Human-centric Computing and Information Sciences

September 2021 | Volume 11

www.hcisjournal.com

KIPS Korea Information Processing Society

KIPS CSWRG Korea Information Processing Society Computer Software Research Group

HCIS | Human-centric Computing and Information Sciences

RESEARCH(Research Manuscript) Open Access

Human-centric Computing and Information Sciences (2021) 11:35
DOI: https://doi.org/10.22967/HCIS.2021.11.035
Received: July 8, 2021; Accepted: August 25, 2021; Published: September 15, 2021

Detection of Adversarial Attacks in AI-Based Intrusion Detection Systems Using Explainable AI

Erzhena Tcydenova, Tae Woo Kim, Changhoon Lee, and Jong Hyuk Park*

Abstract

With the tremendous increase in networking devices connected to the Internet, network security is recognized as an important issue. Intrusion detection systems (IDSs) are one of the important components of network security. There are several methods for implementing an IDS, and one is machine learning. The machine learning performance of IDSs is evolving to a very large extent and is being used in real IDSs. However, recent studies showed that machine learning classification models are vulnerable to adversarial attacks. In this paper, we propose an adversarial attack detection framework in machine learning-based explainable AI intrusion detection systems. The proposed framework consists of two phases: initialization and detection. In the initialization phase, we train an IDS based on a support vector machine classification model and extract explanations of the *Normal* data records from the dataset using LIME (local interpretable model-agnostic explanations). Based on the resulting explanations, results of the classification by the trained IDS are analyzed during the detection phase by explanation to detect an adversarial attack. We evaluate the proposed method using the NSL-KDD dataset.

Keywords

Adversarial Attacks, Explainable AI, Intrusion Detection Systems, Machine Learning

1. Introduction

Every year, the use of cyber networks and the Internet is growing enormously, and security and privacy are highly important concerns. It is expected that the number of devices connected to the Internet will be close to a trillion by 2022 [1]. With the growth of networks and data transferred through the Internet, the number of cyberattacks continues to grow as well, so the detection of attacks is essential in information systems [2]. Intrusion detection systems (IDSs) are one solution to these issues, and they aim to detect malicious traffic and unauthorized use to provide a more secure environment [3, 4].

There are different ways to implement an IDS. One of the most popular and widely used IDSs incorporates machine learning techniques. To implement an IDS, different types of machine learning classification algorithms are used, such as k-nearest neighbors, decision trees, and support vector

***Corresponding Author:** Jong Hyuk Park (jhpark1@seoultech.ac.kr)
Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul, Korea

machines (SVMs). Machine learning has proven to be a very powerful method in various fields, and it has a huge number of different applications [5, 6]. However, recent studies showed that machine learning models are vulnerable to adversarial attacks [7–10]. Adversarial attacks work by modifying the original data slightly which leads to misclassification and significantly reduces performance of target model [9]. Adversarial attacks have been successfully conducted in machine learning IDSs and significantly reduced performance of those IDSs [8]. There are several approaches to detect adversarial attacks, and one of them uses explainable artificial intelligence (XAI). XAI is a concept of AI that aims to provide transparency, causality, fairness, and safety regarding AI decisions [11]. It was successfully used for the detection of adversarial attacks in image classification tasks, but there has not been any application in IDSs.

In this paper, we propose a framework for the detection of adversarial attacks in machine learning-based IDSs using XAI. Our framework is divided into two phases: the initialization phase and the detection phase. In the initialization phase, we train a SVM classification model-based IDS using the NSL-KDD training dataset. Then, we extract explanation of the *Normal* data records from the training dataset. Based on this explanation, we define a set of features that contains features that contributed the most to classify data as *Normal*. In the detection phase, the trained IDS monitors incoming network traffic, and if an intrusion is detected, the IDS sends an alert. However, if the IDS classifies input data as *Normal* an additional step is required. In this step, we extract explanation of this data and check whether features from the explanation are in the set that we extracted in the initialization phase. For proof-of-concept of our framework, we use the local interpretable model-agnostic explanations (LIME) model to explain data, and we use projected gradient descent (PGD) attack to generate adversarial examples. LIME is one of the most popular explanation models along with Shapley additive explanation (SHAP). We chose LIME because of its easy-to-interpret structure. PGD attack was chosen in the experiments because it is one of the most efficient and widely researched methods. Experiments conducted on adversarial examples and *Normal* test data showed that the set of features that contribute to the classification of adversarial examples to the *Normal* class tend to have noise features.

The paper is organized as follows. Section 2 introduces background and related works. Section 3 proposes the adversarial detection framework and shows the detailed architecture. Section 4 presents experiments and results of the proposed framework. Finally, Section 5 concludes the paper.

2. Related Work

2.1 Intrusion Detection System

An IDS is a software or hardware system that aims to detect intrusions in a network. Intrusions refer to any type of malicious activity that attempts to compromise the security of an information system. An IDS's role is to monitor all activities within the network that cannot be identified by a firewall and to detect intrusions in order to send an alarm to an administrator of the system. The role of IDSs is very important to achieve security requirements for information systems, such as availability, integrity, and confidentiality [3, 12].

There are several types of IDS implementation techniques, and one is machine learning. Machine learning is a branch of AI, and machine learning techniques automatically learn from data by finding hidden patterns. Several machine learning models have been used for IDSs, such as k-nearest neighbor or clustering methods, neural networks, decision trees, or SVMs [13]. SVMs are widely used for IDS construction, and this method is one of the most efficient [14, 15]. The general architecture of machine learning IDSs is presented in Fig. 1.

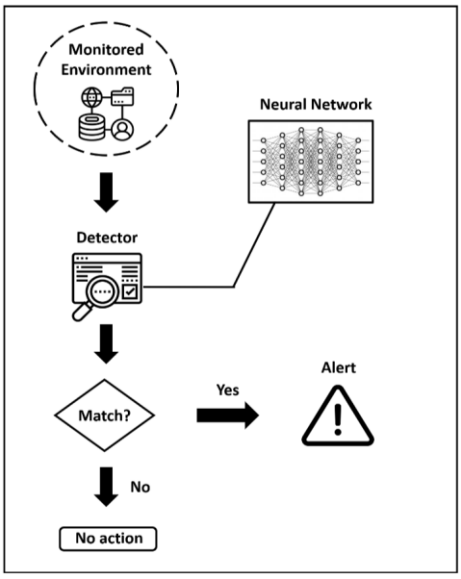

Fig. 1. Artificial intelligence based intrusion detection systems.

2.2 Adversarial Attacks

Machine learning achieved great performance in various applications, including IDSs [16]. But recently it was discovered that machine learning models like deep neural networks (DNNs) can be vulnerable to adversarial attacks that can significantly reduce performance of models. Szegedy et al. [10] proposed the fast gradient sign method (FGSM) attack that works by adding some small noise to the original data that would seem insignificant to humans but leads to misclassification of the trained model. FGSM performs a single-step attack using the loss function of the model L and some perturbation parameter ϵ [17]:

$$x_{adv} = \mathrm{x} + \epsilon \cdot \mathrm{sign}\big(\nabla_x \mathrm{L}(\mathrm{x}, \mathrm{y}; \theta)\big)$$

After this research, various adversarial attacks have been proposed. One of them is PGD attack [18], which is a variation of the FGSM attack. PGD works by iterating FGSM several times in the area S:

$$x_{adv} = \Pi_{x+S}(\mathrm{x} + \epsilon \cdot \mathrm{sign}(\nabla_x \mathrm{L}(\mathrm{x}, \mathrm{y}; \theta)))$$

2.3 Explainable AI

AI performance has increased significantly during last years compared to very first models, however, growth of performance led to different problems which is lack of interpretability. The very first AI models were easily interpretable. Current models, though, like DNNs, have great performance but are hard to interpret and can be considered black-box models [19]. Nowadays, AI is used in a huge number of different areas. Some fields, such as the medical sector, consider transparency and explainability of models highly important. In order to provide transparency to complex AI models, a new wave of AI called explainable AI is now sparking interest [20]. XAI is a class of systems that aim to provide transparency and explainability of model's decisions and also to provide insight of model's possible future behavior. There are several studies and methods proposed and developed to give an explanation of AI models [21]. One of them is the LIME method.

2.3.1 Local interpretable model-agnostic explanations

LIME works by using a surrogate model, which is easy to explain. That surrogate model is used to approximate predictions of a black box model. Surrogate models can be any simple and understandable model, such as linear or logistic regressions or decision trees [22]. LIME produces local explanations, which explain concrete individual input (i.e., the reason why the target model makes a particular decision for some particular input) [23]. LIME generates a new dataset by permuting the original data records and then makes decisions about this new dataset using a target model that needs explanation. Then, the surrogate model is trained using the newly generated dataset and gives an explanation of the black box model's decision for some inputs [24].

2.4 Existing Studies

Several successful studies were conducted on the application of adversarial attacks on machine learning-based IDSs. Attacks using adversarial examples were able to reduce performance of models significantly.

Peng et al. [25] proposed a framework for evaluating deep learning-based network IDSs. They implemented four adversarial attacks (PGD attack, momentum iterative FGSM, limited memory Broyden-Fletcher-Goldfarb-Shanno [L-BFGS], and simultaneous perturbation stochastic approximation [SPSA] attacks) on four machine learning models (DNNs, SVM, random forest, and logistic regression) and evaluated the performance of the models under adversarial attacks. Experiments showed that performance of the models significantly reduced as a result of the attacks. Accuracy of the four clean models was about ≈0.750, and when under attack, accuracy was reduced to ≈0.5. Experiments were conducted using the NSL-KDD dataset.

Yang et al. [8] conducted a study to mimic adversarial attacks on DNN-based network IDSs. Authors implemented three adversarial attacks (substitute model attack, zeroth order optimization [ZOO] attack, and generative adversarial nets [GAN]) on DNN, and the attacks were able to reduce performance of the model. Accuracy of the original model was 0.890. ZOO and GAN attacks reduced model performance to ≈0.5; the substitute model attack reduced accuracy to ≈0.7.

There are several studies that propose defense mechanisms against adversarial attacks, but defending IDSs against adversarial attacks is not widely researched. Pawlicki et al. [26] proposed a method for detecting adversarial attacks. Their method works by dividing a dataset for training an IDS and training an adversarial detector. The adversarial detector was trained using a dataset with adversarial samples, and after training it successfully detected adversarial examples.

One method to defend against adversarial attacks is using XAI. Studies on the detection of adversarial attacks using XAI were mostly conducted on an image classification task. Our method, to our knowledge, is the first application of XAI for detection of adversarial attacks in the context of an IDS.

Klawikowska et al. [27] applied XAI local and global explanation methods to analyze adversarial attacks on DNNs that were trained for image classification. Experiments showed that explanation results of the clear original dataset records and explanation of adversarial samples differed a lot and demonstrated that XAI tools can be valuable for analysis.

Amosy and Checkit [28] proposed an adversarial attack detection approach using the SHAP method to identify images whose explanations do not match the predicted class. They evaluated performance of the proposed method using the CIFAR-10 and SVHN datasets and the FGSM, PGD, and Carlini&Wagner (C&W) adversarial attacks. The proposed method improves detection accuracy from ≈70% to over 90%.

Fidel et al. [29] presented a detection method of adversarial examples on DNN using the SHAP method. Their method works by generating XAI signatures using SHAP for both normal and adversarial dataset samples. Then, these signatures are used to train the detector. Evaluation of performance was done using CIFAR-10 and MNIST image recognition datasets and accuracy of detecting adversarial attacks was about ≈97%.

3. Proposed Framework

In this section we propose an architecture for the detection of adversarial attacks in machine learning-based IDSs. In the initialization phase, we train an IDS using a SVM classification model, and then we extract an explanation of *Normal* data records from the dataset using LIME. Based on the explanation, we extract a set of features that defines *Normal* data. In the detection phase, the trained IDS is used to classify network traffic. If it is classified as *Normal*, an extra validation step is required. A set of features generated in the initialization phase is used to detect adversarial examples in a real-time environment by comparing the explanation of the network traffic that is classified as *Normal* with the set of features. The overall architecture is illustrated in Fig. 2.

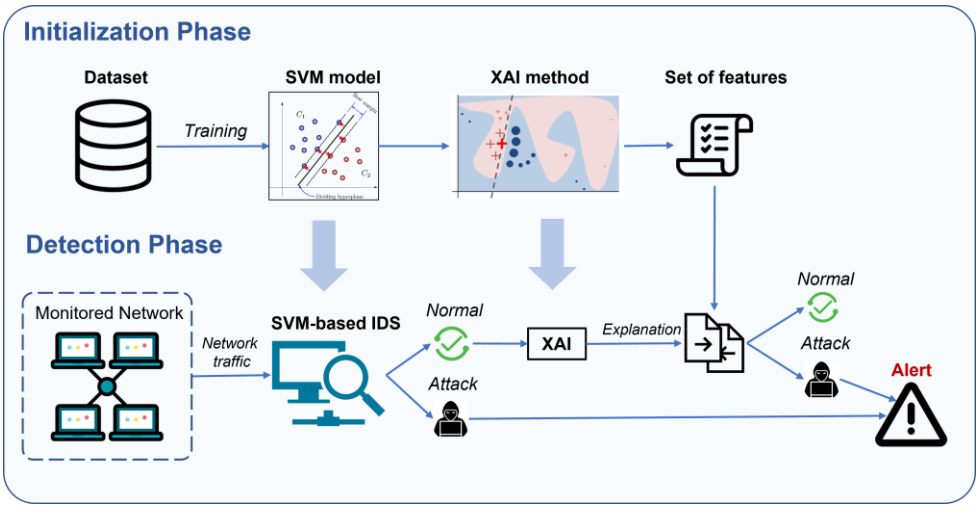

Fig. 2. Architecture of the proposed method.

3.1 SVM-based Intrusion Detection System

A SVM is a widely used supervised machine learning algorithm based on statistical learning theory that can be used for both classification and regression. The main idea of SVM is based on finding the optimal hyperplane so that the difference between classes is maximized. An SVM model is formed by finding support vectors that represent the training data [30].

In the proposed architecture, a SVM classifier SVC from "scikit-learn" machine learning library was used. The model was trained with an "*rbf*" kernel, *random_state* = 0 and with other tuning parameters set by default. Then, the trained model is used to predict new incoming data (monitored network traffic) into two classes: *Normal* and *Attack*. The architecture of the training and prediction phases of our IDS is illustrated in Fig. 3.

The performance of the SVM is evaluated on the NSL-KDD dataset which is a dataset that was proposed to solve issues in the KDD'99 dataset. The KDD'99 dataset is a network connection record set, which was restored from the raw data collected by Lincoln Labs at MIT for an IDS evaluation sponsored by the Defense Advanced Research Projects Agency (DARPA) in 1998. The NSL-KDD contains a training dataset with 21 attacks and a test dataset with 37 attacks. There are five classes in the datasets: normal, probe, denial of service (DoS), user to root (U2R), and remote to local (R2L). Generally, a dataset is divided into two classes, *Normal* and *Attack*. In this paper we conduct binary classification; thus the IDS classifies network traffic into normal data and intrusion data.

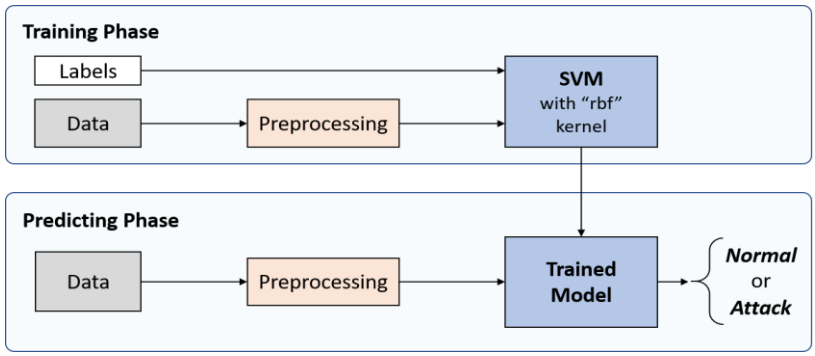

Fig. 3. Training and prediction phases of the IDS.

3.1.1 Data preprocessing

The NSL-KDD dataset has 41 features: three symbolic features, 32 continuous features, and six binary features.

- **Symbolic features:** There are three symbolic features: "*protocol_type*", "*flag*," and "*service*." First, we transform symbolic features into numbers using Label Encoder, and then, we encode the results using One Hot Encoder.
- **Continuous features:** There are 32 continuous features in this dataset: "*duration*," "*src_bytes*," "*dst_bytes*," "*wrong_fragment*," "*num_failed_logins*," etc. We use Standard Scaler as a data normalization method.
- **Binary features:** There are six binary features: "*land"*, "*logged_in*," "*root_shell*," "*su_attempted*," "*is_hot_login*," "*is_guest_login*," and no preprocessing techniques are needed for these features.

After preprocessing, the dataset will have 122 dimensions because the One Hot Encoding method expands the number of features.

3.2 Explanation with LIME

3.2.1. Initialization phase

To detect adversarial examples, we first get an explanation result from the dataset that was used for training our IDS. We use the LIME tool to extract explanations of *Normal* data records from the dataset. Algorithm 1 shows extraction of rules from the training dataset using the LIME explainer. First, we get an Explainer of the training data X_train with its labels Y_train and all features of this dataset. Then, we specify the prediction function of our trained model. Afterwards, we have to specify the number *N* of features that we want to extract. This number of features represents the most important features that we are going to use for explanation. After selecting the number of features, the explanation of *Normal* data records of the dataset using the model's prediction function is complete. Explanation returns a set of *N* most important features of all selected *Normal* data records. Then, we define set *S* of features that become an explanation of the *Normal* data.

3.2.2. Detection phase

In the detection phase, monitored network traffic is first analyzed by the IDS. If the input traffic is classified as *Attack*, the IDS sends *Alert*. If it is classified as *Normal*, then another step is required to verify the IDS's decision. In this step, an explanation of this data is extracted using the Explainer from the initialization phase. The Explainer returns *N* important features about what the IDS's decision was based on. Then, we check if these features are in the set *S* extracted in the initialization phase. If there is more than one feature that does not belong to the set *S*, this input traffic is classified as *Attack* and *Alert* is sent. If all features belong to the set, input traffic is classified as *Normal*, and no action is needed (Algorithm 2).

Algorithm 1. Extraction of important features of the *Normal* data

Input: *Model, X_train, Y_train, AllFeatures*
Output: Set of important features: *S*
Explainer ← *LimeTabularExplainer(X_train, Y_train, AllFeatures)*
prediction_function ← *Model's prediction function*
Number of important features to extract: *N* ← *10*
for *"Normal"* record *nrml* in *X_train* **do**
    explanation[nrml] ← *Explainer(X_train[nrml], prediction_function, N)*
end for
S ← *ExtractFeatures(explanation)*
return *S*

Algorithm 2. Detection of attacks by IDS and Explainer

Input: Input network data N_I, set of features *S*
Output: *Normal* or **ALERT**
Decision ← *IDS(N_I)*
if *Decision == Attack* **then**
    return ALERT
else if *Decision == Normal* **then**
    explanation[N_I] ← *Explainer(N_I, prediction_function, N)*
    if *explanation[N_I]* ϵ *S* **then**
        return *Normal*
    else
        return ALERT
    end if
end if

4. Experiments and Results

4.1. Performance Evaluation of IDS

For evaluation of our model's performance, we use the following evaluation metrics: accuracy, precision, recall, and F1-score. Here, true positive (TP) indicates the number of correctly classified attacks. True negative (TN) is the number of correctly classified normal behavior events. False positive (FP) is the number of attacks wrongly classified as normal behavior events. False negative (FN) represents the number of normal behavior events wrongly classified as attacks [24].

- **Accuracy**: This is the ratio of correctly classified records to the entire dataset.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision**: This is the ratio of records correctly classified as an attack to all records.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**: This is the ratio of records correctly classified as an attack to all the attack records classified as an attack.

$$Recall = \frac{TP}{TP + FN}$$

- **F1-Score**: F1-score is a measure of precision and recall at the same time.

$$F1 - Score = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

Table 1 shows the performance results of the SVM model used for IDS.

Table 1. SVM performance results

Model	Accuracy	Precision	Recall	F1-score
SVM Classifier	0.9684	0.9589	0.9868	0.9726

4.2. Detection of Adversarial Attacks

4.2.1. Initialization phase

First, we extract features that define *Normal* data using Algorithm 1. The inputs of the algorithm are the SVM model, training dataset from the NSL-KDD, and 122 features of the dataset. The 122 features are the features after the data preprocessing phase, and they represent 41 original features. Then using the LIME explainer and the model's prediction function, we get an explanation of *Normal* records within the dataset. The NSL-KDD training dataset consists of 125,973 data records and 67,343 of them are labeled as *Normal*. We conducted experiments using 1,000 data records from the dataset labeled as *Normal* and extracted the explanation result for all of them. We chose 1,000 data records for evaluation of our framework's performance due to time consumption, so this number can be expanded. The number of features extracted for every data record is 10. This means that the explanation result of a single data record has 10 features that had the biggest impact on classification to *Normal* class. Using explanation results from 1,000 data records, we extracted the set of features *S* that contains all the important features from the *Normal* data. In this experiment, we extracted 65 features out of 122 that are important in classifying *Normal* data records. These features are shown in Table 2.

Table 2. Features extracted by Algorithm 1

Extracted features			
land	**wrong_fragment**	**urgent**	**hot**
num_failed_logins	root_shell	su_attempted	num_file_creations
num_shells	num_access_files	is_host_login	service_IRC
service_X11	service_aol	service_ctf	service_discard
service_echo	service_efs	service_exec	service_gopher
service_harvest	service_hostnames	service_http	service_http_2784
service_http_443	service_http_8001	service_imap4	service_klogin
service_kshell	service_ldap	service_link	service_login
service_mtp	service_name	service_netbios_dgm	service_netbios_ns
service_netbios_ssn	service_nntp	service_ntp_u	service_other
service_pm_dump	service_pop_2	service_pop_3	service_printer
service_red_i	service_remote_job	service_rje	service_shell
service_sql_net	service_sunrpc	service_supdup	service_systat
service_tftp_u	service_tim_i	service_urh_i	service_urp_i
service_uucp	service_whois	flag_OTH	flag_RSTO
flag_RSTOS0	flag_S1	flag_S2	flag_S3
flag_SH			

4.3. Detection Phase

To evaluate detection performance, we generated adversarial examples using a PGD attack. We generated adversarial examples of the test data and used 10 FN classification results (Attacks that were classified as *Normal*) from adversarial examples and 10 *Normal* test entries to conduct experiments. We selected 10 data records for each case to evaluate performance of our framework. First, we got explanation results using LIME on all the data, both adversarial and *Normal* test data. Then, we analyzed if the extracted features belonged to the set *S*. If all features of data belonged to the set, then this data was considered *Normal*. If there was even one feature that set *S* did not contain, then we considered this data as adversarial data and classified it as attack.

4.3.1. Experiments on adversarial examples

Fig. 4 shows the explanation result of an adversarial example. In this figure, 10 most important features that contributed to the Normal class are shown: "flag_S1," "service_ntp_u," "service_IRC," "service_urp_i," "flag_S2," "dst_host_same_srv_rate," "same_srv_rate," "flag_SF," "service_tim_i," and "service_kshell." The IDS classified this adversarial example as *Normal* data even it is an *Attack*. However, three features, "same_srv_rate," "dst_host_same_srv_rate," and "flag_SF," do not belong to the set *S*, which means that these features are not the features that contribute to the Normal class.

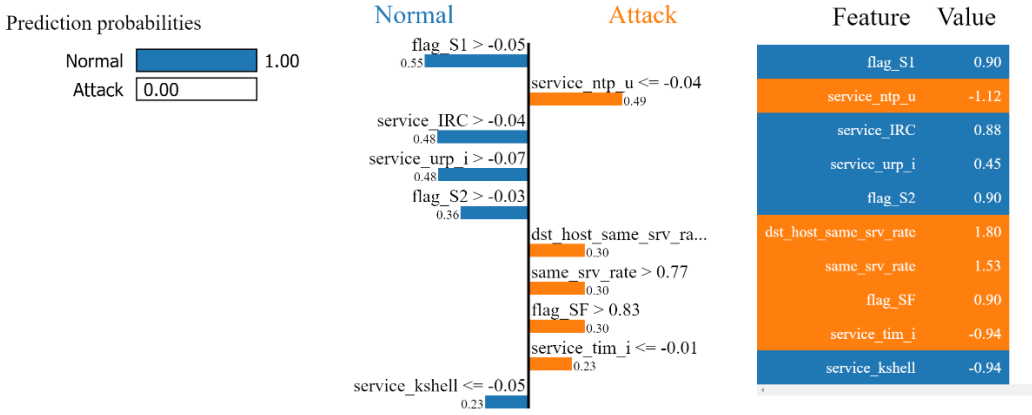

Fig. 4. Explanation of adversarial example 1.

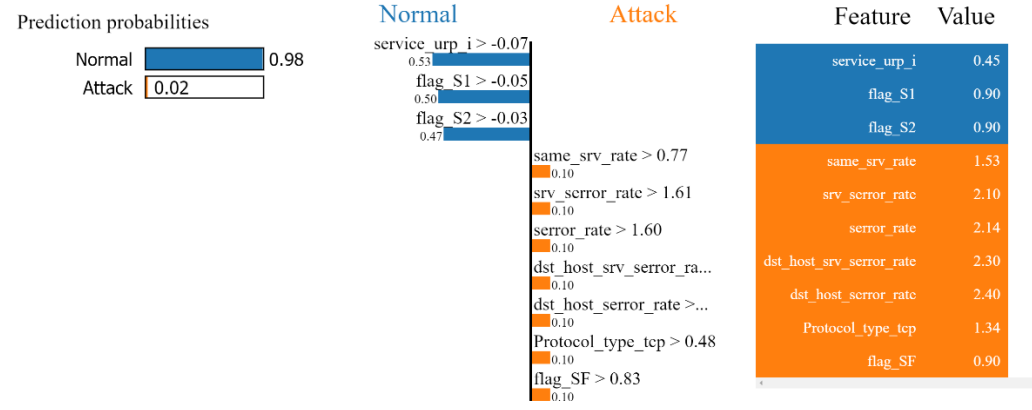

Fig. 5. Explanation of adversarial example 2.

Another adversarial example's explanation results are shown in Fig. 5. This data is also being classified as *Normal* even when it is an *Attack*. There are 10 features that were extracted: "service_urp_i,"

"flag_S1," "flag_S2," "same_srv_rate," "srv_serror_rate," "serror_rate," "dst_host_srv_serror_rate," "dst_host_serror_rate," "Protocol_type_tcp," and "flag_SF." However, there are seven features out of 10 that do not belong to the set *S*: "same_srv_rate," "srv_serror_rate," "serror_rate," "flag_SF," "dst_host_serror_rate," "Protocol_type_tcp," and "dst_host_srv_serror_rate." Even if the IDS did not detect this attack, by explanation of the prediction we can detect the abnormality.

By using the proposed method, all 10 adversarial examples were detected. Results of experiments showed that by using a proposed framework we can detect adversarial attacks that were classified by an IDS as *Normal*.

4.3.2. Experiments on *Normal* test data

We conducted experiments on *Normal* test data that was not used to extract features of set *S* to check whether the explanation of real *Normal* data had features from the set *S* and did not produce FPs. In Fig. 6, explanation of *Normal* data is shown. There are 10 features: "service_IRC," "service_X11," "flag_S1," "flag_OTH," "flag_S2," "num_failed_logins," "service_red_i," "su_attempted," "service_rje," and "flag_S3" and all these features belong to the set *S*, which means that these features are real features that define *Normal* data.

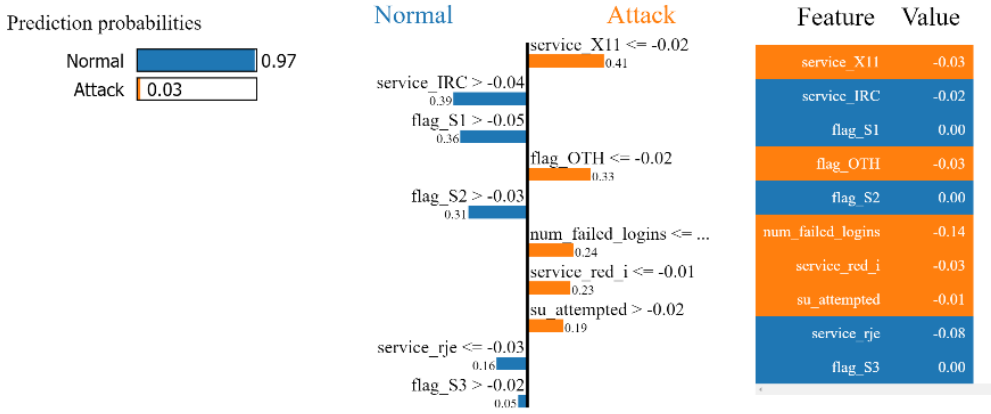

Fig. 6. Explanation of *Normal* example 1.

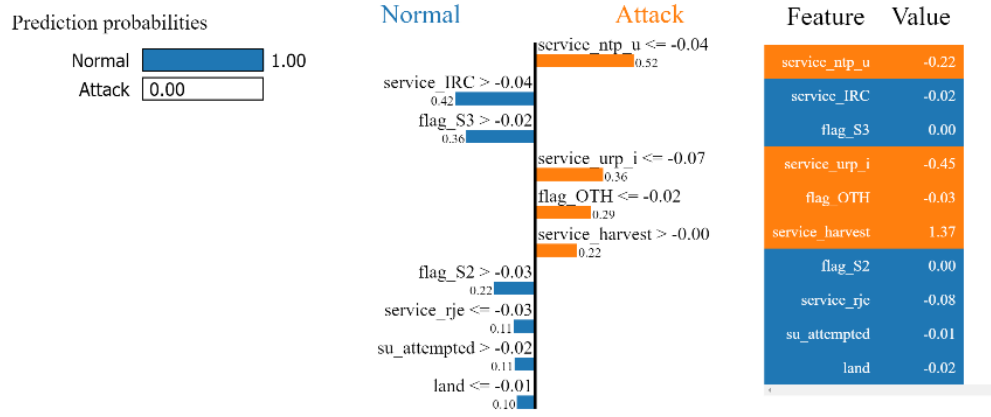

Fig. 7. Explanation of adversarial example 2.

In another experiment shown in Fig. 7 on *Normal* test data, we extracted 10 features from the explanation: "service_IRC," "service_X11," "flag_S1," "flag_OTH," "flag_S2," "num_failed_logins," "service_red_i," "su_attempted," "service_rje," and "flag_S3" and they also belong to the set *S*.

The proposed method was able to classify *Normal* test data without FPs for all 10 examples. By conducting experiments on *Normal* test data examples, we showed that our proposed framework not only is able to detect an adversarial attack, but also correctly classifies new *Normal* data that was not used for producing an explanation set.

5. Conclusion

This paper presented a framework for the detection of adversarial examples in machine learning-based IDSs using XAI. The proposed framework consists of two phases: an initialization phase and a detection phase. In the initialization phase, an IDS based on a SVM model is trained to detect network intrusions. Using the training set data, we get explanations of the *Normal* data records, and based on these explanations, we define a set of features that are important for classification of *Normal* data. In the detection phase, network traffic is monitored using a previously trained IDS. If it detects an intrusion, it sends an alert. If the data is *Normal*, it goes through explanation and the result of the explanation is then compared to the set of features extracted from the initialization phase. If features extracted by explanation belong to the set, it is considered as truly *Normal* data. If it is not, an alert is sent. We conducted experiments to evaluate the performance of the IDS and the detection performance of the framework. Using our framework, it was able to detect 10 out of 10 adversarial examples. Also, experiments on *Normal* test data not used to produce the set of features showed that the proposed framework successfully classifies them as *Normal* data.

Author's Contributions

Conceptualization, ET. Methodology, ET, TWK. Validation, ET. Investigation, ET, TWK. Writing of the original draft, ET. Writing of the review and editing, ET, CL, JHP. Supervision, JHP. Project administration, CL, JHP. Funding acquisition, CL. All authors have read and agreed to the published version of the manuscript.

Funding

This research was supported by Energy Cloud R&D Program (No. 2019M3F2A1073386) through the National Research Foundation of Korea (NRF), both funded by the Ministry of Science and ICT.

Competing Interests

The authors declare that they have no competing interests.

References

[1] L. Santos, C. Rabadao, and R. Goncalves, "Intrusion detection systems in Internet of Things: a literature review," in *Proceedings of 2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*, Caceres, Spain, 2018, pp. 1-7.

[2] W. B. Kim and I. Y. Lee, "Survey on data deduplication in cloud storage environments," *Journal of Information Processing Systems*, vol. 17, no. 3, pp. 658-673, 2021.

[3] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, article no. 20, 2019. https://doi.org/10.1186/s42400-019-0038-7

[4] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, article no. 102419, 2020. https://doi.org/10.1016/j.jisa.2019.102419

[5] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges," *Archives of Computational Methods in Engineering*, vol.

28, no. 4, pp. 3211-3243, 2021.
[6] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, "Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection," *IEEE Access*, vol. 6, pp. 33789-33795, 2018.
[7] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: reliable attacks against black-box machine learning models," 2017 [Online]. Available: https://arxiv.org/abs/1712.04248.
[8] K. Yang, J. Liu, C. Zhang, and Y. Fang, "Adversarial examples against the deep learning based network intrusion detection systems," in *Proceedings of 2018 IEEE Military Communications Conference (MILCOM)*, Los Angeles, CA, 2018, pp. 559-564.
[9] H. Xu, Y. Ma, H. C. Liu, D. Deb, H. Liu, J. L. Tang, and A. K. Jain, "Adversarial attacks and defenses in images, graphs and text: a review," *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 151-178, 2020.
[10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 2016, pp. 2818-2826.
[11] D. Doran, S. Schulz, and T. R. Besold, "What does explainable AI really mean? A new conceptualization of perspectives," 2017 [Online]. Available: https://arxiv.org/abs/1710.00794.
[12] V. Mohammadi, A. M. Rahmani, A. M. Darwesh, and A. Sahafi, "Trust-based recommendation systems in Internet of Things: a systematic literature review," *Human-centric Computing and Information Sciences*, vol. 9, article no. 21, 2019. https://doi.org/10.1186/s13673-019-0183-8
[13] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: a survey," *Applied Sciences*, vol. 9, no. 20, article no. 4396, 2019. https://doi.org/10.3390/app9204396
[14] J. C. S. Sicato, S. K. Singh, S. Rathore, and J. H. Park, "A comprehensive analyses of intrusion detection system for IoT environment," *Journal of Information Processing Systems*, vol. 16, no. 4, pp. 975-990, 2020.
[15] S. Shokat, R. Riaz, S. S. Rizvi, A. M. Abbasi, A. A. Abbasi, and S. J. Kwon, "Deep learning scheme for character prediction with position-free touch screen-based Braille input method," *Human-centric Computing and Information Sciences*, vol. 10, article no. 41, 2020. https://doi.org/10.1186/s13673-020-00246-6
[16] J. S. Park and J. H. Park, "Enhanced machine learning algorithms: deep learning, reinforcement learning, and q-learning," *Journal of Information Processing Systems*, vol. 16, no. 5, pp. 1001-1007, 2020.
[17] F. Wu, R. Gazo, E. Haviarova, and B. Benes, "Efficient project gradient descent for ensemble adversarial attack," 2019 [Online]. Available: https://arxiv.org/abs/1906.03333.
[18] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017 [Online]. Available: https://arxiv.org/abs/1706.06083.
[19] A. Rai, "Explainable AI: from black box to glass box," *Journal of the Academy of Marketing Science*, vol. 48, no. 1, pp. 137-141, 2020.
[20] A. Holzinger, "From machine learning to explainable AI," in *Proceedings of 2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, Kosice, Slovakia, 2018, pp. 55-66.
[21] A. B. Arrieta, N. Diaz-Rodriguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, et al., "Explainable Artificial Intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82-115, 2020.
[22] [22] D. Gunning, "Explainable Artificial Intelligence (XAI)," 2017 [Online]. Available: https://www.cc.gatech.edu/~alanwags/DLAI2016/(Gunning) IJCAI-16 DLAI WS.pdf.
[23] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should i trust you?" Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 2016, pp. 1135-1144.
[24] M. Wang, K. Zheng, Y. Yang, and X. Wang, "An explainable machine learning framework for intrusion detection systems," *IEEE Access*, vol. 8, pp. 73127-73141, 2020.
[25] Y. Peng, J. Su, X. Shi, and B. Zhao, "Evaluating deep learning based network intrusion detection system in adversarial environment," in *Proceedings of 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, Beijing, China, 2019, pp. 61-66.
[26] M. Pawlicki, M. Choras, and R. Kozik, "Defending network intrusion detection systems against adversarial evasion attacks," *Future Generation Computer Systems*, vol. 110, pp. 148-154, 2020.

[27] Z. Klawikowska, A. Mikołajczyk, and M. Grochowski, “Explainable AI for inspecting adversarial attacks on deep neural networks,” in *Artificial Intelligence and Soft Computing*. Cham, Switzerland: Springer, 2020, pp. 134-146.

[28] O. Amosy and G. Chechik, “Using explainabilty to detect adversarial attacks,” 2019 [Online]. Available: https://openreview.net/forum?id=B1xu6yStPH.

[29] G. Fidel, R. Bitton, and A. Shabtai, “When explainability meets adversarial learning: detecting adversarial examples using shap signatures,” in *Proceedings of 2020 International Joint Conference on Neural Networks (IJCNN)*, Glasgow, UK, 2020, pp. 1-8.

[30] H. Dai, J. Li, Y. Kuang, J. Liao, Q. Zhang, and Y. Kang, “Multiscale fuzzy entropy and PSO-SVM based fault diagnoses for airborne fuel pumps,” *Human-centric Computing and Information Sciences*, vol. 11, article no. 25, 2021. https://doi.org/10.22967/HCIS.2021.11.025