Intelligent Automation & Soft Computing
DOI:10.32604/iasc.2022.021879
Article

Tech Science Press

Machine Learning Approach for Improvement in Kitsune NID

Abdullah Alabdulatif[1] and Syed Sajjad Hussain Rizvi[2,*]

[1]Department of Computer, College of Sciences and Arts in Al-Rass, Qassim University, Al-Rass, Saudi Arabia
[2]Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Karachi, Pakistan
*Corresponding Author: Syed Sajjad Hussain Rizvi. Email: sshussainr@gmail.com
Received: 18 July 2021; Accepted: 26 August 2021

Abstract: Network intrusion detection is the pressing need of every communication network. Many network intrusion detection systems (NIDS) have been proposed in the literature to cater to this need. In recent literature, plug-and-play NIDS, Kitsune, was proposed in 2018 and greatly appreciated in the literature. The Kitsune datasets were divided into 70% training set and 30% testing set for machine learning algorithms. Our previous study referred that the variants of the Tree algorithms such as Simple Tree, Medium Tree, Coarse Tree, RUS Boosted, and Bagged Tree have reported similar effectiveness but with slight variation inefficiency. To further extend this investigation, we have explored the performance of variants of above said Tree algorithms on other datasets provided by Kitsune, such as Active Wiretap, ARP MitM, Fuzzing, OS Scan, SSDP Flood, SYN DoS, SSL renegotiation, Mirai, and Video Injection. This investigation ascertains the likely performance of above said tree algorithm variants. After a deep and rigorous analysis, the Fine Tree is highly recommended for the improved version of the Kitsune Tool.

Keywords: Kitsune; machine learning; active wiretap; ARP MitM; fuzzing; OS scan; SSDP flood; SYNDoS; SSL renegotiation; and video injection

1 Introduction

Data security is the pressing need of modern-day data communication. Mainly the security measures are established at the communication channel and/or at the system level. The communication channel security is ensured by encryption techniques [1] such as image encryption [2], data encryption [3], IoT encryption [4] etc. Likewise, the system-level security is equipped with a firewall [5], anti-spyware [6], antivirus [7], and network intrusion detection system(NIDS) [8–11]. Machine learning (ML) and deep learning (DL) based NIDS has gained major attention in the recent decade [12]. Technically they are called intelligent network intrusion detection systems (INIDS)such as, sensor network intrusion detection [13], SDN based network intrusion detection system [14,15], An anomaly-based network intrusion detection [16], In-vehicle network intrusion detection [17] etc.

The INIDS essentially requires a massive and credible dataset. In the recent literature, many NIDS datasets have been proposed. Some of them are publicly available to the research community. It includes,

but not limited to, UNSW-NB15 (2015) [18], NSL-KDD dataset (2019) [19], Kitsune (2019) [20], NSL-KDD dataset (2020) [21], CIDDS-001 dataset (2018) [22], LITNET (2020) [23], KDD CUP (99) [24] etc. It can be observed that during the last couple of years, many benchmark datasets have been proposed by the research community for INIDS. To the best of our comprehension, 'Kitsune' is observed as the benchmark among the dataset. It covers a wide variety of network attacks, it is massive in volume and comprehensive in contents, and is publicly available.

This study first presents a comprehensive view of the existing work on applying machine learning and deep learning algorithm on Kitsune as a literature survey. Second, we have bridged the research gap and present a parametric empirical comparison of machine learning algorithms to find the best candidate of a machine learning algorithm for Kitsune as simulation results. Finally, we have introduced a rationale for opting for the best machine learning algorithm for the improved version of Kitsune analysis.

2 Literature Review

This section presents a comprehensive literature review on the recent work on the Kitsune dataset, specifically applying machine learning and deep learning algorithm on it. In late 2018, Mirsky et al. has contributed a robust plug-and-play NIDS named Kitsune. The Kitsune can detect a large variety of network attacks without substantial supervision. The principal algorithm of Kitsune is KitNET, with an ensemble of artificial neural networks. This arrangement helps to detect the traces of abnormal traffic patterns from the burst of legitimate network traffic. The authors in this research also have presented a benchmark dataset for NIDS. This dataset comprises Active Wiretap, ARP MitM, Fuzzing, OS Scan, SSDP Flood, SYN DoS, SSL renegotiation, Mirai, and Video Injection. The dataset is massive in volume and rich in contents [20]. Fig. 1 illustrates the system overview of Kitsune, where first the network packet is fetched by packet capturer. After capturing, the packet is parsed into the units. These units are fed for feature extraction and mapping. Finally, the packet is labeled as Benign/Malicious. The dataset generated from this model is also facilities to train the machine learning and deep learning models.

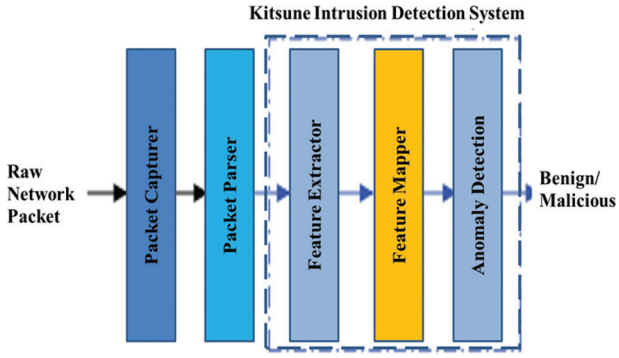

Figure 1: System overview of Kitsune

Peng et al. [25] in their study discussed that the classical Signature-based Network Intrusion Detection Systems are found to be deficient in handling the new disjoint network threats. Specifically, the thread with unknown signatures is significantly less susceptible to detection and tract. This opens the venue to employ machine learning for adaptive learning. The scenario looks beneficial. However, it is also a well-established fact in the literature that machine learning is also prone to adversarial attacks. Hashemi et al., in their study, has evaluated anomaly-based NIDS for test input. Specifically, they have trained the neural network model to handle adversarial information. They have opted for Kitsune to benchmark NIDS to train the network and test

it for adversarial attacks. The scope of this work is only limited to one machine learning algorithm. The investigation of other machine learning algorithms was not present in this study [25].

In the year 2019 Qiu et al. [26] have proposed a novel adversarial network attack to see if the deep learning-based IDS are equally prone to adversarial attacks. They have reported that with this adversarial attack, the accuracy of Kitsune is compromised. In this work, they have merely two types of attacked were targeted, i.e., Mirai Botnet attack and video injections. The scope of this research can further be extended if the investigation on another dataset would also be explored. Moreover, the proposed adversarial attack can be validated to other well-known variants of deep learning models [26].

In the same year, Hashemi et al. [27] seconded the vulnerability of anomaly-based NIDS based on the neural network due to the adversarial attacks. In addition, the author has highlighted that the center of the work was on the older version of the dataset that was not truly mimicking the variety of network attacks., The research work has proposed Reconstruction from Partial Observation (RePO) as a novel appliance to build a NIDS. Functionally, that it uses de-noising auto encoder for detecting different types of network attacks in a low false alert setting. It also defend adversarial example attack. They have opted Kitsune dataset to validate their approach. Later, Zhong et al. [28] also opted for the Kitsune dataset as the benchmark to validate their proposed work. The scope of their work was tri-folded, i.e., practical, generic, and explainable. The 'Practical refers that the suggested attack can repeatedly transform original traffic with minimal information and overhead, while keeping the functionality stable. Second, the generic refers to the proposed attack as operative for evaluating the robustness of various NIDS. Finally, explainable means to propose a reasoning method for the robustness of ML-based NIDSs.

The same researcher in [28], extended their work by proposing a novel anomaly detection framework. This proposed framework was the integration and hybridization of multiple deep learning architectures. This framework first use the Damped Incremental Statistics algorithm to extract features from organic network traffic. Subsequently, the auto encoder was trained with a small amount of labeled data. Likewise, the dataset with abnormal scores is fed to LSTM. Finally, the weightage ranking approach was used to determine the abnormal score. Again the benchmarking of the results for the said work was done on the Kitsune dataset. This study only targeted the Mirai botnet attack dataset and ignored the other available dataset of Kitsune like Active Wiretap, ARP MitM, Fuzzing, OS Scan, SSDP Flood, SYN DoS, SSL renegotiation, Mirai, and Video Injection [29]. The authors of [28,29] presented their allied work in 2020 in which they publisheda practical traffic-space evasion attack on learning-based NIDSs. Again similar to [28] the scope of their work is tri-folded, i.e., practical, generic, and explainable. The 'practical' refers to providing a new framework to mutate malicious traffic with extremely limited information while keeping the functionality stable. The generic refers that the proposed attack was effective for any ML classifiers and non-payload-based features. Finally, 'explainable' means proposing a feature-based interpretation method to measure the robustness of targeted systems against such attacks [30].

Another group of researchers in [31] has presented a comparative analysis of FGSM, JSMA, C&W, and ENM over the Kitsune dataset in the same year [31]. Bai et al. Set up a new approach called FastFE. This high-speed function extractor affects the ability of next-generation programmable switches to deliver the desired traffic functions flexibly and efficiently. The authors have demonstrated that the advancement of FastFE and its low overheads over the Kitsune dataset [32]. Leevy et al. [33] have presented a new IDS called AE-IDF, inspired by Kitsune. However, both differentiate in feature selection in feature mapping. Kitsune used the damping window and dynamic feature retrieval, while AE-IDS used Random Forest to determine optimal functionality. In addition, Kitsune maps 'n' features to 'k' small subset using agglomerated hierarchical clustering; conversely, AE-IDS uses AP clustering to group features according to the degree of similarity.

Recently Wang et al. [34] have presented a comparative study of three different machine learning classifiers namely decision tree, random forest, and logistic regression to classify attack traffic using Kitsune dataset. The performance was measured in accuracy, attack detection rate (ADR), false alar3.

3 Gap Analysis

The analysis of the above literature inferred the following research gaps:

1. Many researchers have advocated that the signature-based NIDS are not capable of handling a new type of attack. Whereas the anomaly-based NIDS use machine learning to handle this issue, but at the same time, machine learning-based NIDS are very much prone to adversarial attacks.
2. Many of the researchers have opted Kitsune to validate only their approach in the limited scope in terms of dataset and algorithms.
3. Kitsune being the machine learning-based NIDS, to the best of the knowledge, no researcher has presented a comprehensive comparative analysis of machine learning algorithm on all dataset of Kitsune.
4. Many of the researchers have only investeded the Mirai botnet dataset of Kitsune, very few have investigated the some of the dataset of Kitsune, including OS Scan, Fuzzing, Video Injection, ARP MitM, Active Wiretap, SSDP Flood, SYN DoS, SSL, and Renegotiation.

Refereeing research gaps 3 and 4, the dataset provided by Kitsune was used for the training and testing of the machine learning algorithm. Our previous study reffered that the variants of tree algorithms such as Simple Tree, Medium Tree, Coarse Tree, RUS Boosted, and Bagged Tree have reported similar effectiveness but with slight variation inefficiency. To further extend this investigation, we have explored the performance of variants of above said Tree algorithms on other datasets provided by Kitsune, such as Active Wiretap, ARP MitM, Fuzzing, OS Scan, SSDP Flood, SYN DoS, SSL renegotiation, Mirai, and Video Injection. This investigation ascertains the likely performance of above said tree algorithm variants. After a deep and rigorous investigation, the Fine Tree is highly recommended for the improved version of the Kitsune Tool.

4 Simulation Setup and Procedure

The proposed study was initiated from the benchmark dataset of Kitsune Network Attack Dataset, publically available on UCI Machine Learning repository. The attributes of the dataset are comprehended and analyzed using the dataset description. Afterwards each dataset was divided into 70% training and 30% testing samples, respectively with random permutation arrangement. The dataset is then converted from .CVS to .mat for Matlab. The simulation setup were established on a high performance computing machine preloaded with Matlab 2020. Following are the specifications of the high performance machine:

- Processor: Intel (R) Xeon (R) CPU E5-2673 v3 @ 2.40 GHz
- RAM 64GB
- GPU 04

The testing of algorithm were established on the Classification Learner App of Matlab 2020, where the performance of each algorithm as a function of confusion matrix, TPR, FNR, average training accuracy, test accuracy, misclassification cost, prediction time, and training time. After training each model were exported as a setup of corresponding .mat file. The ‘predict function’ of each exported model is used to evaluate the test accuracy using testing samples of dataset.

5 Simulation Results

This section present the investigation is performed on eight datasets of Kitsune, namely Active Wiretap, ARP MitM, Fuzzing, OS Scan, SSDP Flood, SYN DoS, SSL renegotiation, Mirai, and Video Injection. The investigation on the Mirai dataset was accomplished in our previous work [35]. Each dataset is divided into 70% training sample and 30% disjoint testing sample in random permutation selection of training and testing sample.

Active wiretapping [36] indicates adding false signals or tampering with communications or devices. This could be established on both guided and unguided media. In Kitsune Active Wiretap dataset have 1595082 instances of training sample and 683607 testing samples. In all the datasets of Kitsune, there are 115 input attributes and one output attributed. Primarily it is a binary classification domain where '0' represents 'No Attack' and '1' refers to the occurrence of "Attack".

A spoofing ARP [37], also known as ARP poisoning, is a middle man (MitM) attack. It allows attackers to intercept communication between and network devices. In Kitsune ARP MitM dataset have 1752987 instances of training sample and 751280 testing samples. Fuzz testing (fuzzing) is a quality assurance technique used to detect encryption errors and security vulnerabilities in software, operating systems or networks [38]. This involves capturing massive quantities of random data, called fuzz, about testing in an attempt to plant it. In Kitsune, it has 1752987 instances of training sample and 751280 samples of testing samples. The OS scan works by using the TCP/IP stack fingerprinting method [39]. Service analytics works by using the N map-service-probes database to identify services performed on a targeted host. In Kitsune, it has 1188496 instances of training sample and 509355 samples of testing samples.

A Simple Service Discovery Protocol (SSDP) [40] attack is a reflection-based distributed denial-of-service (DDoS) attack. It uses Universal Plug and Play (UPnP) networking protocols to send an amplified amount of traffic to a targeted victim. In Kitsune, it has 2854086 instances of training sample and 1223180 samples of testing samples. SSL renegotiation messages (including ciphers and encryption keys) are encrypted and then sent over the existing SSL connection. In Kitsune, it has 1545300 instances of training sample and 662271 testing samples. An SYN flood is a form of DoS attack in which an attacker sends a succession of SYN requests to a target's system [41]. It made an attempt to guzzle enough server resources to make the system unresponsive to authentic traffic. In Kitsune, it has 1939893 instances of training sample and 831383 samples of testing samples.

Tabs. 1–8 illustrate the comprehensive empirical evaluation of eight Kitsune datasets. It can be observed from these tables that all the variants of Tree algorithms are yielding approximately 100% TPR and FNR. The class-wise accuracy is evident from the confusion matrix of each algorithm for each dataset. Therefore, the net training accuracy for almost every variant of the Tree Algorithm in this study seems to be identical. Likewise, the misclassification pattern also seems to be very consistent as for Active Wiretap, ARP MitM, Video Injection and Coarse Tree which reported with the maximum misclassification cost with these attacks. However, Boosted Tree is reported worst with maximum misclassification cost for Fuzzing, OS Scan, SSDP Flood, SSL Renegotiation, and SYN Dos. It has been observed that for every dataset, Bagged Tree is reported with no misclassification cost. However, the prediction speed is on average 9.5 time less than Fine Tree. The fine tree shows less than 1% compromise on test accuracy and misclassification cost.

Similarly, the training time of Bagged Tree is approximately fouretime to the training time of Fine Tree and Medium Tree. It is inferred that the Fine Tree and Medium Tree are the best optimization tool for unified network attack detection on Kitsune. It is highly recommended that the improved version of Kitsune use either Fine Tree or Medium Tree as universal classifiers. Due to the identical and close behavior of training accuracy, misclassification cost, prediction speed, testing accuracy, only the pictorial illustration

of performance measures on Active Wiretap is illustrated in Figs. 2–5. Given that, the amplitude of variation can be derive from the Tabs. 1–8.

Table 1: Performance matrix on active wiretap

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs./s)	Train Time (s)
			Predicted class								
			0	1							
Fine tree	True class	0	100	0	100	0	100	99.99	1	1100000	461
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	19	1000000	463
		1	0	100	0	100					
Coarse tree	True class	0	100	0	100	0	99.5	99.81	3126	1100000	312
		1	1	99	1	99					
Boosted tree	True class	0	100	0	100	0	100	99.99	36	870000	1087
		1	0	100	0	100					
Bagged tree	True class	0	100	0	100	0	100	100	0	130000	1297
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	75	130000	5469
		1	0	100	0	100					

Table 2: Performance matrix on ARP MitM

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs./s)	Train time (s)
			Predicted class								
			0	1							
Fine tree	True class	0	100	0	100	0	100	99.99	13	1300000	567
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	34	1400000	578
		1	0	100	0	100					
Coarse tree	True class	0	100	1	100	1	99.9	99.91	1388	1400000	412
		1	0	99	0	99					
Boosted tree	True class	0	100	0	100	0	100	99.99	36	870000	1087
		1	0	100	0	100					
Bagged tree	True class	0	100	0	100	0	100	99.99	0	130000	7480
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	31	160000	8310
		1	0	100	0	100					

Table 3: Performance matrix on fuzzing

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs. /s)	Train time (s)
			Predicted class								
			0	1							
Fine tree	True class	0	100	0	100	0	100	99.99	2	890000	363
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	2	1200000	383
		1	0	100	0	100					
Coarse tree	True class	0	100	0	100	0	99.9	99.99	59	760000	330
		1	1	99	1	99					
Boosted tree	True class	0	100	0	100	0	80.7	80.71	302954	1700000	451
		1	20	80	20	80					
Bagged tree	True class	0	100	0	100	0	100	99.99	0	130000	1274
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	0	280000	780
		1	0	100	0	100					

Table 4: Performance matrix on OS scan

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs. /s)	Train time (s)
			Predicted class								
			0	1							
Fine tree	True class	0	100	0	100	0	100	99.99	1	810000	229
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	1	670000	238
		1	0	100	0	100					
Coarse tree	True class	0	100	0	100	0	100	99.99	8	1000000	194
		1	0	100	0	100					
Boosted tree	True class	0	99	1	99	1	96.1	96.11	45933	1800000	216
		1	1	95	1	95					
Bagged tree	True class	0	100	0	100	0	100	99.99	0	140000	471
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	0	270000	144
		1	0	100	0	100					

Table 5: Performance matrix on SSDP flood

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs./s)	Train time (s)
			Predicted class								
			0	1							
Fine tree	True class	0	100	0	100	0	100	99.99	1	1100000	1173
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	1	1200000	1232
		1	0	100	0	100					
Coarse tree	True class	0	100	0	100	0	100	99.99	3	890000	1180
		1	0	100	0	100					
Boosted tree	True class	0	80	20	80	20	65	64.61	1006828	1100000	1216
		1	55	45	55	45					
Bagged tree	True class	0	100	0	100	0	100	99.99	0	150000	1967
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	0	380000	1242
		1	0	100	0	100					

Table 6: Performance matrix on SSL renegotiation

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs./s)	Train time (s)
			Predicted class								
			0	1							
Fine Tree	True class	0	100	0	100	0	100	99.99	7	10000000	616
		1	0	100	0	100					
Medium Tree	True class	0	100	0	100	0	100	99.99	42	1300000	586
		1	0	100	0	100					
Coarse Tree	True class	0	100	0	100	0	99.8	99.74	112	800000	350
		1	1	99	1	99					
Boosted Tree	True class	0	80	20	80	20	65	99.99	3820	200000	5082
		1	20	45	20	45					
Bagged Tree	True class	0	100	0	100	0	100	99.99	0	140000	1121
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	0	230000	296
		1	0	100	0	100					

Table 7: Performance matrix on SYN Dos

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs./s)	Train time (s)
			Predicted class								
			0	1							
Fine tree	True class	0	100	0	100	0	100	99.99	3	740000	548
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	3	520000	561
		1	0	100	0	100					
Coarse tree	True class	0	100	0	100	0	99.8	99.99	52	380000	436
		1	1	99	1	99					
Boosted tree	True class	0	80	20	80	20	65	99.74	4878	720000	567
		1	20	45	20	45					
Bagged tree	True class	0	100	0	100	0	100	99.99	0	140000	1241
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	1	190000	256
		1	0	100	0	100					

Table 8: Performance matrix on video injection

Algorithm	Confusion matrix (%)				TPR (TPR) (%)	False negative rate (FNR) (%)	Net accuracy (%)	Test accuracy (%)	Total misclassification cost	Prediction speed (Obs/s)	Train time (s)
			Predicted class								
			0	1							
Fine tree	True Class	0	100	0	100	0	100	99.99	1	1300000	639
		1	0	100	0	100					
Medium tree	True class	0	100	0	100	0	100	99.99	2	960000	683
		1	0	100	0	100					
Coarse tree	True class	0	100	0	100	0	99.5	99.5	8410	840000	370
		1	1	99	1	99					
Boosted tree	True class	0	100	0	100	0	100	99.99	0	420000	2630
		1	0	100	0	100					
Bagged tree	True class	0	100	0	100	0	100	99.99	1	120000	1558
		1	0	100	0	100					
RUS boosted tree	True class	0	100	0	100	0	100	99.99	100	100	100
		1	0	100	0	100					

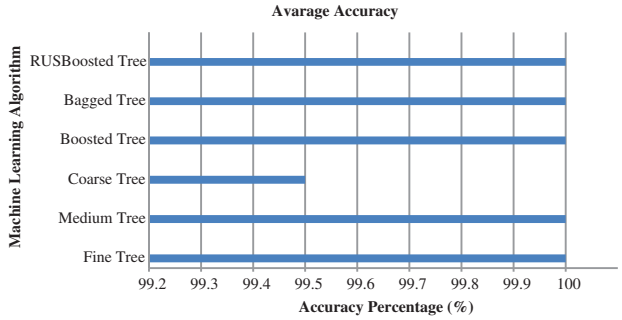

Figure 2: Avarage training accuracy on active wiretap

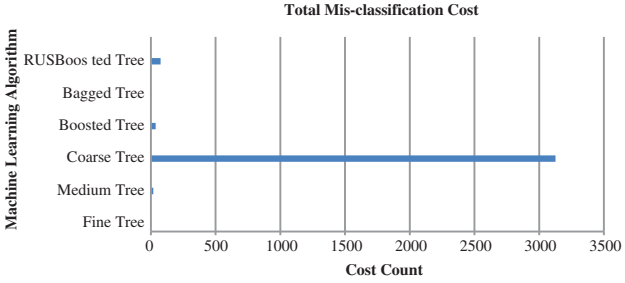

Figure 3: Total misclassification cost on active wiretap

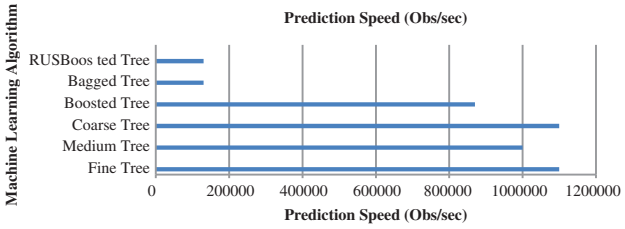

Figure 4: Prediction speed on active wiretap

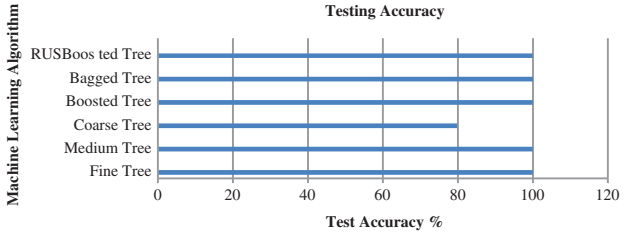

Figure 5: Testing accuracy

Fig. 3 illustrate the total misclassification cost on Active Wiretap for the set of algorithms. In this figure, x-axis represent the total cost count and the y-axis illustrate the algorithm. It is inferred from the figure that Coarse Tree has reported to have the significantly high misclassification cost as compare to the other varients of machine learning algorithm. This advocate strongly against the Coarse Tree for it utility in real-time network attack detection.

Fig. 4 illustrate the Prediction Speed cost of Active Wiretap for the set of algorithms. In this figure, x-axis represent Prediction Speed (Obs/s) and the y-axis illustrate the algorithm. It can be inferred that of the then RUS Boosted, and Bagged Tree the rest of the algorithm shows significant improved prediction speed. Therefore, The Boosted Tree, Coarse Tree, Medium Tree, and Fine Tree wins the race in terms of prediction speed.

Fig. 5 illustrate the Test Accuracy cost of Active Wiretap for the set of algorithms. In this figure, x-axis represent test accuracy and the y-axis illustrate the algorithm. It is inferred from this figure that other then Coarse Tree the rest of the algorithm are found to be efficient in term of testing accuracy.

6 Analysis

The above parametric assessment and evaluation established the response of machine learning algorithms for network intrusion detection in the adversarial nature of test inputs. It is very important to notice that the variants of tree algorithms have submitted the competitive performance during the training process. This can be evident from the confusion matrix of each assessment. Likewise the training accuracy also advocate for the same fact. However, the abrupt performance degradation of testing accuracy and mis-classification cost has been noted. It is mainly due to the adversarial nature of data. Given that the dataset have binary classes, the class level accuracy of a certain algorithm badly hit by adversarial attack.

At the same time the computational efficiency of each algorithm are also recorded to be in close vicinity. It is primarily due to the fact that the adversarial attack in the testing dataset do not notably affect the computation cost. Rather, it badly hit the class level accuracy of the testing results.

It is very hard to limit the adversarial traces in the busty traffic such as the network traffic. The preventive measure to detect or mitigate the adversarial traces with no only case to compromise the network efficiency but also found to be nearly impractical for the real time communication such as network traffic, video surveillance etc. This study help to figure out the most suitable machine learning algorithm for Kitsune that is not very much prone to the adversarial attack on in the network intrusion dataset. It is also important to note that this finding may vary for other dataset due to structural difference of application and dataset. Kitsune being an important network intrusion detection have a pressing need to have empirical justification to opt a machine learning algorithm. Given that it should not create the adverse effect on the computation efficiency. After the rigorous simulation and parametric evaluation it is concluded that the Fine Tree is found to be the most optimum machine learning for the improved version of Kitsune.

7 Conclusion

Kitsune being the machine learning-based NIDS, to the best of the knowledge, no researcher has presented a comprehensive comparative analysis of machine learning algorithm on Kitsune. Many researchers have only investigated the Mirai botnet dataset of Kitsune, and None has investigated the complete dataset of Kitsune, including Active Wiretap, ARP MitM, Fuzzing, OS Scan, SSDP Flood, SYN DoS, SSL renegotiation, Mirai, and Video Injection. Our previous study refers that the variants of tree algorithms such as Simple Tree, Medium Tree, Coarse Tree, RUS Boosted, and Bagged Tree have reported similar effectiveness but with slight variation inefficiency. To further extend this investigation, we have explored the performance of variants of above said tree algorithms on other datasets provided by Kitsune, such as OS Scan, Fuzzing, Video Injection, ARP MitM, Active Wiretap, SSDP Flood, SYN DoS, SSL, and Renegotiation. This study help to figure out the most suitable machine learning algorithm for Kitsune that is not very much prone to the adversarial attack on in the network intrusion dataset. It is

also important to note that this finding may vary for other dataset due to structural difference of application and dataset. This investigation ascertains the likely performance of above said tree algorithm variants. After a deep and rigorous investigation, the Fine Tree is highly recommended for the improved version of the Kitsune Tool.

Acknowledgement: The researchers would like to thank the Deanship of Scientific Research, Qassim University for and Shaheed Zulfikar Ali Bhutto Institute of Science and Technology for funding and support in the publication of this project.

Funding Statement: The researchers would like to thank the Deanship of Scientific Research, Qassim University for and Shaheed Zulfikar Ali Bhutto Institute of Science and Technology for funding and support in the publication of this project.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Atoev, O. Kwon, C. Kim and S. Lee, "The secure UAV communication link based on OTP encryption technique," in *Proc. 2019 Eleventh Int. Conf. on Ubiquitous and Future Networks*, Zagreb, Croatia, pp. 1–3, 2019.

[2] K. Shankar, "An optimal RSA encryption algorithm for secret images," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 20, pp. 2491–2500, 2018.

[3] K. Kapusta, H. Qiu and G. Memmi, "Secure data sharing by means of fragmentation, encryption, and dispersion," in *Proc. IEEE INFOCOM 2019-IEEE Conf. on Computer Communications Workshops*, Paris, France, pp. 1051–1052, 2019.

[4] Hussain, M. C. Negi and N. Pandey, "Proposing an encryption/decryption scheme for IoT communications using binary-bit sequence and multistage encryption," in *Proc. 2018 7th Int. Conf. on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)*, Noida, India, pp. 709–713, 2018.

[5] K. Neupane, R. Haddad and L. Chen, "Next generation firewall for network security: A survey," in *Proc. SoutheastCon 2018*, St. Petersburg, FL, USA, pp. 1–6, 2018.

[6] K. N. Mallikarajunan, S. R. Preethi, S. Selvalakshmi and N. Nithish, "Detection of spyware in software using virtual environment," in *Proc. 2019 3rd Int. Conf. on Trends in Electronics and Informatics*, Tirunelveli, India, pp. 1138–1142, 2019.

[7] F. A. Garba, K. I. Kunya, I. Kabiru, S. A. Ibrahim, A. B. Isa *et al.,* "Evaluating the state of the art antivirus evasion tools on windows and android platform," in *Proc. 2019 2nd Int. Conf. of the IEEE Nigeria Computer Chapter*, Zaria, Nigeria, pp. 1–4, 2019.

[8] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

[9] S. Otoum, B. Kantarci and H. T. Mouftah, "On the feasibility of deep learning in sensor network intrusion detection," *IEEE Networking Letters*, vol. 1, no. 2, pp. 68–71, 2019.

[10] N. Sultana, N. Chilamkurti, W. Peng and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 493–501, 2019.

[11] J. Kim, H. Kim, M. Shim and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics*, vol. 9, no. 6, pp. 916, 2020.

[12] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *Proc. 2016 8th IEEE Int. Conf. on Communication Software and Networks*, Beijing, China, pp. 581–585, 2016.

[13] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, pp. e4150, 2021.

[14] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE transactions on emerging topics in computational intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[15] S. Tiwari, V. Pandita, S. Sharma, V. Dhande and S. Bendale, "Survey on SDN based network intrusion detection system using machine learning framework," *International Research Journal of Engineering and Technology*, vol. 6, no. 12, pp. 493–501, 2019.

[16] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Proc. 2016 Int. Conf. on Wireless Networks and Mobile Communications*, Fez, Morocco, pp. 258–263, 2016.

[17] W. Wu, R. Li, G. Xie, J. An, Y. Bai *et al.,* "A survey of intrusion detection for in-vehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 919–933, 2020.

[18] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. 2015 Military Communications and Information Systems Conference*, Canberra, ACT, Australia, pp. 1–6, 2015.

[19] S. Gurung, M. K. Ghose and A. Subedi, "Deep learning approach on network intrusion detection system using NSL-KDD dataset," *International Journal of Computer Network and Information Security*, vol. 11, no. 3, pp. 8–14, 2019.

[20] Y. Mirsky, T. Doitshman, Y. Elovici and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," *Network and Distributed System Security Symposium*, 2018.

[21] Y. Ding and Y. Zhai, "Intrusion detection system for NSL-KDD dataset using convolutional neural networks," in *Proc. 2018 2nd Int. Conf. on Computer Science and Artificial Intelligence*, New York, NY, United States, pp. 81–85, 2020.

[22] A. Verma and V. Ranga, "Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning," in *Proc. 2017 6th Int. Conf. on Smart Computing and Communications*, Kurukshetra, India, pp. 709–716, 2018.

[23] R. Damasevicius, A. Venckauskas, S. Grigalunas, J. Toldinas and N. Morkevicius, "LITNET-2020: An annotated real-world network flow dataset for network intrusion detection," *Electronics*, vol. 9, no. 5, pp. 800, 2020.

[24] A. Divekar, M. Parekh, V. Savla, R. Mishra and M. Shirole, "Benchmarking datasets for anomaly-based network intrusion detection: KDD CUP 99 alternatives," in *Proc. 2018 IEEE 3rd Int. Conf. on Computing, Communication and Security*, Kathmandu, Nepal, pp. 1–8, 2018.

[25] X. Peng, W. Huang and Z. Shi, "Adversarial attack against dos intrusion detection: An improved boundary-based method," in *Proc. 2019 IEEE 31st Int. Conf. on Tools with Artificial Intelligence*, Portland, OR, USA, pp. 1288–1295, 2019.

[26] H. Qiu, T. Dong, T. Zhang, J. Lu, G. Memmi *et al.,* "Adversarial attacks against network intrusion detection in IoT systems," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10327–10335, 2020.

[27] M. J. Hashemi and E. Keller, "Enhancing robustness against adversarial examples in network intrusion detection systems," in *Proc. 2020 IEEE Conf. on Network Function Virtualization and Software Defined Networks*, Leganes, Spain, pp. 37–43, 2020.

[28] Y. Zhong, Y. Zhu, Z. Wang, X. Yin and X. Shi, "An adversarial learning model for intrusion detection in real complex network environments," in *Proc. 2020 15th Int. Conf. on Wireless Algorithms, Systems, and Applications*, Qingdao, China, pp. 794–806, 2020.

[29] S. Baek, D. Kwon, S. C. Suh, H. Kim and I. Kim, "Clustering-based label estimation for network anomaly detection," *Digital Communications and Networks*, vol. 7, no. 1, pp. 37–44, 2021.

[30] D. Han, Z. Wang, Y. Zhong, W. Chen and J. Yang, "Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2632–2647, 2021.

[31] M. A. Khan, "CRNNIDS: Hybrid convolutional recurrent neural network-based network intrusion detection system," *Processes*, vol. 9, no. 5, pp. 834, 2021.

[32] J. Bai, M. Zhang, G. Li, C. Liu, M. Xu *et al.,* "FASTFE: Accelerating ml-based traffic analysis with programmable switches," in *Proc. 2020 of the Workshop on Secure Programmable Network Infrastructure, Virtual Event USA*, Virtual Event USA, pp. 1–7, 2020.

[33] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 big data," *Journal of Big Data*, vol. 7, no. 1, pp. 1–19, 2020.

[34] X. Wang, S. Bagui and S. Bagui, "Machine learning in spark for attack traffic classification in IoT devices using protocol usage statistics," in *Proc. 2020 Int. Conf. on Innovations in Information and Communication Technologies*, Singapore, pp. 1–11, 2020.

[35] A. Alabdulatif, S. S. H. Rizvi and M. A. Hashmi, "Optimal machine learning models for kitsune to detect mirai botnet malware attack," *Journal of Hunan University Natural Sciences*, vol. 48, pp. 91–102, 2021.

[36] P. Psathas, L. Iliadis and A. Papaleonidas, "A hybrid deep learning ensemble for cyber intrusion detection," in *Proc. 2021 Int. Conf. on Engineering Applications of Neural Networks, Porto Carras Grand Resort*, Halkidiki, Greece, pp. 27–41, 2021.

[37] M. Lin, B. Zhao and Q. Xin, "ERID: A deep learning-based approach towards efficient real-time intrusion detection for IoT," in *Proc. 2020 IEEE Eighth Int. Conf. on Communications and Networking*, Hammamet, Tunisia, pp. 1–7, 2020.

[38] Y. Chen, C. M. Poskitt, J. Sun, S. Adepu and F. Zhang, "Learning-guided network fuzzing for testing cyber-physical system defences," in *Proc. 2019 34th IEEE/ACM Int. Conf. on Automated Software Engineering (ASE)*, San Diego, CA, USA, pp. 962–973, 2019.

[39] M. López-Vizcaíno, F. J. Novoa, D. Fernández and V. Carneiro, "Early intrusion detection for OS scan attacks," in *Proc. 2019 IEEE 18th Int. Symp. on Network Computing and Applications*, Cambridge, MA, USA, pp. 1–5, 2019.

[40] F. Gharibian and A. A. Ghorbani, "Comparative study of supervised machine learning techniques for intrusion detection," in *Proc. Fifth Annual Conference on Communication Networks and Services Research*, Fredericton, NB, Canada, pp. 350–358, 2007.

[41] V. Morfino and S. Rampone, "Towards near-real-time intrusion detection for IoT devices using supervised learning and apache spark," *Electronic*, vol. 9, no. 3, pp. 444, 2020.