A malware detection approach using autoencoder in deep learning," *IEEE Access*, vol. 10, pp. 25 696–25 706, 2022.
[5] H. Kim, H. Lee, and Y. Lee, "A survey analysis of internet of things security issues and combined service," *Journal of The Korea Society of Computer and Information*, vol. 25, no. 8, pp. 73–79, 2020.
[6] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 8th IEEE international conference on data mining*, 2008, pp. 413–422.
[7] D. Xu, Y. Wang, Y. Meng, and Z. Zhang, "An improved data anomaly detection method based on isolation forest," in *2017 10th international symposium on computational intelligence and design (ISCID)*, vol. 2, 2017, pp. 287–291.
[8] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games II*, H. W. Kuhn and A. W. Tucker, Eds. Princeton: Princeton University Press, 1953, pp. 307–317.
[9] L. Puggini and S. McLoone, "An enhanced variable selection and isolation forest based methodology for anomaly detection with oes data," *Engineering Applications of Artificial Intelligence*, vol. 67, pp. 126–135, 2018.
[10] S. Cohen, G. Dror, and E. Ruppin, "Feature selection via coalitional game theory," *Neural Computation*, vol. 19, no. 7, pp. 1939–1961, 2007.
[11] A. Catav, B. Fu, Y. Zoabi, A. L. W. Meilik, N. Shomron, J. Ernst, S. Sankararaman, and R. Gilad-Bachrach, "Marginal contribution feature importance-an axiomatic approach for explaining data," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1324–1335.
[12] B. R. Preiss, *Data Structure and Algorithms: With Object-oriented Design Patterns in Java*. John Wiley & Sons, 1999.
[13] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6.
[14] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.
[15] J. Rheey, D. Choi, and H. Park, "Adaptive loss function design algorithm for input data distribution in autoencoder," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, 2022, pp. 489–491.