EURASIP Journal on Wireless Communications and Networking

RESEARCH **Open Access**

Securing 6G-enabled IoT/IoV networks by machine learning and data fusion

Bin Sun[1], Renkang Geng[1], Lu Zhang[1], Shuai Li[1], Tao Shen[1,2*] and Liyao Ma[1]

*Correspondence: cse_sunb@ujn.edu.cn; cse_st@ujn.edu.cn

[1] School of EE, University of Jinan, Jinan 250022, China
[2] HIT Robot Group Co. Ltd, Harbin 150060, China

Abstract

The rapid growth of Internet of Things (IoT) and Internet of Vehicles (IoV) are rapidly moving to the 6G networks, which leads to dramatically raised security issues. Using machine learning, including deep learning, to find out malicious network traffic is one of practical ways. Though much work has been done in this direction, we found little investigating the effect of using fused network conversation datasets to train and test models. Thus, this work proposes to check conversation dataset characteristics and find suitable ones to fuse into one dataset in order to improve the capability of malicious traffic and malware detection performance. The experiments using real data show that conditioned combination of datasets can be used to enhance algorithm performance and improve detection results. For this reason, it is recommended to profile datasets and conduct conditional fusion of network conversation datasets before using machine learning or deep learning. As the characterization is done using general statistical calculation, it is promising to be used for other domains too.

Keywords: Machine learning, Internet of Things (IoT), Internet of vehicles (IoV), 6G network

1 Introduction

The Internet of Things (IoT), especially the Internet of Vehicles (IoV), is rapidly growing and moving toward the 6G networks [26], which leads to dramatically raised security issues [39]. IoT and IoV are shaping the next generation of networking and communication [3]. IoT/IoV can be defined in different ways [49] and usually covers objects such as physical devices including sensors and vehicles [24]. Also, the meaning has expanded from local region to global region, covering different types of network such as local area network (LAN) and next-generation 6G mobile network. This is due to 5G technology cannot support the near-future IoT/IoV applications requirements [18]. When more things are included into the IoT/IoV network, network traffic among devices increases much more rapidly. On the one hand, IoT/IoV networking enriches daily life as well as industry equipment. On the other hand, malware is also getting multiplied and their threats are spreading across all network layers and periods of lifecycles [29, 30]. Security becomes one of the important research areas that is mainly due to two IoT/IoV characteristics, heterogeneity and dynamic [3]. The 6G network is supposed to support numerous heterogeneous devices and infrastructures and will exceed 5G networks [44].

One way to monitor and detect IoT/IoV cyber threats is conducting network traffic classification and prediction using data-based methods such as machine learning [5, 24]. A typical limitation of machine learning is the quality and quantity of data required. This limitation may be worse especially within specific scenarios on edge devices [6, 22]. Some work uses simulated data while some uses generative adversarial network (GAN) [11] to generate data and some add noise or fluctuation to data.

On each edge or local devices, the data are usually limited, especially abnormal data. Thus, it is a good idea to combine them if possible. Also, avoiding human in this combination procedure is preferred.

Though some studies have shown good data fusion results with similar or the same data scenarios [32, 45], the previous studies have not investigated if and when it is possible to join different datasets with different scenarios to enhance prediction and classification for malicious conversation detection. This work aims to investigate this procedure.

The sections of this article are organized as follows. Section 2 provides backgrounds related information regarding alternative approaches. Section 3 describes used methodologies in detail including methods and metrics, including description for data and experiments. Later, the experimental results are presented and analyzed in Sect. 4. Finally, conclusions are made, and limits are discussed in Sect. 5.

2 Background

Facing the challenges of IoT malware, researchers have done much work and proposed many solutions. One of the good practices is to use machine learning algorithms to distinguish malicious traffic from benign (normal) traffic and then take further actions, such as isolation or replacement of the infected things. When referring to machine learning, the definition can differ from people to people. We here consider the related definitions with the relation in Fig. 1. In most cases, traditional machine learning (excluding deep learning) algorithms are considered to be less power consuming and more comprehensive, while deep learning methods are used when more computing resources are available [37]. Artificial intelligence (AI) is often considered to be a broader concept and may refer to robots more than others. The biggest circle indicating data mining is dashed, as its definition is much more fuzzy and fogged. In this work, we use the definition from Kevin Murphy [27], and the field of machine learning is about developing algorithms that can automatically discover and describe patterns in history data [35].

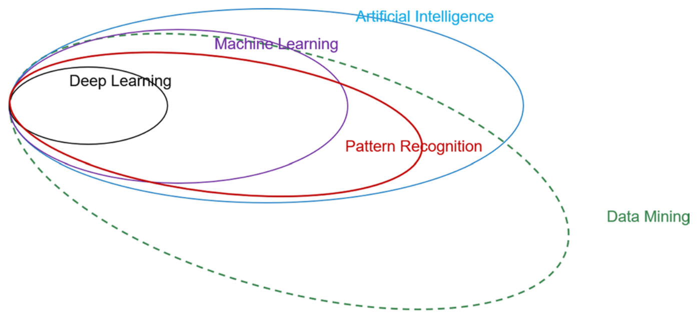

Fig. 1 Relationship of machine learning-related concepts in general, though definitions and opinions may differ

When analyzing temporal data using classical machine learning, researchers often adopt an ensemble learning approach. The idea is to combine weak learners by bagging, boosting and/or stacking considering usage scenarios to finally form a better learner [33].

XGBoost algorithm [7] is an ensemble algorithm proposed in recent years. It is based on decision tree structure CART (classification and regression tree) [19] and enhanced by bagging and boosting. It can also be considered as one kind of gradient boosted decision trees (GBDT) [17]. At present, XGBoost has been used by many researchers in data analysis and processing, especially for tabular data, showing relatively good prediction and classification effects. It is often used as it provides advantages such as easy model construction, strong universality and fast speed. Besides, it shows good performance comparable with or better than classical machine learning methods [25, 38, 42, 43] with better efficiency [7].

When analysing temporal data using deep learning algorithms, methods consisting memory mechanisms are often used, such as recurrent neural networks (RNN) [8] or long short-term memory networks (LSTM) [47] or even Generative Adversarial Network (GAN) [20]. Some researchers consider that for temporal network traffic data, relation among packets should be extracted, and conversations among devices are considered as better analysis targets instead of individual packets. This work take the second way to conduct analysis.

Meta learning has been used for combining different base weak learners to form a better stacked model [46, 51]. However, we consider to reduce the number of models and also consider when different edge devices gathering data locally and send to only similar edges to reduce network traffic as it costs. This is one of important costs to consider even for network operating companies.

Instance-based clustering [1] is not in the coverage of this work, with the similar reason that edge devices are usually not able to store too many history data.

3 Methods, features and metrics

In this work, we consider different datasets can be used together to train models as data from the same domain have relationships. To test this hypothesis, we consider to conduct experiments using a collection of real-world datasets within one domain.

3.1 Dataset characterization methods

The captured network traffic packets are firstly transformed to conversations among devices. Main features include originator (source) and responder (destination) addresses, ports and payload size in bytes and number of packets per conversation. General statistical calculation methods are employed to characterize datasets, including mean, standard deviation, quantiles at 0, 25, 50, 75, 100 percentages, skewness [9], kurtosis [12, 15] and median. Later, the first-order and second-order differentials of the above statistical characteristics are investigated to find insight between performance of algorithms and dataset fusion.

3.2 Machine learning methods

As mentioned previously, this work uses machine learning to indicate non-neural network methods. XGBoost algorithm is currently one of the widely used representative machine learning methods and is used as a baseline here.

Tree-based algorithms have high compatibility and can effectively classify and predict instances with a variety of different types of data.

Bagging trains several sub-models, and each model is trained using a generated bootstrapped dataset. Bootstrapping is a resampling method which samples instances (usually with replacement) from the original dataset. Datasets created by bootstrapping are only weakly dependent and randomness of samples matters. Thus, bagging builds sub-models in a parallel way. Boosting, however, builds sub-models in a serial manner with rounds. In boosting, a base sub-model is firstly build using selected weak learner. Later, the inference results are analyzed, and wrongly predicted instances are paid more attention in the next round using weighting strategies to reduce residuals. This continues until termination conditions are met. The XGBoost objective function consists of loss function part indicating model deviation and regularization part indicating model variance. Suppose the loss function is l and the regularization is Ω, the objective function for each round is:

$$L(\phi) = \sum_{i} l\left(y_i, \hat{y}_i\right) + \sum_{k} \Omega\left(f_k\right) \tag{1}$$

Those boosted models are then ensembled and used consecutively and additively. XGBoost objective functions can be approximated using Taylor series to unary quadratic polynomial functions shown as the formula below:

$$f(x_i) = w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n \tag{2}$$

where the variable $X = x_0, \ldots, x_n$ represents features in the training dataset. Usually, most features contribute little or even negatively and so they are given small weights or even discarded. Another reason to do so is to avoid over-fitting, i.e., avoid over-complex models, which only shows high accuracy during training while reduces generality. A good practice is trying to use as much data as possible during training to get models with high accuracy (low deviation) and high precision (low variance). The above ideas are combined together and lead XGBoost to a common good baseline.

3.3 Deep learning methods

Deep learning methods are developing rapidly though tabular-targeted deep architectures are limited. TabNet is a neural network algorithm for tabular data published by Google [4]. It implements instance-wise feature selection through a sequential attention mechanism similar to the additive model. Self-supervised learning is also realized through encoder-decoder framework. TabNet has some advantages over deep neural network (DNN) [23]. Similar to image processing, TabNet can encode the data of the table and obtain a representation of the original table data, which is a representation learning method. In this way, the model can learn the corresponding features from the original data without human intervention for feature extraction and corresponding processing. At the same time, the model can reduce its dependence on feature engineering [16] (i.e., feature cleaning, processing, selection, etc.). TabNet algorithm also has some advantages of tree algorithm [2] in table data prediction. For example, the decision manifold is good, the model can be learned sparsely, the model has high explanatory degree, and the training speed is fast. This makes it not only has the advantages of DNN, but also can be comparable with the current mainstream tree model in the prediction of table data. TabNet combines the idea of Sequential Attention with the behavioral logic of decision trees, and simulates the algorithmic logic of decision trees through the framework of multi-step neural networks.

A step of the algorithm corresponds to the establishment of a tree in the decision tree algorithm. The steps of the algorithm mainly include two key operations: 1.Attentive Transformer. The operation of this process is mainly to select some of the most important features for the next operation, that is, to simulate the feature selection step of the decision tree. The function of this layer is to calculate the Mask layer of the current step according to the output results of the previous step (the mask layer realizes feature screening through matrix operation).

The Sparsemax layer is a sparse version of Softmax. Fully connected (FC) layer is a fully connected layer where weight and bias are specially set, and it can make conditional judgment on the input feature vector. BN layer can reduce the size difference of different features in numerical value, and reduce the influence on the training process, and reduce the problem of gradient explosion and gradient disappearance. It is worth noting that the Mask vector of different samples can be different, that is to say, TabNet algorithm can make different samples select different features (instance-wise) to reduce the prediction error caused by too many samples. The traditional decision tree model uses the same features for each tree. 2. Feature Transformer. The main operation of this process is to process the input features into a more useful representation, that is, to simulate the decision tree and set the Feature threshold for Feature calculation.

The functions realized by BN (batch normalization) layer and GLU (gated linear units) layer at FC layer are similar to branching tree by setting threshold value in tree algorithm, and the output is the weight of influence feature in the final prediction result. Feature transformer is mainly composed of two parts. Related parameters in the layer of the first half are shared, that is, the parameter is jointly trained in all steps. The main reason for doing so is to extract the common part of feature calculation and determine which features have a high weight in the vast variety of books. In other words, each step is trained separately. In this way, its specific weight for features can be obtained for different samples, that is, the characteristic part of feature calculation can be obtained. Both parts are connected by residuals and multiplied by $\sqrt{0.5}$ to ensure the stability of the network.

Using both attentive and feature transformers, TabNet is able to simulate the decision-making process of a tree-based model. This model has the characteristics of both DNN and tree algorithm, and has been proved to have good prediction effect in image processing and table data processing.

3.4 Optimal cutoff

When considering the tradeoff between true positive versus false positive, or sensitivity versus specificity, an optimal cutoff value is needed. The optimal cutoff is calculated using Youden's J score statistic (Youden's index) [48], which is defined as:

$$J = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{3}$$

$$+ \frac{\text{true negatives}}{\text{true negatives} + \text{false positives}} - 1 \tag{4}$$

that is,

$$J = \text{sensitivity} + \text{specificity} - 1 \tag{5}$$

It is a reflection of to which extend the algorithm is good at distinguishing normal versus abnormal instances at balance considering both sensitivity and specificity simultaneously. The bigger Youden's index, the better algorithm is. From visual perspective, the selected cutoff value is actually in correspondence to the point on the curve that is most close to the top-left corner of the rectangular plotting area.

3.5 Results evaluation methods

Some metrics are used to evaluate algorithms and their results. For prediction and classification, commonly used effective metrics include accuracy, receiver operator characteristic (ROC) curve, the area under ROC curve (AUC). Accuracy is a direct and simple method to provide a quantitative comparison, roughly. Imbalanced data occur now and then [21], and accuracy is not a suitable metric in this situation. Unfortunately, network traffic dataset, in most cases, is imbalanced dataset with much more benign instances than malicious ones. Thus, it is necessary to have a well-recognized metric to evaluate results during network traffic analysis.

For classification, a good way to present results is to use confusion matrix [41] which includes key values that can be used for other purposes. A confusion matrix is an N*N matrix, the horizontal and vertical indexes are related categories, the diagonal lines are correctly classified data, and outside the diagonal lines are incorrectly classified data. For a dichotomous problem, we can divide the classification into the following four types: (1) Positive samples are correctly predicted to be positive samples (true positive, TP). (2) Positive samples are wrongly predicted to be negative samples (false negative, FN). (3) Negative samples were mispredicted as positive samples (false positive, FP). (4) Negative samples are correctly predicted to be negative samples (true negative, TN). These four cases correspond to the values in the confusion matrix. Receiver operating characteristic curve (ROC) [13] and AUC (area under the curve) value [28] are indicators of classification effect evaluation based on confusion matrix. It is often used to evaluate the merits of a binary classifier. At present, ROC curve and AUC value are popular and common evaluation indexes in classification problems [14]. The horizontal and vertical coordinates of the ROC curve are mainly determined by the following two formulas:

$$FPR = \frac{\text{false positive}}{\text{false positive} + \text{true negative}} \tag{6}$$

$$TPR = \frac{\text{true positive}}{\text{true positives} + \text{false negative}} \tag{7}$$

Abscissa (horizontal axis) indicates false positive rate (FPR), i.e., the proportion of samples that are predicted to be positive but actually negative in all negative samples. Ordinate (vertical axis) indicates true positive rate (TPR), i.e., the proportion of predicted positive samples and actually positive samples in all positive samples.

The specific drawing process of ROC curve is as follows: First, the classifier will give the probability of all samples being classified as positive samples. We took this probability as the score of the sample, sorted score from high to low, and classified positive and negative samples by score as threshold successively. (That is, each point constituting the ROC curve corresponds to a threshold.) Those higher than this value are positive samples, and those lower than this value are negative samples. For example, when threshold is the largest, $\text{TP} = \text{FP} = 0$, corresponding to the origin; when threshold is minimum, $\text{TN} = \text{FN} = 1$, corresponding to the point (1,1) in the upper right corner. The coordinates of a series of points (that is, the values of FPR and TPR) can be obtained by the above calculation formula. Finally, all nodes are connected to complete the ROC curve drawing.

The specific drawing process of ROC curve is as follows: First, the classifier will give the probability of all samples being classified as positive samples. So let us take this probability as an example and in the ideal case, the value of TPR should be close to 1, and the value of FPR should be close to 0. That is, the roc curve should be as close as possible to point (0,1) in the figure and deviate from the 45 degree diagonal.

The main advantage of ROC curve is that it can effectively deal with the classification problem under the condition of uneven distribution of positive and negative samples. When the number of positive and negative samples in a data set is too unequal, a phenomenon of class imbalance occurs. (In daily life, uneven sample distribution often occurs.) If we simply take the proportion of misclassified samples to all samples as the accuracy of the experiment, the effect is extremely poor. That is, the classifier can still obtain high accuracy when all samples are predicted to be positive or negative samples. However, the ROC curve can remain unchanged when the distribution of positive and negative samples changes.

AUC value is the area under the ROC curve, which can intuitively evaluate the quality of the classifier, ranging from 0 to 1. The larger the value, the better. Generally, the AUC value can be understood as a probability value, that is, the probability that positive samples rank before negative samples. The larger the AUC value is, the more likely the score of positive samples calculated by the current classification algorithm is to be higher than that of negative samples.

3.6 Data specification and experimental design

A labeled collection of datasets, Aposemat IoT-23 [10], which is available in public domain with malicious and benign IoT network traffic is used for testing our hypothesis. This collection is created by Avast AIC laboratory with the funding of Avast Software and was firstly published in 2020. It contains 23 datasets (corresponding to 23 scenarios) which consists of 20 datasets from a malware-conquered IoT network (botnet) and 3 benign traffic datasets. The datasets are numbered thus noted as Di in this work. The traffic was firstly captured in pcap format which is frequently used during network capturing [31, 40] and then processed by a analyzer to get flows, i.e., communication conversations.

Zeek is a passive and publicly available open-source network traffic data analyzer and supports performance measurement and troubleshooting [34]. It is often used for monitoring of network security status. It can be deployed to provide near real-time analysis to support investigations of suspicious or malicious activity. Through analyzing captured packets, Zeek finds communication conversations between devices and extract features.

Next, we make detailed statistics of the relevant features of these data, and introduce the feature engineering carried out in this experiment.

The extracted conversation features are conversation starting time, originator and responder IP address, originator and responder port number, transportation layer protocol (TCP or UDP), service (such as HTTP, DNS, DHCP, etc., decided mainly using protocol and port number), conversation duration, application layer payload total bytes sent by originator and responder, IP layer payload total bytes sent by originator and responder, connection termination status, total packets number sent by originator and responder. The malware datasets originate from pcap files sizes ranging from 2MB to 6GB (23 kilo to 271 million packets) and capturing duration from 1 h to 112 h. The number of flows in analyzed results ranges from 238 to about 74 million. The scenarios and malware also vary. It is worth mentioning that the above numbers regarding sizes, flows, packets, etc., are not positively correlated; thus, this IoT-23 dataset provides high diversity which is a good candidate to be investigated.

The conversation instances are labelled in Stratosphere laboratory with "label" and "type". Label indicates actual actions including normal traffic benign and abnormal traffic such as HeartBeat, FileDownload, PortScan, Attack, Torii (virus' names) among others. Type is a binary target with benign traffic as "benign," and all non-benign conversations are marked as "malicious". Type is the target that we use in this work to train and test.

We firstly analyze each dataset to get some initial overview including data cleanliness and relations among them. Data cleaning is important as many algorithms cannot work with missing values and may be misled by outliers [36]. To make dataset analysis applicable to other domains, robust and general statistical methods are used. For data cleaning part, datasets that contain missing values cannot be sent directly to TabNet so they are processed by k-nearest neighbor to impute missing values. All experiments are conducted using tenfold cross-validation (CV) to ensure the validity and robustness of results.

The proposed method can be represented in pseudocode as Algorithm 1.

Algorithm 1 Proposed Method Key Steps

```
 1: Read all captured datasets $D_H$
 2: for Subdataset $D_i$ do
 3:     Data cleaning, e.g. fill missing values;
 4:     Build features;
 5:     Give data labels;
 6: end for
 7: for Subdataset $D_i$ , $D_j$ do
 8:     if $i < j$ then
 9:         Fuse dataset from $D_i$ and $D_j$;
10:     end if
11:     Obtain the fused dataset $D_{ij}$;
12: end for
13: for Subdataset $D_i$ , $D_{ij}$ do
14:     Delimit training set and test set;
15:     Build the classification algorithm model;
16:     Train in the model;
17: end for
18: return Classification results & metrics
```

The experiments are conducted inside Jupyter Notebook on a machine with Windows 10, Intel Core i9-10900K CPU 3.7GHz, 64 gigabytes RAM, NVidia GeForce RTX 3080 with 10GB RAM Graphic Processing Unit. The python version 3.6 is used together with PyCharm version 2021.1.2 community edition build 211.7442.25, runtime version 1341.57 amd64, and the key packages are NumPy version 1.19.5, pandas version 1.1.5 and libxgboost version 1.4.2.

4 Results and analysis

In this section, experimental results are presented in a manner of algorithms and performance. Most of the datasets themselves are well organized and labelled which leads to the results that most classification and prediction results tend to be near perfect, thus are not discussed.

Typical results are presented using datasets D20 combined with D21 and D03 combined with D34. The experimental results from datasets D20 and D21 show that combining datasets may improve prediction and classification results. For XGBoost, comparing the results (AUC and residual) of the original D20 (Fig. 2) and D21 (Fig. 3), respectively, with the results of the combined data (Fig. 4), it can be seen that the AUC value of the combined data has significantly improved. The fusion increased AUC by 7.77% and 202% for D20 and D21, respectively. Detailed values are described in Table 1. It is worth mentioning that accuracy is showing its vulnerability and cannot “accurately” track the fusion influence.

Both D03 and D34 produced excellent results (AUC near 1.0) when using XGBoost alone (Figs. 5 and 6). Combining the D03 with D34 using XGBoost has produced some reductions (Fig. 7). After the combination, AUC drops suddenly by 7.39% and 6.56% for those datasets. Detailed values are described in Table 2

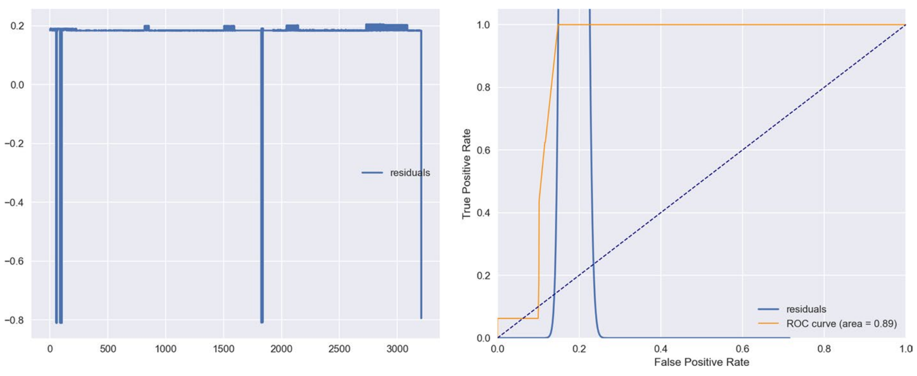

Fig. 2 Results of applying XGBoost to D20 only. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

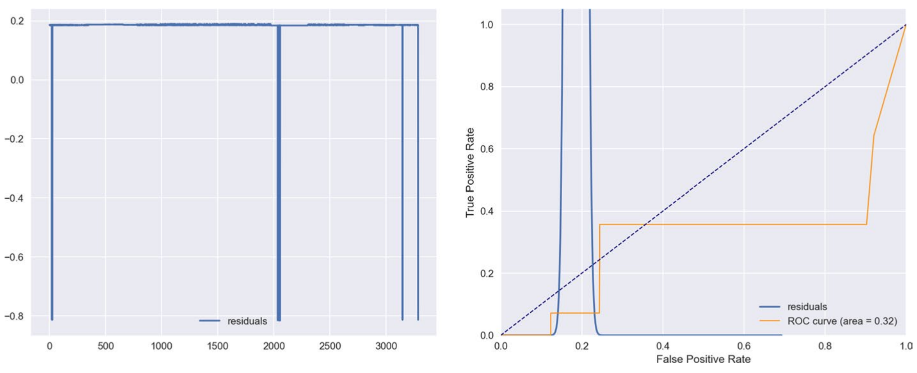

Fig. 3 Results of applying XGBoost to D21 only. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

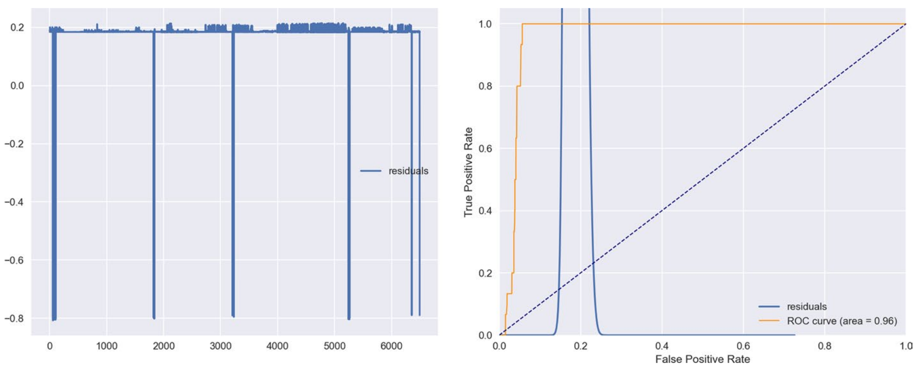

Fig. 4 Results of applying XGBoost to D20 + D21 fused data increase AUC dramatically. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

Table 1 XGBoost gives better results when combining two datasets appropriately

Dataset	AUC	Optimal Cutoff	Accuracy
D20	0.891628171	0.18944216	0.881271424
D21	0.318110374	0.18664216	0.753499696
D20+D21	0.96090745	0.1924175	0.943187067

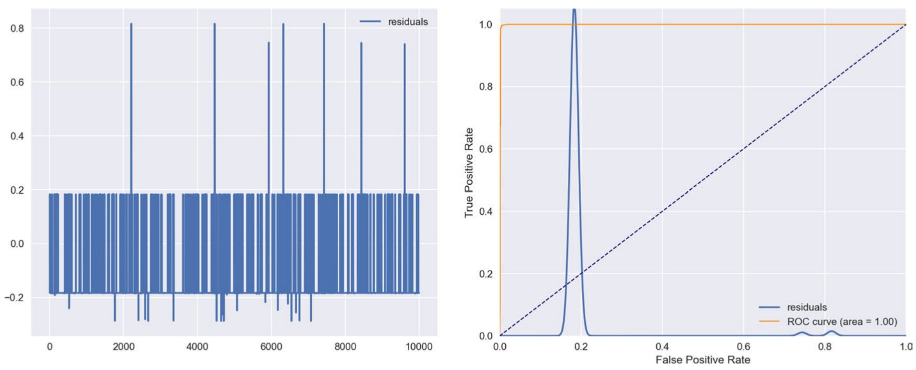

Fig. 5 Results of applying XGBoost to D03 only. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

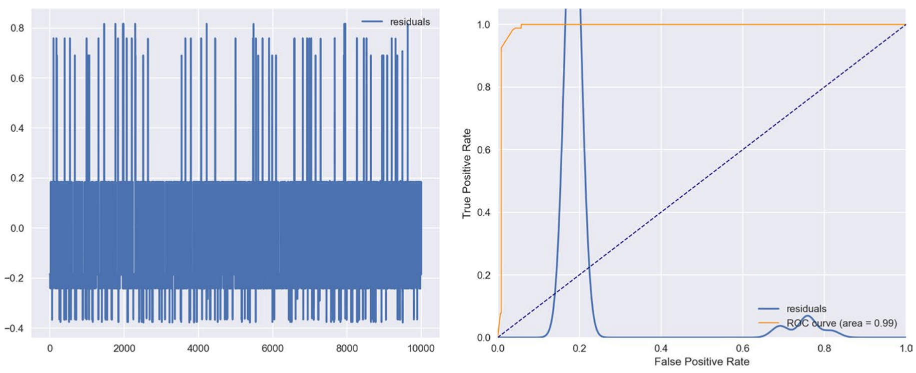

Fig. 6 Results of applying XGBoost to D34 only. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

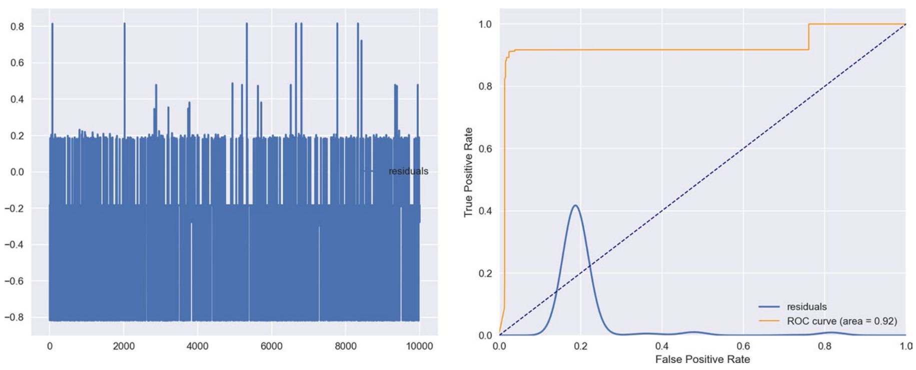

Fig. 7 Results of applying XGBoost to D03 + D34 fused data results worse AUC. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

Table 2 XGBoost gives worse results when combining two datasets appropriately

Dataset	AUC	Optimal Cutoff	Accuracy
D03	0.998767676	0.7754513	0.997559304
D34	0.989943557	0.7583015	0.980168503
D03+D34	0.924983789	0.78958625	0.913047844

TabNet produces similar result patterns and trends with XGBoost. Comparing the results of the original D20 and D21 (Figs. 8 and 9) with the results of the combined data (Fig. 10), the AUC has a higher improvement. This indicates an improvement of 32.1% and 27.7% for those two datasets individually. Detailed values are presented in Table 3.

The trend of the results obtained using TabNet on the D03 and d34 datasets is similar to that obtained using XGBoost. Detailed values are described in Table 4.

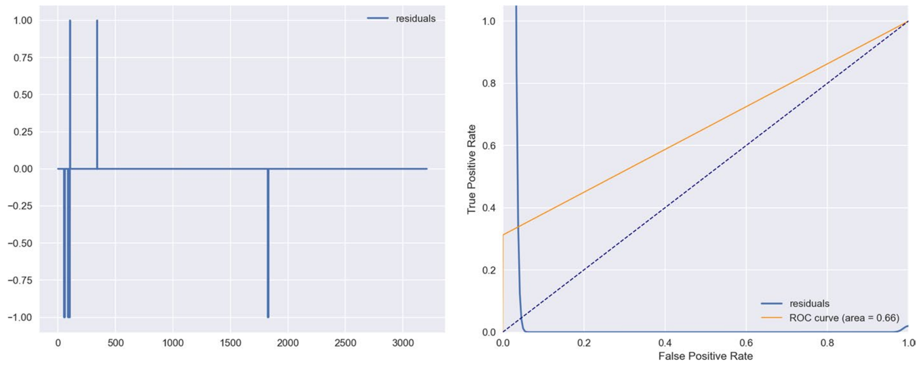

Fig. 8 Results of applying TabNet to D20 only. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

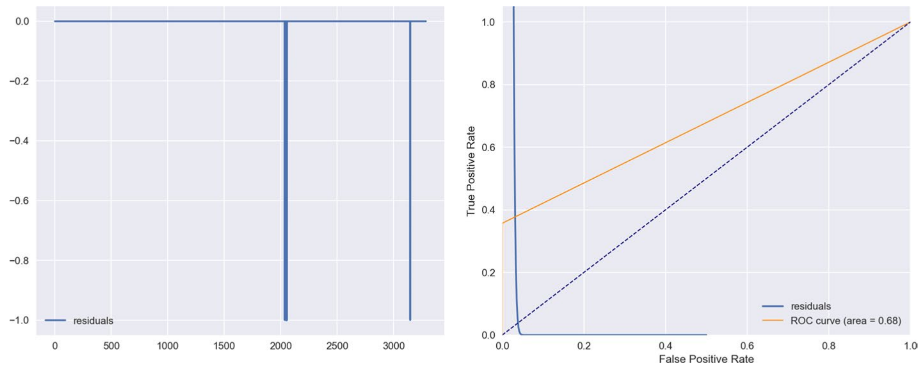

Fig. 9 Results of applying TabNet to D21 only. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

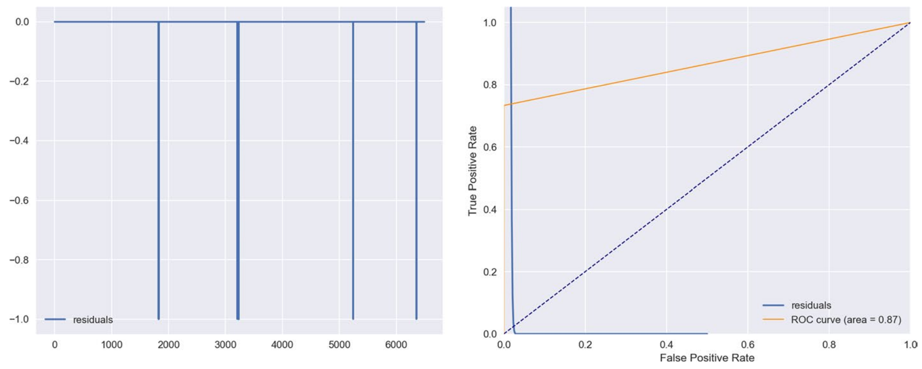

Fig. 10 Results of applying TabNet to D20 + D21. Left: residual sequence with original timed order. Right: ROC curve and residual distribution

Table 3 TabNet gives better results when combining two datasets appropriately

Dataset	AUC	Optimal Cutoff	Accuracy
D20	0.655936815	1	0.995014023
D21	0.678571429	1	0.995739501
D20+D21	0.866666667	1	0.995381062

Table 4 TabNet gives better results when combining two datasets appropriately

Dataset	AUC
D03	0.982570579
D34	0.969258236
D03+D34	0.922084062

Table 5 Performance of the single dataset

Performance	High	Low
Single Dataset	70%	30%

Table 6 Performance changes after data fusion

Performance	Increased	Unchanged	Decreased
Data Fusion	28%	64.5%	7.5%

It can be obtained from the data in Table 5. In the case of no data fusion, 70% of the data have good performance and can be effectively classified, and their AUC is above 0.98. However, the classification effect of 30% data is not good, and the average AUC value is about 0.5.

According to the data performance in Table 6, 28% of the data are significantly improved after fusion, which is especially in the case of the previous single datasets with low performance. 7.5% of the data performance decreased after fusion, while the AUC decreased by about 0.1. As for the rest part, since the AUC value is generally above 0.98, the classification effect of a single dataset is good enough, leading to little difference after data fusion.

From the above results, we can see that fusion datasets may increase algorithm performance given some limitations. Dataset characteristics show that some second-order feature differences between the two example fusions are huge. That is, the diversity of dataset characteristics between D03 and D34 is much higher than the ones between D20 and D21. Three characteristics are gaining more attention compared to others: standard deviation, skewness and kurtosis. For those three characteristics, the differences between D03 and D34 are 12 to 175 times higher the ones between D20 and D21. Thus, current experiments show that it is preferred to combine two datasets if there those three characteristics have less differences; otherwise, fusion may be a bad idea.

It can be found from the experiment that in D20 and D21 data sets, the distribution of positive and negative samples is obviously uneven, and the number of positive samples is very small, accounting for less than 5%. The results show that the algorithm is not sensitive to the classification of some anomalies in D20 and D21 datasets, but it is improved obviously after data fusion. The same is true in D42 and D44.

In D03 and D34, the proportion of positive samples is as high as 97% and 91%, respectively. (That is, there are many abnormal cases.) After data fusion, the performance is degraded to a certain extent. The same is true in other data sets with degraded performance.

Therefore, we judge that the lower the proportion of abnormal samples in the original samples, the more obvious the performance improvement after data fusion. That is, the algorithm becomes more sensitive to positive samples. However, the higher the proportion of abnormal samples in the original samples, the worse the performance of data fusion.

5 Conclusion

For the results and analysis, we can see that combing datasets with similar characteristics improves the algorithm performance.

The results show that combining datasets can improve AUC 8% to 202% using traditional machine learning (XGBoost here) and 28% to 32% using deep learning (TabNet here).

The improvement is produced given the condition that fused datasets have similar deviation, skewness and kurtosis. This is a novel finding that we have not found during literature review.

Though combining datasets may lead to good results, improper combination may give worse results than individual datasets. Combining D3 and D48 reduces AUC by about 7% when using XGBoost, and about 5% when using TabNet.

In summary, combination of datasets may help or worsen the performance. The second-order dataset characteristic differencing should be considered to decide fusion. For the data with similarity characteristics, it is a good way to consider combining datasets to improve the algorithm classification and prediction performance. However, if the datasets differ with each other much, the combination should not be used as it will decrease the performance.

This work is currently done within one collection of cybersecurity domain, datasets of other domains can be used to check the robustness of our methodology. Big–small-sized data fusion can be interesting given suitable situations especially when transfer learning is considered [50]. Also, TabNet seems to be able to provide better results with fine-tuned epoch rounds and patience. We plan to spend some time on those topics in exchanged for valuable performance enhancement.

Abbreviations

AI	Artificial Intelligence
AUC	Area Under Curve
BN	Batch Normalization
CART	Classification and Regression Tree
CV	Cross-Validation
DHCP	Dynamic Host Configuration Protocol

DNN	Deep Neural Network
DNS	Domain Name System
Dx	Dataset x
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FC	Fully Connected
GAN	Generative Adversarial Network
GBDT	Gradient Boosted Decision Trees
GLU	Gated Linear Units
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
LAN	Local Area Network
LSTM	Long Short-Term Memory Network
RNN	Recurrent Neural Network
ROC	Receiver Operator Characteristic
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
TPR	True Positive Rate
UDP	User Datagram Protocol

Acknowledgements
We would like to thank reviewers for their valuable suggestions.

Author contributions
BS and RG drafted the paper. LZ and SL conducted the experiments and analysis. TS reviewed the process and guided the direction. LM helped to revise. All authors read and approved the final manuscript.

Funding
This work is supported by National Key R &D Program of China 2018AAA0101703, Shandong Key Technology R &D Program 2019JZZY021005 and Natural Science Foundation of Shandong ZR2020MF067, ZR2020KF027.

Availability of data and materials
The data and material are published publicly.

Declarations

Competing interests
The author declare no competing interests with this research.

Received: 7 June 2022 Accepted: 26 October 2022
Published online: 22 November 2022

References
1. M. Aljabri, S.S. Aljameel, R.M.A. Mohammad, S.H. Almotiri, S. Mirza, F.M. Anis, M. Aboulnour, D.M. Alomari, D.H. Alhamed, H.S. Altamimi, Intelligent techniques for detecting network attacks: review and research directions. Sensors **21**(21), 7070 (2021)
2. R.H. Alsagheer, A.F. Alharan, A.S. Al-Haboobi, Popular decision tree algorithms of data mining techniques: a review. Int. J. Comput. Sci. Mob. Comput. **6**(6), 133–142 (2017)
3. A.H.M. Aman, E. Yadegaridehkordi, Z.S. Attarbashi, R. Hassan, Y.J. Park, A survey on trend and classification of internet of things reviews. IEEE Access **8**, 111763–111782 (2020)
4. S.O. Arık, T. Pfister, Tabnet: Attentive interpretable tabular learning, in *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*, vol. 35, pp. 6679–6687. Virtual (2021)
5. C. Chen, Q. Hui, W. Xie, S. Wan, Y. Zhou, Q. Pei, Convolutional neural networks for forecasting flood process in internet-of-things enabled smart city. Comput. Netw. **186**, 107744 (2021)
6. C. Chen, Y. Zhang, Z. Wang, Distributed computation offloading method based on deep reinforcement learning in ICV. Appl. Soft Comput. **103**, 107108 (2021)
7. T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in *22nd SIGKDD Conference on Knowledge Discovery and Data Mining*. San Francisco (2016)
8. D.K. Dennis, S. Gopinath, C. Gupta, A. Kumar, A. Kusupati, S. Patil, H. Simhadri, *EdgeML Machine LEARNING for Resource-Constrained Edge Devices* (Microsoft Research India, 2020)
9. D.P. Doane, L.E. Seward, Measuring skewness: a forgotten statistic? J. Stat. Educ. **19**(2) (2011)
10. S. Garcia, A. Parmisano, M. Erquiaga, IoT-23: A labeled dataset with malicious and benign IoT network traffic (v1.0.0) [Dataset]. Stratosphere Lab., Praha, Czech Republic, Technical Report (2020)
11. I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Networks. arXiv:1406.2661 [Cs, Stat] (2014)
12. R.A. Groeneveld, G. Meeden, Measuring skewness and kurtosis. J.R. Stat. Soc. Ser. D (Stat.) **33**(4), 391–399 (1984)

13. Z.H. Hoo, J. Candlish, D. Teare, What is an ROC curve? Emerg. Med. J. **34**(6), 357–359 (2017)
14. J. Huang, C.X. Ling, Using AUC and accuracy in evaluating learning algorithms. IEEE Trans. Knowl. Data Eng. **17**(3), 299–310 (2005)
15. D.N. Joanes, C.A. Gill, Comparing measures of sample skewness and kurtosis. J. R. Stat. Soc. Ser. D (Stat.) **47**(1), 183–189 (1998)
16. A. Jović, K. Brkić, N. Bogunović, A review of feature selection methods with applications, in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1200–1205. IEEE (2015)
17. G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.Y. Liu, LightGBM: a highly efficient gradient boosting decision tree, in *Advances in Neural Information Processing Systems*, vol. 30 (Curran Associates, Inc., 2017)
18. J.H. Kim, 6G and internet of things: a survey. J. Manag. Anal. **8**(2), 316–332 (2021)
19. R.J. Lewis, An introduction to classification and regression tree (CART) analysis, in *Annual Meeting of the Society for Academic Emergency Medicine*, vol. 14. Citeseer, San Francisco, California (2000)
20. W. Li, X. Zhong, H. Shao, B. Cai, X. Yang, Multi-mode data augmentation and fault diagnosis of rotating machinery using modified ACGAN designed with new framework. Adv. Eng. Inform. **52**, 101,552 (2022)
21. X. Li, J. Cheng, H. Shao, K. Liu, B. Cai, A fusion CWSMM-based framework for rotating machinery fault diagnosis under strong interference and imbalanced Case. IEEE Trans. Ind. Inform. **18**(8), 5180–5189 (2021)
22. S. Liu, J. Yu, X. Deng, S. Wan, FedCPF: an efficient-communication federated learning approach for vehicular edge computing in 6G communication networks. IEEE Trans. Intell. Transp. Syst. **23**(2), 1616–1629 (2022)
23. W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, F.E. Alsaadi, A survey of deep neural network architectures and their applications. Neurocomputing **234**, 11–26 (2017)
24. M.R. Mahmood, M.A. Matin, P. Sarigiannidis, S.K. Goudos, A comprehensive review on artificial intelligence/machine learning algorithms for empowering the future IoT toward 6G era. IEEE Access **10**, 87535–87562 (2022)
25. S. Mousa, P. Bakhit, O. Osman, S. Ishak, A comparative analysis of tree-based ensemble methods for detecting imminent lane change maneuvers in connected vehicle environments. Transp. Res. Rec. **42**, 268–279 (2018)
26. S. Mumtaz, M.I. Ashraf, V.G. Menon, T. Abbas, A. Al-Dulaimi, Guest editorial introduction to the special issue on intelligent autonomous transportation system With 6G. IEEE Trans. Intell. Transp. Syst. **23**(2), 1585–1586 (2022)
27. K.P. Murphy, *Machine Learning: A Probabilistic Perspective,* 1st edn. (The MIT Press, Cambridge, 2012)
28. J. Myerson, L. Green, M. Warusawitharana, Area under the curve as a measure of discounting. J. Exp. Anal. Behav. **76**(2), 235–243 (2001)
29. K. Peng, M. Li, H. Huang, C. Wang, S. Wan, K.K.R. Choo, Security challenges and opportunities for smart contracts in internet of things: a survey. IEEE Internet Things J. **8**(15), 12004–12020 (2021)
30. P. Radoglou-Grammatikis, K. Rompolos, P. Sarigiannidis, V. Argyriou, T. Lagkas, A. Sarigiannidis, S. Goudos, S. Wan, Modeling, detecting, and mitigating threats against industrial healthcare systems: a combined software defined networking and reinforcement learning approach. IEEE Trans. Ind. Inf. **18**(3), 2041–2052 (2022)
31. M.Z.N.L. Saavedra, W.E.S. Yu, Towards large scale packet capture and network flow analysis on hadoop, in *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, pp. 186–189. IEEE (2018)
32. H. Shao, J. Lin, L. Zhang, D. Galar, U. Kumar, A novel approach of multisensory fusion to collaborative fault diagnosis in maintenance. Inf. Fusion **74**, 65–76 (2021)
33. X. Shi, Y. Li, H. Li, R. Guan, L. Wang, Y. Liang, An integrated algorithm based on artificial bee colony and particle swarm optimization, in *2010 Sixth International Conference on Natural Computation*, vol. 5, pp. 2586–2590 (2010)
34. J. Siwek, Zeek (2022). https://docs.zeek.org/en/master/about.html
35. B. Sun, Automated Traffic Time Series Prediction. No. 10 in Blekinge Institute of Technology Doctoral Dissertation Series. Blekinge Tekniska Högskola, Karlskrona (2018)
36. B. Sun, W. Cheng, G. Bai, P. Goswami, Correcting and complementing freeway traffic accident data using mahalanobis distance based outlier detection. Teh. Vjesn. **24**(5), 1597–1607 (2017)
37. B. Sun, W. Cheng, P. Goswami, G. Bai, Short-term traffic forecasting using self-adjusting k-nearest neighbours. IET Intell. Transp. Syst. **12**(1), 41–48 (2018)
38. H. Sun, K. Zhang, T. Wang, W. Ma, Q. Zhao, Clustering-XGB Based Dynamic Time Series Prediction, in *2nd EAI International Conference on IoT and Big Data Technologies for HealthCare*. Leicester, Great Britain (2021)
39. F. Tang, Y. Kawamot, N. Kato, J. Liu, Future intelligent and secure vehicular network toward 6G: machine-learning approaches. Proc. IEEE **108**(2), 292–307 (2020)
40. V. Vesely, Extended comparison study on merging PCAP files. ElectroScope **2012**(5), 1–6 (2012)
41. S. Visa, B. Ramsay, A.L. Ralescu, E. Van Der Knaap, Confusion matrix-based feature selection. MAICS **710**, 120–127 (2011)
42. J. Vitorino, R. Andrade, I. Praça, O. Sousa, E. Maia, A Comparative Analysis of Machine Learning Techniques for IoT Intrusion Detection. Tech. Rep. arXiv:2111.13149, arXiv (2021)
43. D. Wang, Q. Zhang, S. Wu, X. Li, R. Wang, Traffic flow forecast with urban transport network, in *2016 IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, pp. 139–143 (2016)
44. J. Wang, X. Ling, Y. Le, Y. Huang, X. You, Blockchain-enabled wireless communications: a new paradigm towards 6G. Natl. Sci. Rev. **8**(9), nwab069 (2021)
45. Y. Xu, J. Cao, Y.S. Shmaliy, Y. Zhuang, Distributed Kalman filter for UWB/INS integrated pedestrian localization under colored measurement noise. Satell. Nav. **2**(1), 1–10 (2021)
46. A. Yang, C. Lu, J. Li, X. Huang, T. Ji, X. Li, Y. Sheng, Application of meta-learning in cyberspace security: a survey. Digit. Commun. Netw. (2022). https://doi.org/10.1016/j.dcan.2022.03.007
47. J.M. Yang, Z.R. Peng, L. Lin, Real-time spatiotemporal prediction and imputation of traffic status based on LSTM and Graph Laplacian regularized matrix factorization. Transp. Res. Part C Emerg. Technol. **129**, 103228 (2021)
48. W.J. Youden, Index for rating diagnostic tests. Cancer **3**(1), 32–35 (1950)
49. H.Y. Youm, An overview of security and privacy issues for internet of things. IEICE Trans. Inf. Syst. **E100.D**(8), 1649–1662 (2017)

50. H. Zhiyi, S. Haidong, J. Lin, C. Junsheng, Y. Yu, Transfer fault diagnosis of bearing installed in different machines using enhanced deep auto-encoder. Measurement **152**, 107393 (2020)
51. T. Zoppi, M. Gharib, M. Atif, A. Bondavalli, Meta-learning to improve unsupervised intrusion detection in cyber-physical systems. ACM Trans. Cyber-Phys. Syst. **5**(4), 42:1-42:27 (2021)

Publisher's Note