


RESEARCH

Open Access



Ensemble learning based anomaly detection for IoT cybersecurity via Bayesian hyperparameters sensitivity analysis

Tin Lai^{1*} , Farnaz Farid², Abubakar Bello² and Fariza Sabrina³

Abstract

The Internet of Things (IoT) integrates more than billions of intelligent devices over the globe with the capability of communicating with other connected devices with little to no human intervention. IoT enables data aggregation and analysis on a large scale to improve life quality in many domains. In particular, data collected by IoT contain a tremendous amount of information for anomaly detection. The heterogeneous nature of IoT is both a challenge and an opportunity for cybersecurity. Traditional approaches in cybersecurity monitoring often require different kinds of data pre-processing and handling for various data types, which might be problematic for datasets that contain heterogeneous features. However, heterogeneous types of network devices can often capture a more diverse set of signals than a single type of device readings, which is particularly useful for anomaly detection. In this paper, we present a comprehensive study on using ensemble machine learning methods for enhancing IoT cybersecurity via anomaly detection. Rather than using one single machine learning model, ensemble learning combines the predictive power from multiple models, enhancing their predictive accuracy in heterogeneous datasets rather than using one single machine learning model. We propose a unified framework with ensemble learning that utilises Bayesian hyperparameter optimisation to adapt to a network environment that contains multiple IoT sensor readings. Experimentally, we illustrate their high predictive power when compared to traditional methods.

Keywords IoT cybersecurity, Anomalies detection, Ensemble machine learning, Bayesian optimisation, IoT security

Introduction

We live in a modern era where internet-connected devices are ubiquitous, and cybersecurity threats persist everywhere. The connected devices, often known as the Internet of Things (IoT) (Lee and Lee 2015), refer to all electronic devices connected to the internet or other devices. IoT devices are capable of transmitting

and collecting data for various tasks. For example, nowadays, household appliances, cars, tools, or personal devices can sense, process, and connect to the internet (González-Zamar et al. 2020). The development of technologies enables the usage of IoT in urban environments, creating smarter cities (Okano 2017). Devices can now communicate with each other, such as household appliances, consumer devices, sensors, or even industrial controls (Okano 2017). IoT products resemble a powerful approach for increasing connectivity between devices. Altogether it represents interconnected devices and services that can communicate and share data and information among various domains and applications. IoT lead systems can transform our lives by making intelligent decisions to improve the quality of daily tasks. For example, seamless home integration, smart

*Correspondence:

Tin Lai
tin.lai@sydney.edu.au

¹ School of Computer Science, The University of Sydney, Sydney, NSW, Australia

² School of Social Sciences, Western Sydney University, Sydney, NSW, Australia

³ School of Engineering and Technology, Central Queensland University, Emerald, QLD, Australia



© The Author(s) 2024. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

city infrastructures, and transportation are relevant sectors in which IoT devices can be used. The high availability of data generated by IoT devices enables researchers to perform extensive scientific analysis on data mining and extracting the underlying relationship that is otherwise hard to discover (Kaur et al. 2018). However, like many other new innovations, it comes with specific security risks (Wheelus and Zhu 2020). By extending internet connections to everyday devices, these threats have expanded from our homes to workplaces, healthcare and other facilities. Many security risks exist that enable various electronic devices to exhibit the capability to record and send received signals to remote locations over the internet (Hassija et al. 2019). However, this unified capability also opens a powerful opportunity in the application of anomaly detection.

Cybersecurity threat modelling and detection via anomaly detection is a multidisciplinary problem with applications in various domains. It involves identifying unusual or unexpected observations within the captured data due to uncommon values or data sequence (Chandola et al. 2009). Anomaly detection finds patterns in data that do not conform to expected behaviour, and these non-conforming patterns are referred to as *anomalies*. Traditional anomaly detection applications are typically used extensively in intrusion detection, spotting malicious activities and even in safety-critical systems like military surveillance. The usage is applicable in numerous domains, and anomalies in data often translate to critical actionable information. Anomalies might be induced due to various reasons like malicious activity, credit card fraud, cyber-intrusion, breakdown of a system, or malfunction of some components. Therefore, the ability to detect such events provides us with an opportunity to react to such an anomaly event.

Moreover, anomalies in data often imply critical and actionable information essential to implement a secure network system. Anomaly detection had been successfully used to identify thief (Aleskerov et al. 1997), presence of tumours in medical settings (Spence et al. 2001), compromised computer sending sensitive data to remote location (Kumar 2005), or even signifying faulty component within space craft (Fujimaki et al. 2005). Anomaly detection enhances cybersecurity's resilience and robustness in mission-critical applications.

The abundance of sensor data enables a new opportunity for vendors and device owners to utilise the sensor data for continuous system monitoring. Using a machine-intelligent approach to tackle cybersecurity via automatic detection of anomaly events can better utilise the available information. The use of machine learning (ML) in IoT devices is still relatively

new compared to other domains (Cook et al. 2020). Traditional methods in for anomaly detection often involves rule-based approaches, which require manual adjustments to accommodate changing conditions. In contrast, ML have a remarkable ability to adapt and generalise on patterns that are directly learned from historical data, making them suitable for environments where the characteristics of normal and anomalous behaviour may evolve over time. Moreover, ML models can be fine-tuned to reduce false positive rates by learning from data and making informed decisions based on observed patterns. Traditional methods may produce more false positives since they often rely on static thresholds that may not account for contextual nuances.

In contrast to typical datasets in traditional ML, most data sources collected in IoT devices are time-series because of the continuous monitoring nature of the sensory devices. Most *contextual anomalies* are observations that deviate from the expected patterns within the time-sires (Chandola et al. 2009). The recent advancement of Deep Learning (DL) also enables us to utilise them as a universal function approximator to extract patterns that are had to hand-craft automatically. For example, DL can detect suspicious activities in a practical real-time system for intrusion detection (Alghamdi et al. 2021).

In this study, we improve the robustness of the existing state-of-the-art ensemble machine learning model in cybersecurity via a systematic hyperparameter optimisation process. We comprehensively investigate the predictive performance of traditional machine learning and ensembles of ML models in a wide range of datasets. In particular, machine learning models tend to be sensitive to the choice of hyperparameters, which can significantly affect the model's predictive accuracy. Our contributions are summarised as follows.

- (i) We present an in-depth empirical study on ensemble models' predictive performance for IoT Cybersecurity, which combines multiple weak predictors into ensembles of models with a much higher predictive capability.
- (ii) We investigate the choice of hyperparameter via parametric sensitivity analysis and present the important set of hyperparameters for each type of ensemble model.
- (iii) We propose a Bayesian-based framework for training ensemble models that utilise Bayesian Optimisation techniques for automatic search for the best set of hyperparameters via optimising a surrogate model.

- (iv) Finally, we summarise the essential network features for detecting network cybersecurity threats on IoT devices.

Experimentally, we demonstrate the Bayesian-based framework's effectiveness, which can improve the F1 score of state-of-the-art ensemble models by 10% to 30% for models with and without hyperparameter optimisation.

Related works

IoT systems have benefited the development and data usage across numerous domains. However, several weaknesses exist across IoT systems, for example, vulnerability, security, device disruption, data theft, interruption, and Man-in-The-Middle attacks (Gou et al. 2013). In addition to the research and development of communication techniques, a large amount of work within IoT involves imposing security measures (Hassan 2019) and bridging with the domain of cyber computing. For example, it is essential to strategise a security system (Hassija et al. 2019) for minimising the potential vulnerability and composing network architectures that address security threats (Hwang 2015). Most recent researches focus on challenges and status within IoT security measure (Mahmoud et al. 2015), IoT exploitations (Neshenko et al. 2019), possible approaches to enhance security (Riahi et al. 2013) and possible resolution by employing a deep learning approach (Al-Garadi et al. 2020). Innovative smart city applications connect enormous IoT devices to real-world objects spanning large distances and can often provide essential benefits to urban life (Borgia 2014). Moreover, the massive number of IoT devices contain heterogeneous services and protocols, which leads to the complexity of managing numerous devices across networks. Therefore, integrating devices without proper management might introduce serious cybersecurity threats and vulnerabilities for malicious actors to attack and extract daily activities and information about the average citizen's life.

Machine Learning (ML) is a powerful approach to enhancing the detection of IoT cyberattacks and malicious events. It is an integral part of extracting patterns within data (Mahesh 2020) for providing a data-driven approach to creating predictive models (Carbonell et al. 1983). There have been lots of recent development in artificial intelligence in various domains; for example, in computer vision (Gerald et al. 2019) and robotics (Lai and Ramos 2021) domains, for problems such as classification (Kotsiantis et al. 2007), extracting patterns from time series data (Wang et al. 2021), and for processing sequential data (Dietterich 2022). A smart city calls for the use of more innovative systems, Big Data analytics and the use of pattern recognition

for detecting trends that are abnormal from previous observations. For example, we can associate the current observation of some sensor data against previously observed data to assess the degree of deviations from the expected value defined by the anomaly detection model. There are a variety of methods of generating anomaly scores that are unique to each detection algorithm and model. Anomaly scoring is helpful in the identification and management of outliers when performing analytical tasks such as predictive analytics.

Machine Learning allows for utilising sensor data from IoT devices to improve their associated security (Xiao et al. 2018). Using pattern matching and computing statistical inference can help to address some of the ongoing IoT security challenges (Zhang et al. 2014). The collected IoT sensor data typically contain temporal components as they are some time series describing monitoring system attributes. Anomaly detection in time series is a well-known problem that has been tackled with a long history (Hawkins 1980; Abraham and Chuang 1989). We refer interested readers to existing surveys like (Markou and Singh 2003) and (Zhang et al. 2010) for a comprehensive overview of the various statistical approaches and network models employed in time-series data.

More recently, deep learning approaches have also been employed in the space of time-series anomaly detection, with some focus on applications in Industrial IoT. Most current approaches involve some statistical inference for utilising historical data for modelling the expected system behaviour. It involves pattern matching of anomaly events in a supervised-learning setting with known anomalous characteristics or distance-based approaches that detect anomalies by detecting outliers. Deep learning cybersecurity approach can be adopted in applications like smart grids (Ruan et al. 2023) and DDoS intrusion detection (Akgun et al. 2022). Most existing techniques can be grouped into the following categories:

- (i) Statistical and Probabilistic approach—utilise historical data to model the expected behaviour of a system. New observations are compared against the current model for the system of interest. When the observation does not agree with the model, the observation is registered as an anomaly.
- (ii) Predictive model—where a regression model is used to predict potential future values, and when the predicted value does not agree with the observed values, they are flagged as an anomaly.
- (iii) Clustering-based methods—project the data into a high-dimensional space and uses the density of the resulting clusters to determine outliers. Significant clusters are typically indicated as regular observations, while outliers are identified as anomalies.

- (iv) Pattern matching—directly models the time series in a supervised setting with known characteristics for anomalous data.

New observations are compared against a database of labelled anomaly events and flags observations substantially different from data within the database as an anomaly. Recent advancements in big data have enabled supervised learning methods in IoT settings due to the availability of sufficiently large IoT datasets. The IOT-23 (Garcia et al. 2020) dataset is a recent network traffic dataset that consists of recorded data from multiple intelligent home IoT devices. The recording devices include *Amazon Echo*, *Phillips HUE*, and *Somfy Door Lock*. The dataset contains real and labelled IoT malware infections and benign traffic. The mass adoption of IoT devices enables researchers to access numerous datasets. For example, MQTTset (Vaccari et al. 2020) is another dataset that recorded network traffics under MQTT protocol. The availability of IoT deployment has increased the integration pace and extended the internet to multiple physical devices in our physical world. This paper comprehensively studies the current state-of-the-art ensemble models on IoT anomaly detection tasks on multiple cybersecurity datasets.

Methodology

We propose a unified framework for evaluating Machine Learning models for use in an internet-connected environment via a monitoring network for detecting cybersecurity related anomaly events. The proposed methodology implements a detector for anomalies. Systematically, we analyse the overall performance of various machine learning models and their respective model architectures.

To identify IoT machine learning models that perform well across a wide range of domains, we empirically evaluate the performance of various baseline models by optimising their hyperparameters and their sensitivity toward the choice of parameters. Our framework uses a Bayesian approach in searching for the most optimal set of hyperparameters. Bayesian optimisation is a sequential design strategy that aims to optimise some objective value, which corresponds to the anomaly detection performance in our scenario. Then, we present a set of ensemble models capable of combining the strength of multiple weaker predictive models into one that achieves superior predictive performance. This section describes the systematic procedure of data preprocessing, data cleaning, feature engineering, and the various models used in this study. Then, in section “Experimental results”, we present the empirical results obtained from our benchmark models, followed by a discussion of model sensitivity

and important parameters in section “Hyperparameters optimisation”.

Data preprocessing

We first standardise each input feature in our evaluation by removing the mean and scaling to unit variance from the dataset \mathcal{X} . Let $\mathbf{x}_i \in \mathcal{X}$ denote the i^{th} datapoint, where we will use $\mathbf{x}_{i,j}$ to index into the j^{th} feature of the datapoint. We perform the standardisation by transforming each feature $\mathbf{x}_{i,j}$ to $\mathbf{x}'_{i,j}$ by

$$\mathbf{x}'_{i,j} = \frac{\mathbf{x}_{i,j} - \bar{\mathbf{x}}_j}{\hat{\sigma}_j} \quad (1)$$

where

$$\bar{\mathbf{x}}_j = \frac{1}{|\mathcal{X}|} \sum_{i=1}^{|\mathcal{X}|} \mathbf{x}_{i,j}, \quad (2)$$

$$\hat{\sigma}_j = \sqrt{\frac{1}{|\mathcal{X}|} \sum_{i=1}^{|\mathcal{X}|} (\mathbf{x}_{i,j} - \bar{\mathbf{x}}_j)^2}, \quad (3)$$

and $|\mathcal{X}|$ denote the cardinality of \mathcal{X} . In addition, we compute pairwise Pearson correlation coefficient for the input features, where the correlation coefficient $r_{j,k}$ for the j^{th} and k^{th} feature is given by

$$r_{j,k} = \frac{\sum_{i=1}^{|\mathcal{X}|} (\mathbf{x}_{i,j} - \bar{\mathbf{x}}_j)(\mathbf{x}_{i,k} - \bar{\mathbf{x}}_k)}{\sqrt{\sum_{i=1}^{|\mathcal{X}|} (\mathbf{x}_{i,j} - \bar{\mathbf{x}}_j)^2 \sum_{i=1}^{|\mathcal{X}|} (\mathbf{x}_{i,k} - \bar{\mathbf{x}}_k)^2}} \quad (4)$$

which gives us the measure of linear correlation between the two features. If $r_{j,k} > \delta$ for $\delta \in \mathbb{R}, 0 < \delta \leq 1$, then we will remove the k^{th} feature to help avoid overfitting. The δ acts as a threshold to avoid highly correlated features, and in our experiments, we set $\delta = 0.7$. We also convert all categorical features into one hot encoded feature, except for the IP address features. It contains more than 5000 unique sparse categories in some scenarios, dramatically increasing the feature set size without providing meaningful predictive power.

Anomaly detection dataset

We evaluate each benchmark model against the following set of cybersecurity datasets to develop a model architecture that performs reasonably well across a range of IoT datastream domains. In the following, we will briefly describe each dataset's content and the type of features the dataset contains.

The IOTID20 (Kim 2019) is an IoT intrusion dataset designed to be a comprehensive network dataset with flow-based features. These flow-based features in this new botnet dataset help analyse flow-based intrusion

detection systems, especially for monitoring anomalous activity across IoT networks. The dataset contains captured attack packets from the smart home devices *NUGU (NU 100)* and *EZVIZ Wi-Fi Camera (C2C Mini O Plus 1080P)*, alongside some other laptops and smartphones within the same wireless network. In particular, *IoTID20* contains 80 network features and detailed categories. The dataset contains three variants for the labelled classes:

- 1 *IoTID20 Binary* contains two possible classes. (i) *Normal*: which indicates that there are no suspicious or malicious activities found within the connections, and (ii) *Malicious*: which indicates malicious network traffics from infected devices.
- 2 *IoTID20 Multi-Cat* contains multiple categories with five possible classes. (i) *Normal*: represents the same set of flow data as the *Binary* dataset. The *Malicious* is divided into individual classes, with (ii) *DoS*: denoting network traffic that belongs to a Denial-of-Service attack; (iii) *MITM ARP Spoofing*: denoting Man in the Middle attack by ARP spoofing; (iv) *Mirai*: denoting traffics coming from devices that are infected by the Mirai malware, which will turn networked devices into remotely controlled bots; and (v) *Scan*: which denote traffics scanning the IoT devices.
- 3 *IoTID20 Multi-SubCat* divides the malicious traffic into sub-categories with nine possible classes. (i) *Normal*: are the same as the previous variants; (ii) *DOS-Synflooding*: denote Denial-of-Service attack by SYN flood; (iii) *MITM ARP Spoofing*: denoting Man in the Middle attack by ARP spoofing; (iv) *Mirai-Ack-flooding*: denote traffics from Mirai Bot that is performing flooding with TCP ACK packets; (v) *Mirai-HTTP flooding*: denote traffics from Mirai Bot that is performing flooding with HTTP requests; (vi) *Mirai-Host Bruteforcing*: denote traffics from Mirai Bot that is performing brute force attack on a virtual host; (vii) *Mirai-UDP flooding*: denote traffics from Mirai Bot that is performing flooding with User Datagram Protocol (UDP) packets; (viii) *Scan Hostport*: denote traffics that is scanning the host for open ports; and (ix) *Scan Port OS*: denotes traffic scanning the ports of the OS.

The *IoT-23* (Garcia et al. 2020) dataset is a recent network traffic dataset that consists of recorded data from multiple smart home IoT devices. The recording devices include *Amazon Echo*, *Phillips HUE*, and *Somfy Door Lock*. The dataset contains real and labelled IoT malware infections and benign traffic. In particular, *IoT-23* contains twenty-three captured scenarios, including twenty

malicious network traffic and three benign traffic captures. Moreover, the dataset recorded newer devices that the current cyber security systems have not interacted with before, which helps evaluate the current security measure against newer IoT devices. In our evaluation of the candidate models, we clean the dataset by unifying the mismatched labels, loading up the first 1, 000, 000 entries of each captured scenario, and removing classes containing less than five instances. The dataset contains two variants for the labelled classes:

- 4 *IoT-23 Binary* contains two possible classes. (i) *Benign*: which indicates that there are no suspicious or malicious activities found within the connections, and (ii) *Malicious*: which indicates malicious network traffics from infected devices.
- 5 *IoT-23 Multi-Cat* contains two possible classes. (i) *Benign*: which indicates that there are no suspicious or malicious activities found within the connections, and (ii) *Malicious*: which indicates malicious network traffics from infected devices.

Table 1 summarises the content of all the evaluating datasets. The dataset content and associated labels are available to download in the linked address for reproducing our empirical study. In the following, we will detail the models used in this study and our methodology for our Bayesian approach to optimising the hyperparameters.

Bayesian optimisation with tree-structured parzen estimator

This study uses tree-Structured Parzen Estimator (TPE) to implement our automatic model training framework. TPE is a sequential model-based Bayesian Optimisation (BO) approach, where it sequentially constructs models to approximate the performance of hyperparameters based on historical measurements. Then, the algorithm updates its internal model and selects a new candidate of hyperparameters with a high potential for better performance.

Hyperparameter optimisation is formally defined as follows. The performance of hyperparameters x of some model f can be measured by the mapping $f : \mathcal{X} \rightarrow \mathbb{R}$, where \mathcal{X} denote the space of all possible hyperparameters and \mathbb{R} is the real domain. The functional $f(x)$ is the optimisation objective, and BO aims to minimise the objective score. We use x^* to denote the hyperparameters that can yield the lowest possible objective score, which is given by

$$x^* = \arg \min_{x \in \mathcal{X}} f(x). \quad (5)$$

Let $y = f(x)$ denote our performance measure in this study, where y is typically *F1 score* or *accuracy* in our IoT

Table 1 A detailed overview of the distribution of classes within the evaluated dataset

| Dataset | | Label | Train set | Test set | Label instances |
|---------|--------------|-------------------------|------------|-----------|-----------------|
| IoTID20 | Binary | Normal | 31,979 | 8,094 | 40,073 |
| | | Anomaly | 468,353 | 116,989 | 585,342 |
| | | Total | 500,332 | 125,083 | 625,415 |
| IoTID20 | Multi-Cat | Normal | 31,979 | 8,094 | 40,073 |
| | | DoS | 47,537 | 11,854 | 59,391 |
| | | MITM ARP Spoofing | 28,214 | 7,163 | 35,377 |
| | | Mirai | 332,546 | 82,763 | 415,309 |
| | | Scan | 60,056 | 15,209 | 75,265 |
| | | Total | 500,332 | 125,083 | 625,415 |
| IoTID20 | Multi-SubCat | Normal | 31,979 | 8,094 | 40,073 |
| | | DoS-Synflooding | 47,537 | 11,854 | 59,391 |
| | | MITM ARP Spoofing | 28,214 | 7,163 | 35,377 |
| | | Mirai-Ackflooding | 44,117 | 11,007 | 55,124 |
| | | Mirai-HTTP Flooding | 44,643 | 11,175 | 55,818 |
| | | Mirai-Host Bruteforcing | 97,093 | 24,085 | 121,178 |
| | | Mirai-UDP flooding | 146,693 | 36,496 | 183,189 |
| | | Scan Hostport | 17,756 | 4,436 | 22,192 |
| | | Scan Port OS | 42,300 | 10,773 | 53,073 |
| | | Total | 500,332 | 125,083 | 625,415 |
| IoT-23 | Binary | Benign | 1,462,947 | 366,180 | 1,829,127 |
| | | Malicious | 9,099,261 | 2,274,373 | 11,373,634 |
| | | Total | 10,562,208 | 2,640,553 | 13,202,761 |
| IoT-23 | Multi-Cat | Benign | 1,462,947 | 366,180 | 1,829,127 |
| | | PartOfPortScan | 5,966,736 | 1,492,481 | 7,459,217 |
| | | Okiru | 2,102,304 | 523,948 | 2,626,252 |
| | | DDoS | 1,010,098 | 252,925 | 1,263,023 |
| | | Attack | 5,518 | 1,425 | 6,943 |
| | | C &C | 12,501 | 3,026 | 15,527 |
| | | C &C-HeartBeat | 2,022 | 541 | 2,563 |
| | | C &C-FileDownload | 63 | 16 | 79 |
| | | C &C-Torii | 19 | 11 | 30 |
| Total | 10,562,208 | 2,640,553 | 13,202,761 | | |

cybersecurity detection domain. We can then reformulate the BO optimisation objective to minimise the negative F1 score as our performance measure.

In a typical BO setting, we aim to find the conditional probability of the objective score given hyperparameters, i.e., $\mathbb{P}(y | x)$. In TPE, we instead model $\mathbb{P}(x | y)$ and $\mathbb{P}(y)$ by transforming the generating process of hyperparameters and replacing the distribution of the hyperparameters prior with non-parametric densities. TPE begins by collecting a few observations using a randomly selected set of hyperparameters. Then, TPE sorts the collected observations by their respective objective score and divide them into groups based on quantile. The quantiles are used to model empirical densities using Parzen Estimators. A new sample of hyperparameters is then drawn

from the densities, returning hyperparameters that yield the greatest expected improvements. Therefore, TPE is an iterative process that uses the history of evaluated hyperparameters to create a probabilistic model, which is used to suggest the next set of hyperparameters to evaluate.

Benchmarking models

This section presents the experimental details of each dataset's machine-learning models. We evaluated 14 types of machine learning models, including six traditional ML models and eight ensemble models. Each model has its respective set of hyperparameters. We perform hyperparameters optimisation via a hierarchical Gaussian Process and a tree-structured Parzen

estimator (Bergstra et al. 2011) with a budget of 45 trials to estimate the best performing set of parameters (see section “Hyperparameters optimisation” for visualisation of the influences of hyperparameters).

Our benchmarking models are as follows.

- 1 *Ridge Regressor (Ridge)* (Hoerl and Kennard 1970): a model that addresses the classification problems as ordinary least squares by imposing a penalty on the size of the coefficients. Ridge treats the multi-class classification setup as a regression task with multiple outputs. It is a linear model that trains fast large dataset but cannot predicts well for nonlinear data.
- 2 *Naive Bayes (NB)* (Flach and Lachiche 2004): is a model that uses Bayes theory for performing class conditional density estimation and with classes prior probability. The posterior class probability of input test data is derived using the Bayes theory to be assigned to the class with the maximum posterior class probability. NB can often learn quickly from a large dataset when compared to other classifiers; however, the conditional independence assumption in NB is rarely applicable to real-world problems.
- 3 *Multi-layer Perceptron (MLP)* (Gardner and Doring 1998): a feed-forward neural network that uses backpropagation for model training. MLP updates the weights between neurons to minimise the prediction error and can often learn nonlinear features and patterns within the dataset. MLP can often generalise well to new unseen data, but it is slow in convergence and often stuck in local minima.
- 4 *Support Vector Machine (SVM)* (Pisner and Schnyer 2020): a well-known model that classifies inputs by creating a hyperplane to separate each class. Due to its kernel trick of implicitly mapping the input features into higher-dimensional feature spaces, SVM can efficiently perform nonlinear classification problems.
- 5 *Decision Tree (DT)* (Myles et al. 2004): is a non-parametric model that uses a tree-like structure to construct its model for making decisions. A DT comprises a series of nodes and branches representing the decision rules inferred from the input features. The main benefit of DT lies in its interpretability due to its rule-based logic.
- 6 *K Nearest Neighbour (kNN)* (Zhu et al. 2020): is a classifier that simply stores the given input features and classifies input data via some similarity metrics. kNN, as a non-parametric method with a simple approach to the typical classification setup, is widely adopted in many domains.
- 7 *Bootstrap aggregation (Bagging)* (Bühlmann 2012): is an ensemble method that trains multiple weak classifiers on a random subset of the given dataset. Bagging can often reduce the variance within a noisy dataset, and the output of the Bagging model is obtained by averaging the predictions by its weak internal classifiers. In contrast to Boosting methods, the weaker classifiers in Bagging are trained independently, and Bagging can often avoid over-fitting in high-variance datasets.
- 8 *Adaptive Boosting (AdaBoost)* (Hastie et al. 2009): is a meta-classifier that works in conjunction with other learning algorithms. AdaBoost uses a weighted sum to combine the predicted output from other weak classifiers to output its final prediction. In particular, AdaBoost adaptively improves its performance under challenging classes by using additional weak learners to minimise the misclassification induced by previous classifiers.
- 9 *Random Forest (RF)* (Pal 2005): an ensemble method that internally combines the predictive power of multiple Decision Trees for outputting a final prediction. RF trains each DT using a random subset of the input features and uses bootstrapping approach by subsampling random features with replacement.
- 10 *Extremely Randomised Trees (ERT)* (Geurts et al. 2006): an ensemble method that behaves similarly to RF, which trains multiple weak decision trees as the weak learner. However, unlike RF, each decision tree is trained with the whole dataset instead of subsampling. ERT also randomly select the split point to split nodes in decision trees instead of finding the optimal split as in RF. ERT can often train faster than RF and attains a lower overall variance due to the random splitting of nodes.
- 11 *Gradient Boosting Machine (GBM)* (Friedman 2001): is an ensemble technique that uses multiple weak prediction models built in a stage-wise fashion. Similar to RF, a decision tree is a common choice for being the weak prediction model within GB. However, in GB, each weak predictor is trained to correct the residuals of its predecessor, and as a result, it can often achieve a lower model bias than RF.
- 12 *Extreme Gradient Boosting (XGB)* (Chen and Guestrin 2016): a tree boosting method that uses the same idea of gradient boosting. In contrast to GB, XGB uses the second-order derivatives method to find the optimal constant in each terminal node and uses regularisation of the tree to avoid overfitting.
- 13 *Voting* (Ruta and Gabrys 2005): an ensemble method that uses voting to combine the predicted outputs from several individual predictors. The Voting ensemble can use the majority vote or the average predicted probability to predict the class labels. The Voting ensemble can incorporate arbitrary predic-

tors and often balance out the weakness between the nested predictors.

14 *Stacked Generalisation (Stacking)* (Naimi and Balzer 2018): an ensemble method that allows one to combine several different prediction algorithms by stacking the individual predictors' output and uses a final classifier to compute a final prediction. The stacking approach is straightforward, and different kinds of predictors can be easily combined, which could potentially use the strength of some models to compensate for the weaknesses of other models.

Experiments

In this section, we present the experimental results from benchmarking the discussed Machine Learning models against various IoT anomaly detection datasets.

Experimental setups

We randomly split the dataset for each target IoT anomaly detection dataset by using 20% of the data as the unseen test set. Hyperparameter optimisation is then performed by using the training set for testing various hyperparameters, thereby computing the corresponding F1-score with five-fold cross-validation. During the hyperparameter optimisation process, we kept the test set hidden along with the five-fold cross-validation to ensure that the optimised hyperparameters can perform well robustly across a wide range of domain subsets. This procedure is repeated 45 times, and we use the best-performing hyperparameters under this cross-validation to report the following scores on the test set.

The reported score is computed using the predictive results of the models using the tuned hyperparameters. The experimental results are performed on an HPC cluster with 32 requested CPU cores of Intel Xeon E5-2680 V3 2.50GHz processor, 128GB RAM, and NVIDIA V100 SXM2 GPU equipped with 16GB vRAM. In total, this study performed five-fold cross-validation \times 45 trials \times 14 models \times 5 datasets, with a total of 15,750 model training episodes.

Evaluation metrics

In this study, we use *accuracy*, *precision*, *recall*, and *F1-score* to evaluate our models. In the classification problem, there are four possible results in the model prediction: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP refers to the number of instances that have been correctly identified as normal. TN represents the number of instances that are classified correctly as malicious. FP represents the number of malicious instances that are wrongly classified as normal. FN refers to the number of instances that misclassify normal

data as malicious data. Our evaluation metrics are then given by

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (6)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (7)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (8)$$

$$\text{F1-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \quad (9)$$

Accuracy is an intuitive measure summarising the ratio of correctly predicted observation to total observation. However, when the dataset contains an asymmetric number of classes, which is the case for some of our datasets, accuracy cannot convey the whole picture of model performance. Precision refers to the ratio of correctly predicted positive observations to the total predicted observations. A high value in precision denotes a low false positive rate. Recall, sometimes referred to as sensitivity, is the ratio of correctly predicted positive observation to the total number of that class. Recall answers the percentage of correct positives our model can capture. The F1 score is a weighted average of precision and recall. F1 score is usually more helpful information than accuracy, especially in datasets with an uneven class distribution.

Experimental results

All models are trained on the same training set and evaluated on the same test set, and we use five-fold cross-validation on the train set to select the set of hyperparameters that achieved the highest performance in the F1 score. The numerical results from the models with the best-performing hyperparameters are summarised and presented in Table 2.

The results are indicative of the overall performance of the various approaches. Traditional linear models, such as Ridge, perform poorly across all five datasets. The reason is likely due to the limited expressiveness of the model to learn the nonlinearity input features. Models such as DT can achieve a decent result on more straightforward datasets with binary labels, but their performance drops rapidly once the dataset contains more label classes. Moreover, models like RF exhibit a higher accuracy value but with a considerably low precision value, indicating that the model is biased towards the classes with a higher number of instances leading to a high false positive rate.

Table 2 Experimental results of the benchmarking ML models

| Model ensemble? | Dataset | Type | Metric | | | |
|-----------------|---------|--------------|----------|-----------|--------|------|
| | | | Accuracy | Precision | Recall | F1 |
| Ridge | IoTID20 | Binary | 0.94 | 0.83 | 0.55 | 0.57 |
| | | Multi-Cat | 0.77 | 0.78 | 0.50 | 0.54 |
| | | Multi-SubCat | 0.54 | 0.43 | 0.35 | 0.32 |
| | IoT23 | Binary | 0.86 | 0.43 | 0.50 | 0.46 |
| | | Multi-Cat | 0.81 | 0.78 | 0.54 | 0.57 |
| | | Multi-SubCat | 0.45 | 0.55 | 0.69 | 0.39 |
| NB | IoTID20 | Binary | 0.45 | 0.55 | 0.69 | 0.39 |
| | | Multi-Cat | 0.57 | 0.47 | 0.58 | 0.47 |
| | | Multi-SubCat | 0.46 | 0.34 | 0.36 | 0.32 |
| | IoT23 | Binary | 0.81 | 0.45 | 0.48 | 0.46 |
| | | Multi-Cat | 0.43 | 0.32 | 0.35 | 0.31 |
| | | Multi-SubCat | 0.95 | 0.95 | 0.60 | 0.65 |
| MLP | IoTID20 | Binary | 0.95 | 0.95 | 0.60 | 0.65 |
| | | Multi-Cat | 0.78 | 0.84 | 0.53 | 0.57 |
| | | Multi-SubCat | 0.50 | 0.45 | 0.39 | 0.36 |
| | IoT23 | Binary | 0.86 | 0.43 | 0.50 | 0.46 |
| | | Multi-Cat | 0.57 | 0.06 | 0.11 | 0.08 |
| | | Multi-SubCat | 0.96 | 0.91 | 0.71 | 0.78 |
| SVM | IoTID20 | Binary | 0.96 | 0.91 | 0.71 | 0.78 |
| | | Multi-Cat | 0.80 | 0.80 | 0.57 | 0.62 |
| | | Multi-SubCat | 0.58 | 0.49 | 0.42 | 0.38 |
| | IoT23 | Binary | 0.90 | 0.95 | 0.62 | 0.67 |
| | | Multi-Cat | 0.49 | 0.42 | 0.41 | 0.43 |
| | | Multi-SubCat | 0.96 | 0.91 | 0.73 | 0.79 |
| DT | IoTID20 | Binary | 0.96 | 0.91 | 0.73 | 0.79 |
| | | Multi-Cat | 0.76 | 0.31 | 0.40 | 0.35 |
| | | Multi-SubCat | 0.58 | 0.39 | 0.42 | 0.39 |
| | IoT23 | Binary | 0.86 | 0.43 | 0.50 | 0.46 |
| | | Multi-Cat | 0.64 | 0.18 | 0.20 | 0.18 |
| | | Multi-SubCat | 0.98 | 0.96 | 0.90 | 0.93 |
| kNN | IoTID20 | Binary | 0.98 | 0.96 | 0.90 | 0.93 |
| | | Multi-Cat | 0.85 | 0.81 | 0.77 | 0.78 |
| | | Multi-SubCat | 0.60 | 0.56 | 0.51 | 0.52 |
| | IoT23 | Binary | 0.92 | 0.89 | 0.85 | 0.87 |
| | | Multi-Cat | 0.71 | 0.23 | 0.15 | 0.20 |
| | | Multi-SubCat | 0.99 | 0.99 | 0.93 | 0.96 |
| Bagging | IoTID20 | Binary | 0.99 | 0.99 | 0.93 | 0.96 |
| | | Multi-Cat | 0.87 | 0.84 | 0.80 | 0.81 |
| | | Multi-SubCat | 0.65 | 0.65 | 0.55 | 0.56 |
| | IoT23 | Binary | 1.00 | 1.00 | 1.00 | 1.00 |
| | | Multi-Cat | 0.85 | 0.51 | 0.43 | 0.49 |
| | | Multi-SubCat | 0.99 | 0.98 | 0.89 | 0.93 |
| AdaBoost | IoTID20 | Binary | 0.99 | 0.98 | 0.89 | 0.93 |
| | | Multi-Cat | 0.81 | 0.85 | 0.63 | 0.66 |
| | | Multi-SubCat | 0.61 | 0.60 | 0.46 | 0.46 |
| | IoT23 | Binary | 0.99 | 0.99 | 0.98 | 0.99 |
| | | Multi-Cat | 0.84 | 0.36 | 0.32 | 0.33 |
| | | Multi-SubCat | 0.94 | 0.47 | 0.50 | 0.48 |
| RF | IoTID20 | Binary | 0.94 | 0.47 | 0.50 | 0.48 |
| | | Multi-Cat | 0.66 | 0.13 | 0.20 | 0.16 |
| | | Multi-SubCat | 0.29 | 0.03 | 0.11 | 0.05 |
| | IoT23 | Binary | 0.86 | 0.43 | 0.50 | 0.46 |
| | | Multi-Cat | 0.57 | 0.06 | 0.11 | 0.08 |
| | | Multi-SubCat | 0.57 | 0.06 | 0.11 | 0.08 |

Table 2 (continued)

| Model ensemble? | | Dataset | Type | Metric | | | |
|-----------------|---|---------|--------------|----------|-----------|--------|------|
| | | | | Accuracy | Precision | Recall | F1 |
| ERT | ✓ | IoTID20 | Binary | 0.94 | 0.47 | 0.50 | 0.48 |
| | | | Multi-Cat | 0.75 | 0.35 | 0.39 | 0.37 |
| | | | Multi-SubCat | 0.52 | 0.22 | 0.31 | 0.25 |
| | | IoT23 | Binary | 0.97 | 0.98 | 0.89 | 0.93 |
| | | | Multi-Cat | 0.84 | 0.30 | 0.31 | 0.30 |
| | | | Multi-SubCat | 0.61 | 0.53 | 0.48 | 0.48 |
| GBM | ✓ | IoTID20 | Binary | 0.95 | 0.96 | 0.63 | 0.69 |
| | | | Multi-Cat | 0.83 | 0.85 | 0.69 | 0.70 |
| | | | Multi-SubCat | 0.61 | 0.53 | 0.48 | 0.48 |
| | | IoT23 | Binary | 0.86 | 0.43 | 0.50 | 0.46 |
| | | | Multi-Cat | 0.89 | 0.38 | 0.38 | 0.38 |
| | | | Multi-SubCat | 0.61 | 0.53 | 0.48 | 0.48 |
| XGB | ✓ | IoTID20 | Binary | 0.99 | 0.99 | 0.94 | 0.96 |
| | | | Cat | 0.87 | 0.84 | 0.81 | 0.81 |
| | | | SubCat | 0.66 | 0.65 | 0.56 | 0.57 |
| | | IoT23 | Binary | 1.00 | 1.00 | 1.00 | 1.00 |
| | | | Multi | 0.98 | 0.74 | 0.63 | 0.66 |
| | | | Multi-SubCat | 0.67 | 0.65 | 0.57 | 0.58 |
| Stacking | ✓ | IoTID20 | Binary | 0.99 | 0.98 | 0.94 | 0.96 |
| | | | Multi-Cat | 0.87 | 0.83 | 0.78 | 0.80 |
| | | | Multi-SubCat | 0.67 | 0.65 | 0.57 | 0.58 |
| | | IoT23 | Binary | 0.88 | 0.85 | 0.90 | 0.87 |
| | | | Multi-Cat | 0.59 | 0.32 | 0.36 | 0.35 |
| | | | Multi-SubCat | 0.63 | 0.61 | 0.54 | 0.53 |
| Voting | ✓ | IoTID20 | Binary | 0.98 | 0.95 | 0.93 | 0.94 |
| | | | Multi-Cat | 0.85 | 0.84 | 0.74 | 0.77 |
| | | | Multi-SubCat | 0.63 | 0.61 | 0.54 | 0.53 |
| | | IoT23 | Binary | 0.99 | 0.99 | 0.95 | 0.97 |
| | | | Multi-Cat | 0.82 | 0.39 | 0.41 | 0.41 |
| | | | Multi-SubCat | 0.63 | 0.61 | 0.54 | 0.53 |

In contrast, ensemble models can utilise multiple weaker predictors' predictive power to improve the overall predictive accuracy. For example, XGB can achieve superior results on most datasets due to its utilisation of multiple DT under the hood. Most ensemble models can consistently achieve higher model performance across a wide range of IoT anomaly scenarios. For example, ensemble models like Stacking and XGB have consistently high performance across the more trivial dataset IoTID20 Binary and the more complicated IoTID20 Multi-SubCat. When the number of classes increases, most traditional models like RF have a dramatic drop in performance, while ensemble models are still able to maintain performance in the higher end of the spectrum

Hyperparameters optimisation

Hyperparameters tend to be one of the aspects that can mystify ML due to models' sensitiveness towards the choice of hyperparameters. A decent model can achieve mediocre results due to a poor choice of hyperparameters. In this section, we study the models' sensitivity

toward their choice of hyperparameters. All of the results are obtained via 45 trials of hyperparameter optimisation for each of the models. As mentioned earlier, the tree-structured Parzen estimator (Bergstra et al. 2011) is used for our Bayesian Optimisation procedure by fitting multiple Gaussian Mixture Models for searching the most promising hyperparameter values. In the following, we focus on the multi-class anomaly detection dataset IoTID20 Multi-SubCat and provide visualisation of the hyperparameter optimisation procedure.

Visualising optimisation history

The hyperparameter optimisation history of a few selected models is shown in Fig. 1, which includes a few traditional ML models and ensemble models. The figure illustrates the optimisation history of the object value, which is the F1 score obtained with a five-fold cross-validation on the training set. The plots include the history of the 45 trials horizon and keep track of the best-obtained value. The scattering of the objective value also illustrates the consistency of each method. For example,

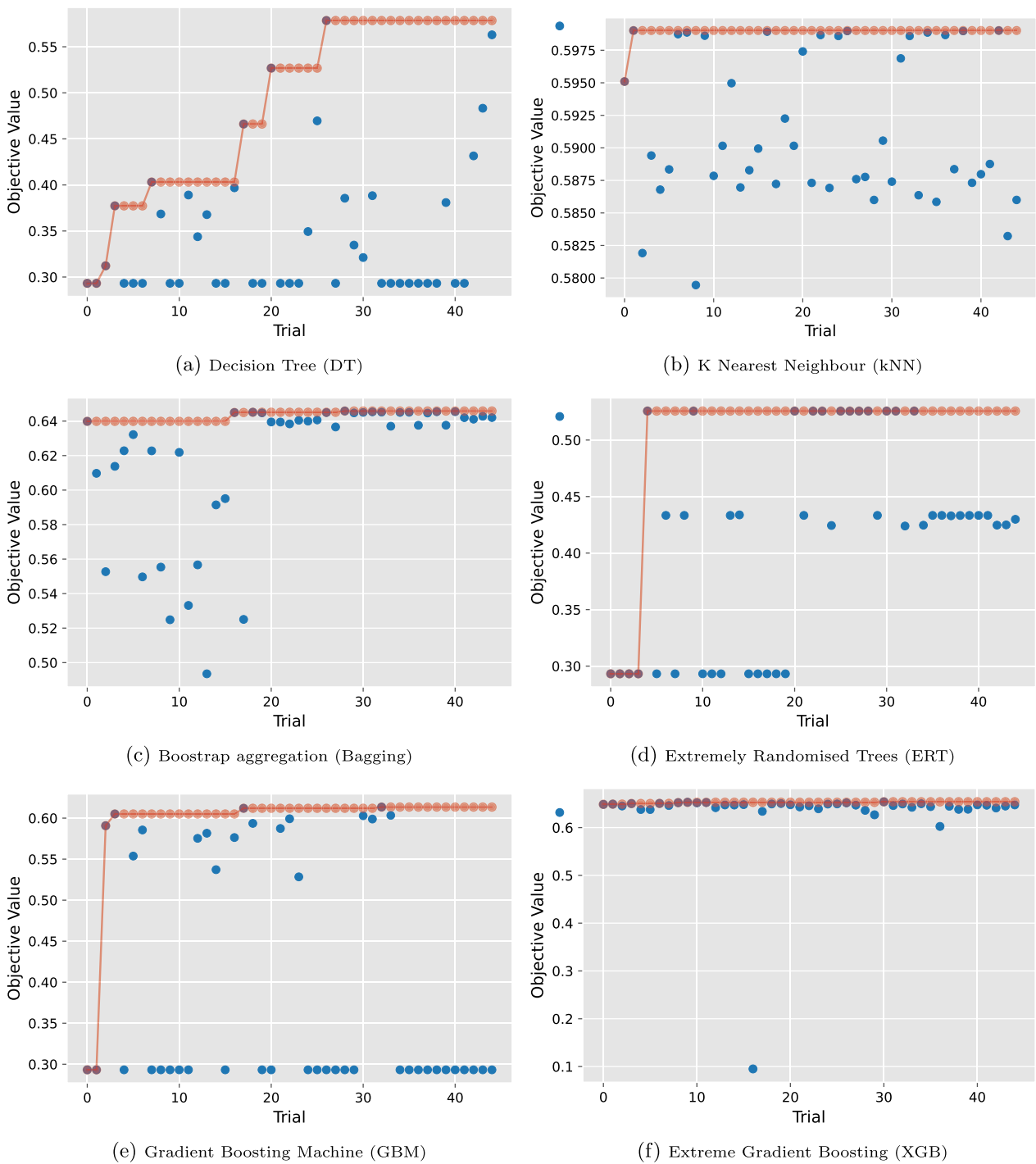


Fig. 1 Plots of the history of hyperparameter optimisation process against the achieved F1 objective values

traditional methods such as DT can achieve an F1 score above 0.55 in their training set. However, its optimisation history has a significant variance. Moreover, the same model only obtains an F1 score of 0.39 in its test set (see Table 2). The result suggests that models like DT

are prone to overfitting on their training set despite using k-fold cross-validation (Fig. 2).

On the other hand, ensemble models seem to have less variance in their optimisation history. The extreme gradient boosting model contains an outlier that achieves

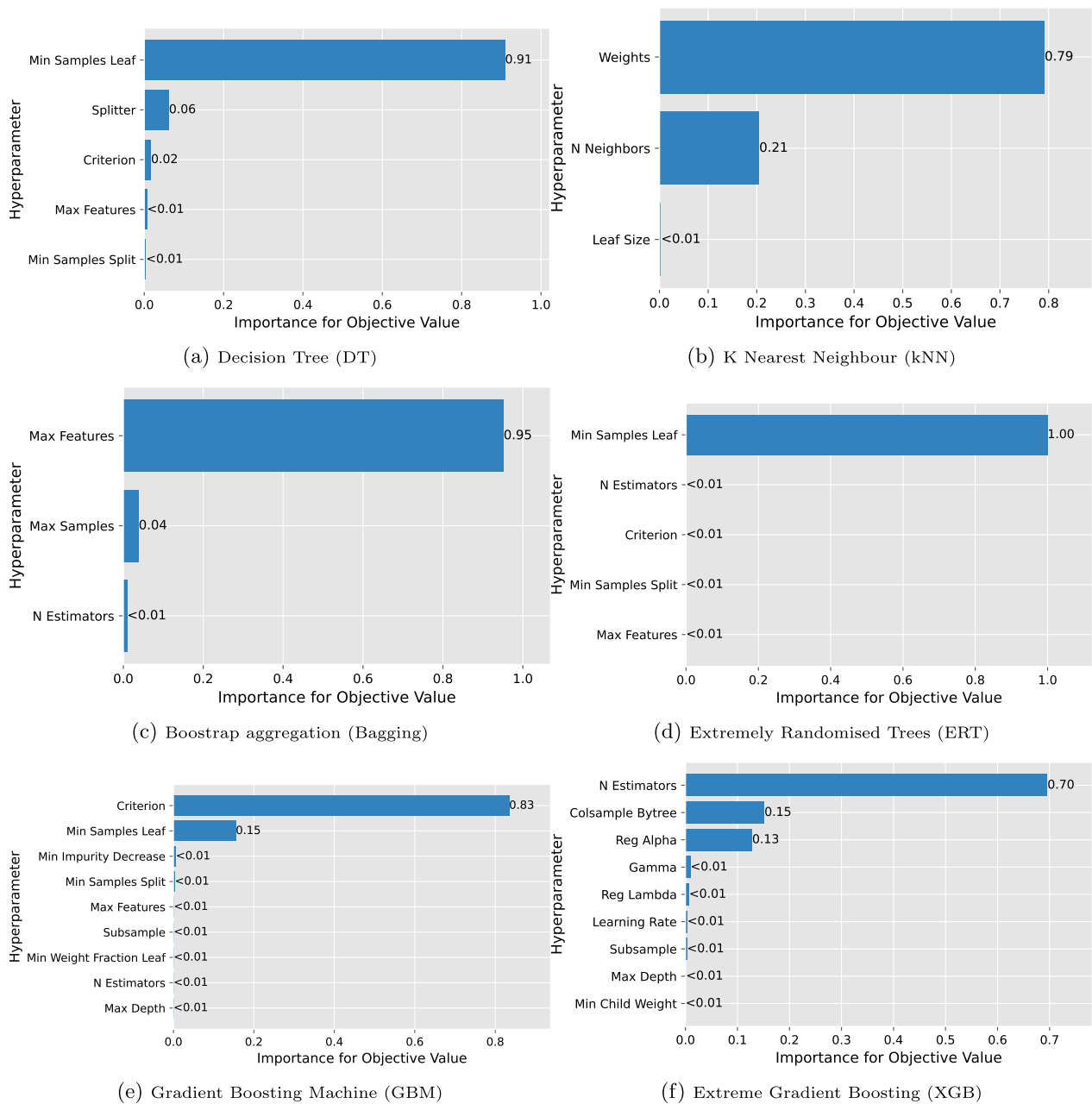


Fig. 2 Plots of quantifying the importance of each hyperparameter with respect to the objective value via the functional ANOVA framework

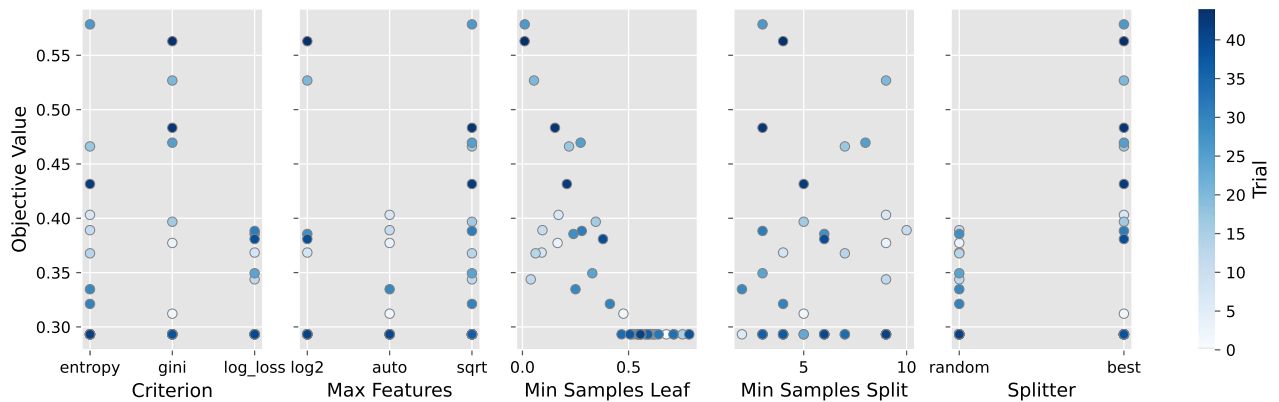
poor performance (around 0.1); however, generally, most attainable objective values tend to fluctuate in the higher end of the spectrum. As a result, ensemble models are less prone to the choice of hyperparameters. Poor choices of hyperparameters can still lead to inferior performance. However, the end results from ensemble models are still more consistent than traditional results due to the way that ensemble models aggregate results from their internal estimators.

Estimating the importance of hyperparameter

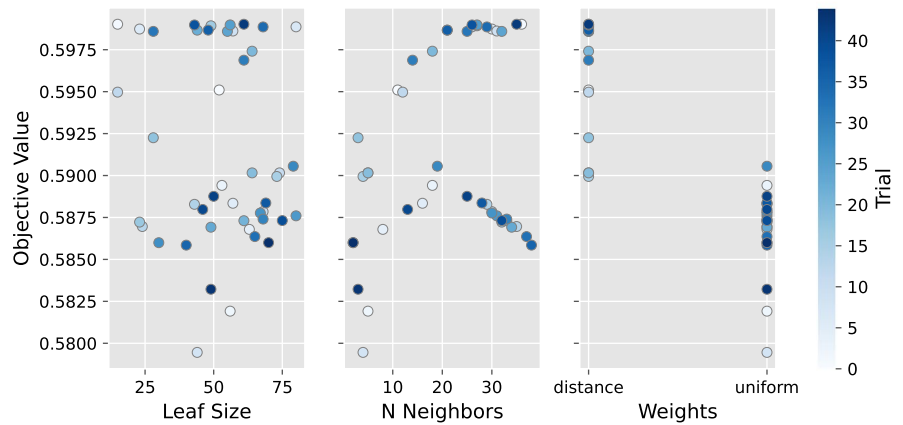
The performance of the evaluated methods can be heavily dependent on the hyperparameter settings. Therefore, in our parametric study, we leverage a functional ANOVA framework to quantify the importance of the hyperparameters used in the various modes (Hutter et al. 2014). The estimation leverages random forest models to analyse the target model’s variance, which helps decompose the importance of its corresponding hyperparameters.

Figure 3 illustrates the critical features in our best-performing model. This measure is obtained by updating the hyperparameters and evaluating the model F1 score whilst treating the model as a black-box function. The fANOVA framework (Hutter et al. 2014) is then

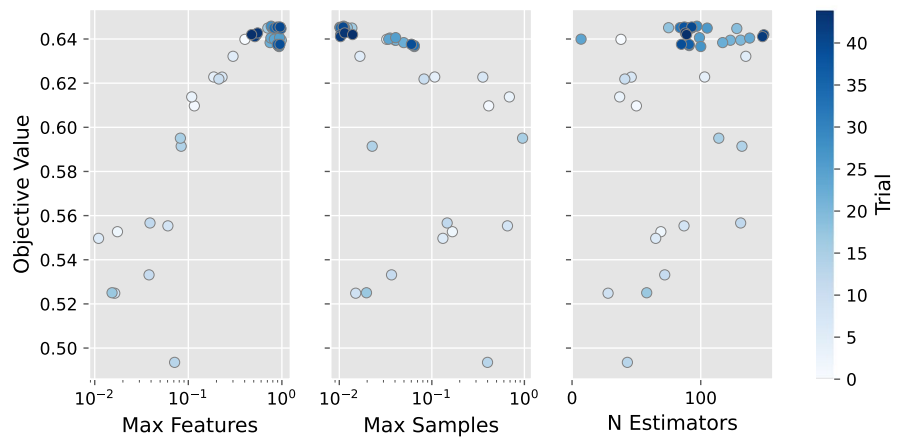
used to estimate how the changes of each hyperparameter affect the final F1 score on the training set. The degree of importance can have a high variation among hyperparameters, and the numerical value presented



(a) Decision Tree (DT)

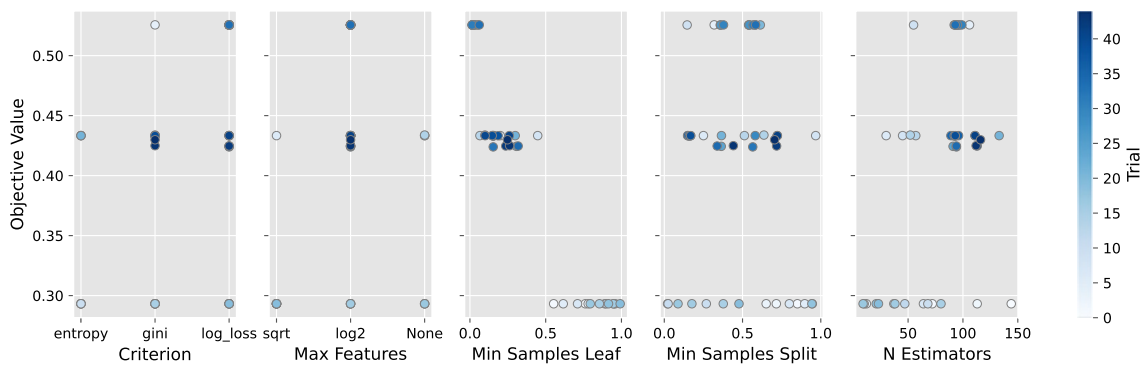


(b) K Nearest Neighbour (kNN)

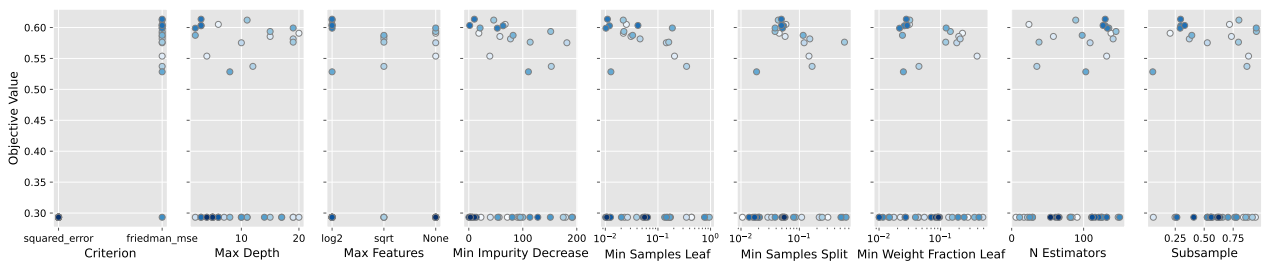


(c) Bootstrap aggregation (Bagging)

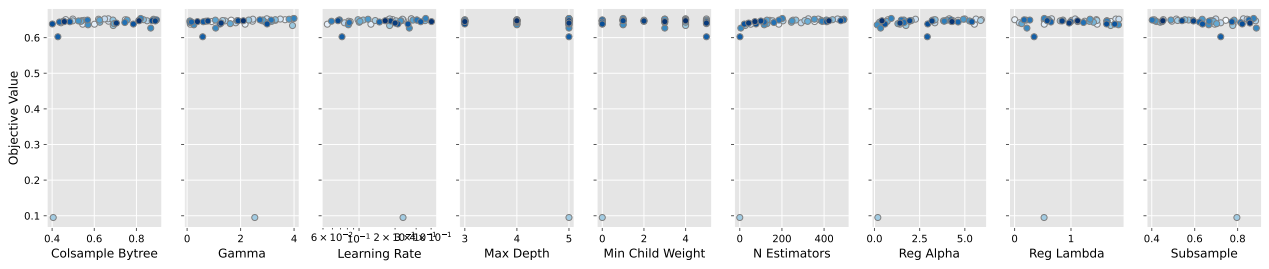
Fig. 3 Slice plot of parameter relationship with respect to the F1 objective value



(d) Extremely Randomised Trees (ERT)



(e) Gradient Boosting Machine (GBM)



(f) Extreme Gradient Boosting (XGB)

Fig. 3 continued

in Fig. 3 can be seen as the proportion of effect that the hyperparameter exhibit on the final F1 score.

We can observe common patterns among various models. For example, the “Min Samples Leaf” hyperparameter in both tree-based methods (DT and ERT) is the most contributing factor. The parameter controls the minimum number of samples required to be a leaf node, which determines the resulting structure of the trees. Similarly, we can see that parameters that control the number of features or estimators also contribute tremendously to the F1 score (“Max Features” in Bagging and “N Estimators” in XGB). The former is the maximum threshold of features used to train its internal sub-estimator, while the latter directly controls the number of sub-estimators. Finally, the “Criterion” hyperparameter in GBM controls the metric used to measure the quality of a split in GBM, which seems to be essential in GBM. Overall, it

can be seen that only a small set of hyperparameters generally contribute towards the actual model performance. However, the hyperparameters influential to the model might differ, and the set of hyperparameters generally differs significantly across model types. Therefore, it is essential to obtain a rough estimate of hyperparameters’ importance when deploying ML models on IoT detection tasks, such that we can account for the model’s sensitivity to hyperparameters and tune their corresponding value when necessary.

Visualising parametric relationship

Continuing from the previous parametric study, we further study the relationship between each hyperparameter with respect to the model performance on its final F1 score. Figure 3 illustrates a set of subplots that plots the value of each hyperparameter against their objective

values. Scatters with the same hue of colour refer to the same trial, and the scatters visualise the attainable objective values for various sets of hyperparameters and their distribution.

Some subplots illustrate a clear relationship between the choice of hyperparameters, while some hyperparameters do not indicate any apparent effect on the objective value. For example, most hyperparameters for GBM in Fig. 3e do not appear to contain any clear indications of best value, with the exception for “Criterion”, where the Friedman Mean Square Error (MSE) (Friedman 2001) is superior to the traditional MSE. In contrast, hyperparameters for Bagging in Fig. 3c exhibit clear trends. On the one hand, a higher value of “Max Features” (the maximum number of features used to train sub-estimators) seems to provide a better objective value; on the other hand, a lower value of “Max Samples” (the maximum subset of the dataset used to sample for training sub-estimators) seems to yield better results. Moreover, most hyperparameters for XGB appear to consistently score high objective value, with some exceptional outliers that occur when both “Min Child Weight” and “N Estimators” are close to zero.

It should be noted that the plots in Fig. 1 illustrates the complex interaction in-between hyperparameters; therefore, it is difficult to provide a definitive conclusive decision on the effect of a single hyperparameter. However, the general trend of a hyperparameter can illustrate the model’s sensitivity towards such a hyperparameter. Moreover, the use of Bayesian Optimisation techniques like tree-structured Parzen estimator (Bergstra et al. 2011) helps to search for the set of hyperparameters that are best suited for our objectives.

Most influential features for detecting cyber attacks in IoT devices

This section summarises the numerous cybersecurity attacks studied in this paper and the type of features that are useful for identifying such an attack. The degree of influential for each type of cyber attacks is computed using the absolute value of the pearson correlation coefficients. The following summarises the top 3 most important features for detecting the studied IoT cyber attacks.

- i. **Distributed denial of service (DDoS):** DDoS attack is a form of a malicious attempt to disrupt the normal traffic of some targeted server. These traffic flows are detected as part of a DDoS attack because of the number of flows directed to the same IP address. The essential network features that can identify whether a network package belongs to a DDoS attack are: (1) the originating or responding port number, (2) the maximum time between two packets sent in the backward direction, and (3) the number of packets with SYN / ACK flag being set.
- ii. **Man-in-the-middle attack (MITM):** MITM refers to an attacker being positioned between the IoT device and the communication endpoint to intercept and potentially alter data travelling between them. The network features that contribute primarily to identifying MITM attacks are: (1) the number of packets with ACK flag set, (2) the size (length) of packet in forward direction, and (3) the size of the package in the backward direction.
- iii. **Host port scan:** Port scanning is a method where attackers scope out their target environment by sending packets to specific ports on a host and using the responses to find vulnerabilities. Notable network features that identify such attack includes: (1) the number of packets with ACK flag set, (2) the size of packet in backward direction, and (3) the total number of bytes sent in the initial window in the backward direction.
- iv. **Mirai—HTTP flooding/ACK flooding:** *Mirai* is a malware that exploits security holes in IoT devices and attempts to harness the collective power of millions of IoT devices into botnets for launching distributed attacks. For *Mirai*-infected devices that are performing HTTP Flooding attacks (a type of DDoS), the following network features are most helpful in identifying them: (1) the size of packet in forward direction, (2) the size of packet in backward direction, and (3) the total number of bytes sent in the initial window in the backward direction. The same goes for ACK flooding attack.
- v. **Mirai—UDP flooding:** UDP flooding is a type of DDoS attack where a large number of User Datagram Protocol (UDP) packets are sent to a targeted server to overwhelm that device’s ability to process and respond. The identifiable network features include: (1) the number of packets with ACK flag set, (2) the count of packets with at least 1 byte of TCP data payload in the forward direction, and (3) the number of backward packets per second.
- vi. **Mirai—Host brute force:** Brute force attacks executed by *Mirai*-infected devices attempt to gain access to a site or server by systematically trying every possible combination. The typical preventative approach of blocking brute force attacks by locking out IP addresses would be less effective because the *Mirai* botnet is a distributed attack. The important identifiable network features include: (1) the number of packets with ACK / SYN flag set, (2) the size of packet in backward direction, and (3) the number of flow bytes per second.

The summarised network features insights on popular cyber security attacks are useful for IoT device designers and network security experts to design an integrated system that can monitor and potentially identify and prevents IoT devices from being hijacked by the cyber threats. Interested reader can refer to reviews on cybersecurity like (Hussain et al. 2020; Lu et al. 2021; Mijwil et al. 2023) for a more in-depth discussion on the emerging threats within the cyberspace.

Conclusion

This paper presents a comprehensive study on applying ensemble machine learning methodologies for detecting cybersecurity attacks in an IoT environment. A wide range of traditional and ensemble machine learning models is analysed, along with a Bayesian Optimisation framework and analysis of the sensitivity of the choice of hyperparameters on model performance. This study highlighted the set of influential configurations on the popular models and the optimisation approach of automatic tuning of such hyperparameters with Bayesian Optimisation. In addition, we evaluate the various models against a wide range of IoT anomaly datasets to evaluate each model's robustness. Empirical evidence suggests that Tree-based boosting methods like GBM and XGB can consistently achieve high accuracy and recall. Moreover, the parametric study on hyperparameters illuminates their importance and effects on the final model performance. Most machine learning models contain a large set of hyperparameters, but only a small set exhibit a strong influence on the model performance. Therefore, such hyperparameters should either be derived from best practice or by performing the present parametric study to determine important hyperparameters.

Cybersecurity, IoT devices and related technologies are the current focus in many multi-discipline domains, for example, electronics manufacturers on designing smart devices, network security analysts on improving the current network protocol, and data scientists on better utilising the vast amount of available data. In particular, people rely heavily on IoT devices and services due to the wide range of applications in our daily lives.

Future direction

Looking ahead, the convergence of cybersecurity, IoT devices, and related technologies continues to be a focal point across multidisciplinary domains. The widespread reliance on IoT devices in our daily lives, with applications spanning from smart homes to industrial settings, underscores the importance of advancing cybersecurity measures.

A possible direction for future studies is testing more variety of networks protocol and device types as they can provide a more diverse set of representations for the

model to extract more meaningful features. Investigating how different IoT device types interact with various network protocols and assessing the unique challenges they pose for anomaly detection is crucial.

Given the critical nature of IoT systems in domains such as healthcare, smart cities, and critical infrastructure, enhancing real-time threat detection capabilities is paramount. Future studies should focus on developing models that can deliver instantaneous responses to security threats, minimising the potential for breaches.

Leveraging transfer learning and federated learning techniques in IoT cybersecurity can also enable the sharing of knowledge and models across devices and networks without compromising sensitive data. Future work should explore these methodologies for scalable and efficient anomaly detection, while considering privacy-preserving techniques within the realm of anomaly detection.

Author contributions

Conceptualization, T.L. F.F. and A.B.; methodology, T.L., F.F. and A.B.; software, T.L.; validation, T.L., F.F. A.B and F.S.; formal analysis, T.L.; investigation, T.L., F.F. A.B and F.S.; writing --original draft preparation, T.L.; writing --review and editing, T.L., F.F. A.B and F.S.; visualisation, T.L.; supervision, F.F. A.B and F.S.; project administration, T.L.T.L., F.F. A.B and F.S.; funding acquisition, F.F. and A.B.

Funding

The funding was provided by University of Western Sydney.

Availability of data and materials

The datasets analysed during the current study are available at <https://sites.google.com/view/iot-network-intrusion-dataset/home> (accessed on 15 July 2022) and <https://www.stratosphereips.org/datasets-iot23> (accessed on 22 July 2022).

Declarations

Consent for publication

All authors have read and agreed to the published version of the manuscript.

Competing interests

The authors declare that they have no competing interests.

Received: 17 January 2023 Accepted: 2 April 2024

Published online: 12 June 2024

References

- Abraham B, Chuang A (1989) Outlier detection and time series modeling. *Technometrics: J Stat Phys Chem Eng Sci* 31(2):241–248. <https://doi.org/10.1080/00401706.1989.10488517>
- Akgun D, Hizal S, Cavusoglu U (2022) A new ddos attacks intrusion detection model based on deep learning for cybersecurity. *Comput Secur* 118:102748
- Aleskerov E, Freisleben B, Rao B (1997) Cardwatch: A neural network based database mining system for credit card fraud detection. In: *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFER)*, pp. 220–226. IEEE. <https://doi.org/10.1109/CIFER.1997.618940>
- Al-Garadi MA, Mohamed A, Al-Ali AK, Du X, Ali I, Guizani M (2020) A survey of machine and deep learning methods for internet of things (IoT)

- security. *IEEE Commun Surv Tutor* 22(3):1646–1685. <https://doi.org/10.1109/COMST.2020.2988293>
- Alghamdi R, Bellaïche M (2021) A deep intrusion detection system in lambda architecture based on edge cloud computing for IoT. In: 2021 4th international conference on artificial intelligence and Big Data (ICAIBD). IEEE, pp 561–566. <https://doi.org/10.1109/ICAIBD51990.2021.9458974>
- Bergstra J, Bardenet R, Bengio Y, Kégl B (2011) Algorithms for hyper-parameter optimization. *Adv Neural Inform Process Syst* 24
- Borgia E (2014) The Internet of Things vision: key features, applications and open issues. *Comput Commun* 54:1–31. <https://doi.org/10.1016/j.comcom.2014.09.008>
- Bühlmann P (2012) Bagging, boosting and ensemble methods. In: *Handbook of computational statistics*. Springer, pp 985–1022
- Carbonell JG, Michalski RS, Mitchell TM (1983) An overview of machine learning. *Mach Learn* 3–23
- Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. *ACM Comput Surv (CSUR)* 41(3):1–58. <https://doi.org/10.1145/1541880.1541882>
- Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp 785–794. <https://doi.org/10.1145/2939672.2939785>
- Cook AA, Misirlı G, Fan Z (2020) Anomaly detection for IoT time-series data: a survey. *IEEE Internet Things J* 7(7):6481–6494. <https://doi.org/10.1109/JIOT.2019.2958185>
- Dieterich TG (2022) Machine learning for sequential data: a review. In: *Joint IAPR international workshops on statistical techniques in pattern recognition (SPR) and structural and syntactic pattern recognition (SSPR)*. Springer, pp 15–30
- Flach PA, Lachiche N (2004) Naive Bayesian classification of structured data. *Mach Learn* 57(3):233–269. <https://doi.org/10.1023/B:MACH.0000039778.69032ab>
- Friedman JH (2001) Greedy function approximation: a gradient boosting machine. *Ann Stat* 1189–1232
- Fujimaki R, Yairi T, Machida K (2005) An approach to spacecraft anomaly detection problem using kernel feature space. In: *Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery in data mining*, pp 401–410. <https://doi.org/10.1145/1081818.1081917>
- García S, Parmisano A, Erquiaga MJ (2020) IoT-23: A labeled dataset with malicious and benign IoT network traffic. *Zenodo*. <https://doi.org/10.5281/zenodo.4743746>
- Gardner MW, Dorling SR (1998) Artificial neural networks (the multilayer Perceptron)-a review of applications in the atmospheric sciences. *Atmos Environ* 32(14–15):2627–2636. [https://doi.org/10.1016/S1352-2310\(97\)00447-0](https://doi.org/10.1016/S1352-2310(97)00447-0)
- Geraldes R, Gonçalves A, Lai T, Villerabel M, Deng W, Salta A, Nakayama K, Matsuo Y, Prendinger H (2019) UAV-based situational awareness system using deep learning. *IEEE Access: Pract Innov Open Solut* 7:122583–122594. <https://doi.org/10.1109/ACCESS.2019.2938249>
- Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. *Mach Learn* 63(1):3–42. <https://doi.org/10.1007/s10994-006-6226-1>
- González-Zamar M-D, Abad-Segura E, Vázquez-Cano E, López-Meneses E (2020) IoT technology applications-based smart cities: Research analysis. *Electronicsweek* 9(8):1246. <https://doi.org/10.3390/electronics9081246>
- Gou Q, Yan L, Liu Y, Li Y (2013) Construction and strategies in IoT security system. In: 2013 IEEE international conference on green computing and communications and IEEE Internet of Things and IEEE cyber, physical and social computing. IEEE, pp 1129–1132. <https://doi.org/10.1109/GreenCom-IThings-CPSCom.2013.195>
- Hassan WH (2019) Current research on Internet of Things (IoT) security: a survey. *Comput Netw* 148:283–294. <https://doi.org/10.1016/j.comnet.2018.11.025>
- Hassija V, Chamola V, Saxena V, Jain D, Goyal P, Sikdar B (2019) A survey on IoT security: application areas, security threats, and solution architectures. *IEEE Access: Pract Innov Open Solut* 7:82721–82743. <https://doi.org/10.1109/ACCESS.2019.2924045>
- Hastie T, Rosset S, Zhu J, Zou H (2009) Multi-class adaboost. *Stat Interface* 2(3):349–360. <https://doi.org/10.4310/SII.2009.v2.n3.a8>
- Hawkins DM (1980) *Identification of outliers*, vol 11. Springer, New York
- Hoerl AE, Kennard RW (1970) Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics: J Stat Phys Chem Eng Sci* 12(1):55–67
- Hussain A, Mohamed A, Razali S (2020) A review on cybersecurity: Challenges & emerging threats. In: *Proceedings of the 3rd international conference on networking, information systems & security*, pp 1–7
- Hutter F, Hoos H, Leyton-Brown K (2014) An efficient approach for assessing hyperparameter importance. In: *Proceedings of the 31st international conference on machine learning*, pp 754–762
- Hwang YH (2015) IoT security & privacy: threats and challenges. In: *Proceedings of the 1st ACM workshop on IoT privacy, trust, and security*, p 1. <https://doi.org/10.1145/2732209.2732216>
- Kaur J, Wongthongtham P, Abu-Salih B, Fathy S (2018) Analysis of scientific production of IoT big data research. In: 2018 32nd international conference on advanced information networking and applications workshops (WAINA). IEEE, pp 715–720. <https://doi.org/10.1109/WAINA.2018.00173>
- Kim HK (2019) IoT network intrusion dataset. *Creative Commons Attribution, IEEE*. <https://creativecommons.org/licenses/by/4.0/>
- Kotsiantis SB, Zaharakis I, Pintelas P (2007) Supervised machine learning: a review of classification techniques. *Emerg Artif Intell Appl Comput Eng* 160(1):3–24
- Kumar V (2005) Parallel and distributed computing for cybersecurity. *IEEE Distrib Syst Online*. <https://doi.org/10.1109/MDSO.2005.53>
- Lai T, Ramos F (2021) Plannerflows: Learning motion samplers with normalising flows. In: 2021 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 2542–2548. <https://doi.org/10.1109/IROS51168.2021.9636190>
- Lee I, Lee K (2015) The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Bus Horiz* 58(4):431–440. <https://doi.org/10.1016/j.bushor.2015.03.008>
- Lu H, Jin C, Helu X, Du X, Guizani M, Tian Z (2021) Deepautod: research on distributed machine learning oriented scalable mobile communication security unpacking system. *IEEE Trans Netw Sci Eng* 9(4):2052–2065
- Mahesh B (2020) Machine learning algorithms-a review. *Int J Sci Res (IJSR)* 9:381–386
- Mahmoud R, Yousuf T, Aloul F, Zualkernan I (2015) Internet of things (IoT) security: current status, challenges and prospective measures. In: 2015 10th international conference for internet technology and secured transactions (ICITST). IEEE, pp 336–341. <https://doi.org/10.1109/ICITST.2015.7412116>
- Markou M, Singh S (2003) Novelty detection: a review-part 2: neural network based approaches. *Signal Process* 83(12):2499–2521
- Mijwil M, Salem IE, Ismaeel MM (2023) The significance of machine learning and deep learning techniques in cybersecurity: a comprehensive review. *Iraqi J Comput Sci Math* 4(1):87–101
- Myles AJ, Feudale RN, Liu Y, Woody NA, Brown SD (2004) An introduction to decision tree modeling. *J Chemometrics: J Chemometrics Soc* 18(6):275–285. <https://doi.org/10.1002/cem.873>
- Naimi AI, Balzer LB (2018) Stacked generalization: an introduction to super learning. *Eur J Epidemiol* 33(5):459–464. <https://doi.org/10.1007/s10654-018-0390-z>
- Neshenko N, Bou-Harb E, Crichigno J, Kaddoum G, Ghani N (2019) Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations. *IEEE Commun Surv Tutor* 21(3):2702–2733. <https://doi.org/10.1109/COMST.2019.2910750>
- Okano MT (2017) IOT and industry 4.0: the industrial new revolution. In: *International conference on management and information systems*, vol 25, p 26
- Pal M (2005) Random forest classifier for remote sensing classification. *Int J Remote Sens* 26(1):217–222. <https://doi.org/10.1080/01431160412331269698>
- Pisner DA, Schnyer DM (2020) Support vector machine. In: *Machine learning*. Elsevier, pp 101–121
- Riahi A, Challal Y, Natalizio E, Chtourou Z, Bouabdallah A (2013) A systemic approach for IoT security. In: 2013 IEEE international conference on distributed computing in sensor systems. IEEE, pp 351–355. <https://doi.org/10.1109/DCOSS.2013.78>
- Ruan J, Liang G, Zhao J, Zhao H, Qiu J, Wen F, Dong ZY (2023) Deep learning for cybersecurity in smart grids: review and perspectives. *Energy Convers Econ* 4(4):233–251

- Ruta D, Gabrys B (2005) Classifier selection for majority voting. *Inform Fusion* 6(1):63–81. <https://doi.org/10.1016/j.inffus.2004.04.008>
- Spence C, Parra L, Sajda P (2001) Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. In: *Proceedings IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA 2001)*, pp. 3–10. IEEE. <https://doi.org/10.1109/MMBIA.2001.991693>
- Vaccari I, Chiola G, Aiello M, Mongelli M, Cambiaso E (2020) MQTTset, a new dataset for machine learning techniques on MQTT. *Sensors* 20(22):6578. <https://doi.org/10.3390/s20226578>. (Publisher: MDPI)
- Wang X, Zhang H, Zhang Y, Wang M, Song J, Lai T, Khushi M (2021) Learning non-stationary time-series with dynamic pattern extractions. *IEEE Trans Artif Intell*. <https://doi.org/10.1109/TAI.2021.3130529>
- Wheelus C, Zhu X (2020) IoT network security: threats, risks, and a data-driven defense framework. *IoT* 1(2):259–285. <https://doi.org/10.3390/iot1020016>
- Xiao L, Wan X, Lu X, Zhang Y, Wu D (2018) IoT security techniques based on machine learning: How do IoT devices use AI to enhance security? *IEEE Signal Process Mag* 35(5):41–49. <https://doi.org/10.1109/MSP.2018.2825478>
- Zhang Z-K, Cho MCY, Wang C-W, Hsu C-W, Chen C-K, Shieh S (2014) IoT security: ongoing challenges and research opportunities. In: *2014 IEEE 7th international conference on service-oriented computing and applications*. IEEE, pp 230–234. <https://doi.org/10.1109/SOCA.2014.58>
- Zhang Y, Meratnia N, Havinga P (2010) Outlier detection techniques for wireless sensor networks: a survey. *IEEE Commun Surv Tutor* 12(2):159–170. <https://doi.org/10.1109/SURV.2010.021510.00088>
- Zhu R, Ji X, Yu D, Tan Z, Zhao L, Li J, Xia X (2020) KNN-based approximate outlier detection algorithm over IoT streaming data. *IEEE Access: Pract Innov Open Solut* 8:42749–42759. <https://doi.org/10.1109/ACCESS.2020.2977114>.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.