Adversarial Machine Learning: A Multilayer Review of the State-of-the-Art and Challenges for Wireless and Mobile Systems

Jinxin Liu, *Student Member, IEEE*, Michele Nogueira, *Senior Member, IEEE*, Johan Fernandes, and Burak Kantarci, *Senior Member, IEEE*

***Abstract*—Machine Learning (ML) models are susceptible to adversarial samples that appear as normal samples but have some imperceptible noise added to them with the intention of misleading a trained classifier and misclassifying the input. Adversarial Machine Learning (AML) was initially coined following upon researchers pointing out certain blind spots in image classifiers in computer vision field which were exploited by these adversarial samples to deceive the model. Although this has been investigated remarkably in computer vision, the impact of AML in wireless and mobile systems has recently attracted attention. Wireless and mobile networks have intensely benefited from the application of ML classifiers to detect network traffic anomalies and malware detection. However, ML detectors themselves can be exfiltrated/evaded by the samples carefully designed by attackers, raising security concerns for ML-based network applications. Thus, it is crucial to detect such samples to safeguard the network. This survey article presents a systematic mapping and a comprehensive literature review on AML to wireless and mobile systems from physical layer to network and application layers. The article reviews the state-of-the-art AML approaches in the generation and detection of adversarial samples. The samples can be generated by adversarial models such as Generative Adversarial Networks (GANS) and techniques such as Fast Gradient Sign Method (FGSM). The samples can be detected by adversarial models acting as classifiers or ML classifiers reinforced with knowledge on how to detect such samples. For each approach, a high-level overview is provided alongside its impact on solving the problems in wireless and mobile settings. Furthermore, this article provides detailed discussions to highlight the open issues and challenges faced by these approaches, as well as research opportunities which can be of interest to the researchers and developers in Artificial Intelligence (AI)-driven wireless and mobile networking.**

***Index Terms*—Wireless networks, mobile networks, adversarial machine learning, artificial intelligence, intrusion detection.**

Manuscript received May 14, 2021; revised November 1, 2021; accepted December 14, 2021. Date of publication December 16, 2021; date of current version February 24, 2022. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) DISCOVERY Program under Grant RGPIN/2017-04032; in part by the NSERC and Ontario Centre for Innovation VIP Alliance Program under Grant ALLRP 561676–21; in part by the National Council for Scientific and Technological Development (CNPq), Brazil, under Grant 432204/2018-0; and in part by Fundação de Apoio á Pesquisa do Estado de São Paulo (FAPESP), Brazil, under Grant 18/23098-0. *(Corresponding author: Burak Kantarci.)*

Jinxin Liu, Johan Fernandes, and Burak Kantarci are with the School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON K1N 6N5, Canada (e-mail: jliu367@uottawa.ca; jfern090@uottawa.ca; burak.kantarci@uottawa.ca).

Michele Nogueira is with the Department of Computer Science, Federal University of Minas Gerais, Belo Horizonte 31270-901, Brazil (e-mail: michele@dcc.ufmg.br).

Digital Object Identifier 10.1109/COMST.2021.3136132

NOMENCLATURE

AC	Attack Confidence
ACK	Acknowledge Signal
ADR	Adversarial Detection Rate
AE	Auto-Encoder
AKA	Authentication and Key Agreement
AM−DSB	Amplitude Modulation Double Side-Band
AM−SSB	Amplitude Modulation Single Side-Band
AMC	Adaptive Modulation Classification
AML	Adversarial Machine Learning
APK	Android Application Package
AS	Attack Severity
ASR	Adversarial Attack Success Rate
BER	Bit Error Rate
BFAM	Brute Force Attack Method
BIM	Basic Iteration Method
BL−AMD	Broad Learning Android Malware Detector
BL−TDA	Broad Learning Training and Detection Algorithm
BPSK	Binary Phase Shift Keying
C&W	Carlini and Wagner
CFG	Control Flow Graphs
CIDS	Collaborative Intrusion Detection System
CNN	Convolutional Neural Network
CPFSK	Continuous-Phase Frequency-Shift Keying
CPS	Cyber-Physical System
DAC	Digital to Analog Converter
DAE	Denoising Autoencoder
DL	Deep Learning
DNN	Deep Neural Network
DoS	Denial of Service
DSA	Dynamic Spectrum Access
DT	Decision Tree
EIR	Evasion Increase Rate
FGM	Fast Gradient Method
FGSM	Fast Generative Sign Method
FNR	False Negative Rate
FPGA	Field-Programmable Gate Array
FPR	False Positive Rate
GAN	Generative Adversarial Network
GEA	Graph Embedding and Augmentation
GFSK	Gaussian Frequency-Shift Keying
HIDS	Host-based Intrusion Detection System
IDS	Intrusion Detection System

IL−TDA	Incremental Learning Training and Detection Algorithm
IoT	Internet of Things
IQ	In-phase and Quadrature
JSMA	Jacobian-based Saliency Map Attack
KNN	K-Nearest Neighbors
L−BFGS	Limited-Memory Broyden-Fletcher-Goldfarb-Shanno algorithm
LightGBM	Light Gradient Boosting Machine
LR	Logistic Regression
LSTM	Long Short Term Memory
MCC	Matthews Correlation Coefficient
MCTS	Monte Carlo Tree Search
MD	Malware Detection
MDP	Markov Decision Process
MIM	Momentum Iterative Method
MIMO	Multiple Input Multiple Output
ML	Machine Learning
ML−IDS	Machine Learning-based IDS
MLP	Multi-Layer Perceptron
MMD	Mobile Malware Detection
MMSE	Minimum Mean Square Error
MRPP	Maximum Received Perturbation Power
MSE	Mean Square Loss
NB	Nave Bayes
NIDS	Network-based Intrusion Detection System
NLP	Natural Language Processing
NTE	Noise Tolerance Estimation
ODR	Original Detection Rate
PA	Power Amplifier
PAM	Pulse-Amplitude Modulation
PCA	Principal Component Analysis
PGD	Projected Gradient Descent
PLA	Physical Layer Authentication
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
R2L	Remote to Local
RBF	Radial Basis Function
RF	Random Forest
RFN	Random Feature Nullification
RL	Reinforcement Learning
SA	Singnal Authentication
SIDS	Substitute Intrusion Detection System
SMOTE	Synthetic Minority Oversampling Technique
SNN	Spiking Neural Networks
SVM	Support Vector Machine
TNR	True Negative Rate
TPR	True Positive Rate
TTC	Total Time Cost
U2R	User to Root
VAE	Variational Auto-Encoder
VAM	Virtual Adversarial Method
WBFM	WideBand Frequency Modulation
WGAN	Wasserstein Generative Adversarial Network
WGAN−GP	Wasserstein Generative Adversarial Network with Gradient Penalty
WLAN	Wireless Local Area Network
WMN	Wireless and Mobile Network
WSN	Wireless Sensor Network
XGBoost	Extreme Gradient Boosting
ZOO	Zeroth Order Optimization.

I. Introduction

MACHINE Learning (ML) algorithms recognize patterns in data. Unlike conventional rule-based algorithms, ML algorithms are scalable and heuristic in nature [1]. ML algorithms learn remarkable patterns in data without being provided hand crafted rules beforehand as it is the case with rule-based algorithms. The ML algorithms can be used for classification [2] and regression [3] tasks. Irrespective of the task, conventional ML algorithms require the features of the input to be engineered in a format that could be processed by the ML algorithms. For instance, if the input is an image, the features would be the pixel intensity of the image. For a black and white image, each pixel will have a value in the range $[0 - 255]$. For a color image, each pixel will have three values each within that range representing RGB values.

Using Deep Learning (DL), which is a sub-field of ML, one could automatically extract features without human intervention through the use of different types of neural networks [4]. Thus, tasks, such as intrusion detection in wireless network [5], are now achievable without feature engineering or manually extracting the features before providing them to the DL model. This has unquestionably led to advancements in DL and its promising applications in multiple fields, such as computer vision [6], Natural Language Processing (NLP) [7] and wireless and mobile networks [8]. As another sub-field of ML, Adversarial Machine Learning (AML) studies the effect of adversarial samples/input which attack classification/detection systems by misleading them into classifying an adversarial sample under a different class than the original sample class [9]. These classification systems could be based on DL or the conventional ML models [10]–[13]. Independent of the type of the classifier, adversarial samples can be injected by perturbations to the input which is perceptually indistinguishable from a normal input [9]. Such perturbations mislead the classification models so as to have them predict an undesirable output [14].

Taking Fast Gradient Sign Method (FGSM) [15] as an example, which produces adversarial images to deceive a trained image classifier, the misleading classifier predicts labels that differ from the expectations. Such images should ideally be classified under their original class. Adversarial models, such as Generative Adversarial Networks (GANs) [16], and adversarial techniques, such as FGSM [15], exploit the feature extraction capabilities of ML/DL models by adding some perturbations in the features that lead the model to misclassify the image under a different class than its original class. Adversarial samples can be generated through different techniques [15]–[18] and adversarial models [16], [19], [20]. Szegedy *et al.* [17] were the first to notice certain blind spots in DL for image classifiers that were exploited by adversarial samples generated by the L-BFGS [17] sample generation technique. Papernot *et al.* [21] have further proven that adversarial samples, generated for a certain image classifier, can be used to undermine the performance of other

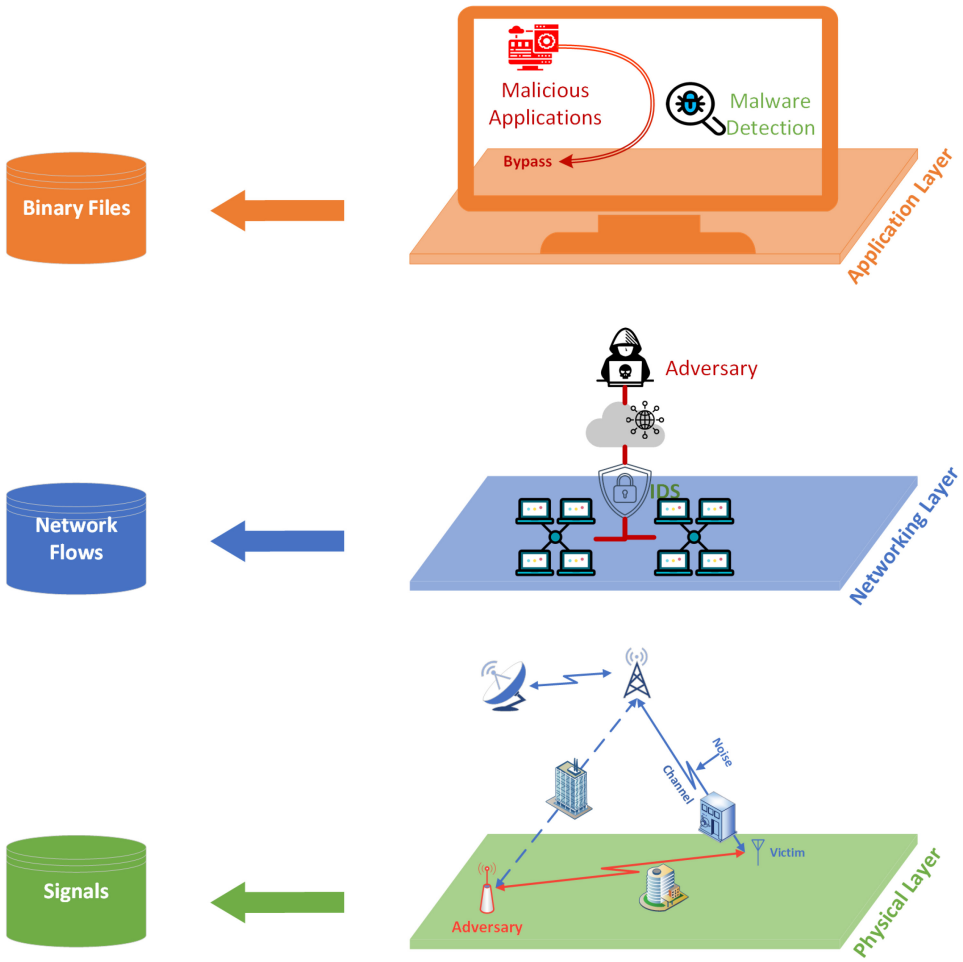

Fig. 1. Attacks in Different Layers of Wireless and Mobile Networks.

models performing the same task, even if they were trained on a different dataset.

Wireless and Mobile Networks (WMN) include but are not limited to Wireless Local Area Networks (WLANs), wireless ad-hoc networks, cellular networks, and Wireless Sensor Networks (WSNs) [22]. As illustrated in Fig. 1, in WMN, different layers contain various AML attacks. In physical layer, adversaries sense the environments/background signals, eavesdrop neighbors' channels and transmit malicious signals to target receivers to achieve goals such as downgrade the efficiency of transmission and evading physical layer authentication mechanisms. This survey combines the perspectives from network and transport layers into what is called networking layer, as most of works of AML use features extracted from both layers. Since AML literature is not sufficient explored yet in data link layer, we will not dive into it deeply. In the networking layer, attackers send elaborately designed packets/network flows to victims; nonetheless, the target networks normally deploy security mechanisms including network-based intrusion detection systems, network-based intrusion prevention systems, and firewall. Hence, in networking layer, attackers use AML to modify the patterns/characteristics of network flows to by pass the network security mechanisms. For the application layer, there are malicious mobile applications (e.g., APK, binary files) aiming to compromise or crash the target system, and malware detection is used for protecting systems from malware. In order to evade from malware detection, adversaries can add benign application's contents to malicious applications, use AML methods to reduce the features that may raise alters, and so on. One of the challenge for adversaries is to make sure that after the malicious applications are altered, they still can be executed and harmful for target victims.

As the models used for classification in the mobile systems and computer vision fields are similar, such as Deep Neural Networks as per [21], they are also vulnerable to attacks caused by adversarial samples [23]–[25]. As organized in Table I, there have been studies on the impact of ML / DL for IDS and Malware Detection (MD). The contributions of the surveys and comparative studies, such as Zhang *et al.* [5] and Otoum *et al.* [26], can be complemented by describing the impact of ML / DL classifiers for IDS and MD. Furthermore, some of the works study both the attack and defense standpoints such as [27], [28]. Alhajjar *et al.* [29] survey the impacts of AML models and techniques such as Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Monte Carlo (MC) and GAN on various machine learning models and use the NSL-KDD and UNSW-NB15 datasets. Li *et al.* [28] discuss the application of AML in Cyber Physical Systems (CPS) and the Internet of Things (IoT), such as images from surveillance system, audio input and textual data. Newaz *et al.* [27] detail the attack types and

TABLE I
SURVEYS AND FEASIBILITY STUDIES ON AI-BASED INTRUSION AND MALWARE DETECTION

Authors	Datasets	ML-based Classifiers	AML Scheme	Evaluation Metrics	Gap
Alhajjar et al. [29]	NSL-KDD, UNSW-NB15	SVM, DT, NB, KNN, RF, MLP, GB, LR, LDA, QDA, Baging	PSO, GA, GAN, MC	Evasion Rate	The authors present the results of different AML impacts on NSL-KDD and UNSW-NB15 while limited AML methods and metrics are discussed.
Li et al. [28]	Sensor Data, Surveillance, Audio, Textual	Neural Networks, ELM, KNN, RNN, Discriminator of GAN, Linear Model	GAN, FGSM, Neural Network, Iterative Optimization	TPR, FPR, FNR, Accuracy, Detection Rate, Attack Profits,	Security of cyber-physical systems is surveyed in general but network intursion or malware is not the main focus.
Martins et al. [32]	NSL-KDD, CICIDS2017	Random Forest, Decision Tree, MLP, Denoising Autoencoder, Naive Bayes, SVM	FGSM, JSMA, C&W, ZOO, GAN, WGAN, Random Noise, Q-Learning	TPR, FNR, Accuracy, Precision, F1-Score, AUC, Detection Rate, Misclassification Rate	Focus on adversarial machine learning methodologies. Limited to two networking intrusion and malware datasets.
Miller et al. [33]	None (Classified according to Attack Types)	SVM, DNN, CNN, RNN	Data poisoning, backddor data poisoning, and reverse engineering	Accuracy, TPR, FPR, ROC AUC	Miller et al. do not consider communication's applications, without mentioning WMNs
Newaz et al. [27]	None (Classified according to Attack Types)	DT, SVM	None	None	Limited number of machine/adversarial learning methods are introduced.
Olowononi et al. [34]	None (Classified according to Attack Types)	DNN, Deep Q Learning, DDQN, CNN, RNN, SVM, KNN, PCA	FGSM, BIM, JSMA, C&W, DeepFool, ATNs	None	The authors do not discuss AML from physical layer to application layer in WMNs
Otoum et al. [26]	KDD99	Random Forest, E-DBSCAN, RBM, Q-Learning	None	FNR, Accuracy, F1-Score, AUC, Detection Rate,	Performance of various machine learning methods for IDS on KDD99 investigated. Adversarial processes are not involved. The number and the type of datasets are limited
Park et al. [35]	None	DNN, MalConv, GB, DT, CNN, Drebin, Bayesian Classifier	Gradient or Problem Driven	None	Defensive approaches are not discussed
Sadeghi et al. [36]	MNIST, DREBIN, Contagio,TREC-07, CIFAR-10, GTSRB, Google TTS, ILSVRC-2012, ImageNet	LR, SVM, DBM, DNN, CNN, KNN, DT, NB, HMM, RF, RNN, DBM, DBN, RBM	Evolution Algorithms, Generative Model, Gradient Estimation, ZOO, Local Search Optimization, Hill Climbing, Deep Fool, ElasticNet, Dense Adversariay Generation, JSMA, JSMA-Z, C&W, RP2, FGS, Least Likely Class, FGS, IFGS, BFGS, L-BFGS, GAN, WGAN, Adversarial Transformation Network, UAP, Two-Person Game	Accuracy, Attack Success Rate, Denfence Accuacy	Comprehensive survey of adversarial machine learning schemes. Image datasets form the vast majority of the datasets introduced and used in the survey.
Zhang et al. [5]	NSL-KDD	CNN, LSTM, MLP, Autoencoder, Variational Autoencoder	None	None	Adversarial machine learning in security is not the main focus.
This survey article	NSL-KDD, CICIDS2017, Botnet2014, CTU-13, TRAbID, Drebin, MaMaDroid, Tencent Myapp, AndroZoo, Caontagio, VirusShare, APKPure	Random Forest, DT, MLP, Denoising Autoencoder, NB, KNN, SVM, LSTM, CNN, GB, XGB, SNN, LR, SAVER, Discriminator of GAN	L-BFGS, FGSM, JSMA, Deep Fool, C&W, ZOO, AESMOOT, Q-Learning, GAN, WGAN, IDSGAN, GAN-AAL, GEA, MIM, PGD, VAM, ElasticNet, BIM, AESMOTE, IDSGAN, UAP, RL, MRPP, MMSE, JDBA, IDBA	TPR, FPR, TNR, FNR, Accuracy, Precision, Recall, F1, AUC, MCC, G-Mean, AS, AC, ACAC, ACTC, NTE, AASR, ODR, ADR, TTC, EIR, MR, CT	N/A [This survey aims to bridge the gaps reported above]

practical defence mechanisms by focusing on health care as a use case for IoT; and despite the limited ML content inclusion, the study well-defines the scenarios of network attacks and defense. Thus, the ground work for this article lies in presenting the state-of-the-art on ML/DL classifiers applied to IDS and MD in wireless and mobile systems.

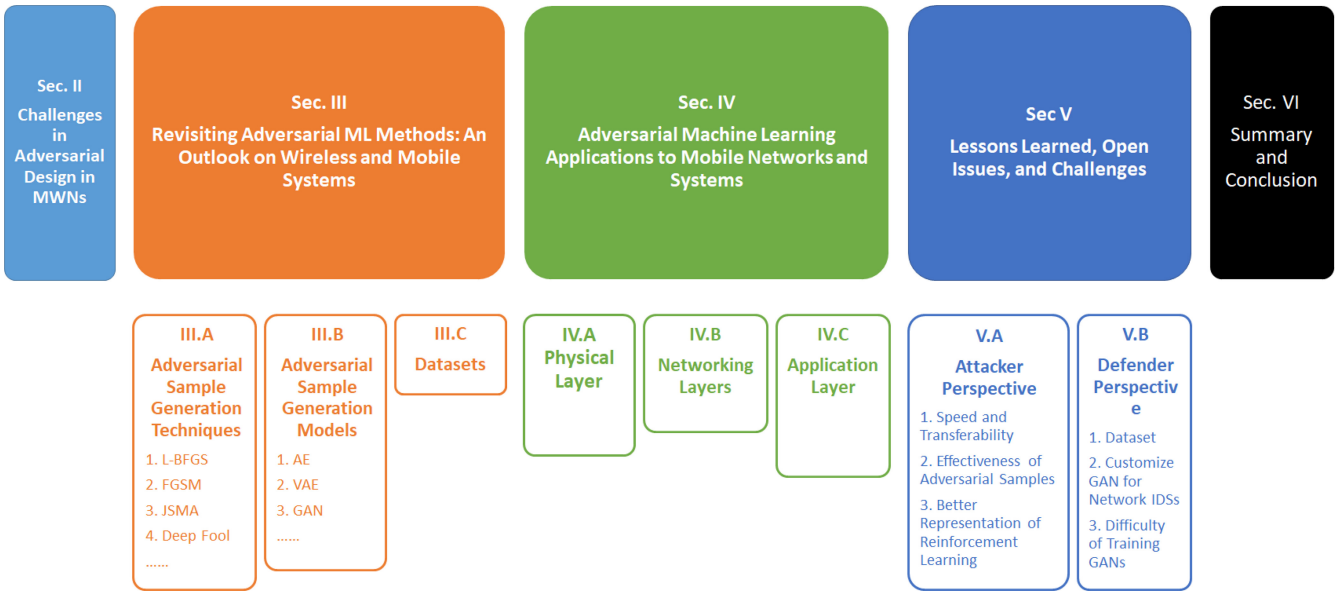

Fig. 2. Organization of the survey.

Adversarial attacks against an IDS and MD mask malicious samples, making them to seem to be normal samples [30], [31]. As further detailed in the next sections, this occurs injecting some noise to malicious samples, which aim to mislead the target classifier into categorizing samples as normal.

There have been earlier attempts at describing the state-of-the-art approaches taken towards generating attacks that leverage adversarial intelligence for signal security, IDS and MD. The closest survey/review to this article has been conducted by Martins *et al.* [32] where the authors focus on the impact of AML on network and host-based IDS and MD. In this survey, we cover eight remarkable IDS approaches that were described in that survey and, further, we review 44 additional studies on wireless signal perturbations, mobile malware detection (MMD) and IDS overall to tailor this review towards wireless and mobile settings.

To the best of our knowledge, there has not been a survey on the impact of AML towards signal security, network and host-based IDS for mobile and wireless networks from an attack and defence standpoint. Moreover, there has not been a survey that describes the impact of AML towards MMD for mobile systems. Thus, this article aims to provide an overview of state-of-the-art approaches taken towards the generation of adversarial samples and defences against the attacks created by these samples against signals, ML based IDS and MMD of wireless and mobile systems. With these in mind, the key contributions of this survey are as follows:

- a thorough review of the state-of-the-art approaches employed to generate adversarial samples against signals, IDS and MMD in mobile systems either through adversarial techniques, such as FGSM [15], or adversarial models, such as GANs [16];
- a detailed study of the approaches that use adversarial models, such as GANs, or reinforce ML / DL-based IDS with adversarial knowledge to detect such adversarial samples against mobile systems instead of the conventional ML classifiers, such as Decision Trees, to detect adversarial samples;
- a comprehensive review of the state-of-the-art adversarial generation for mobile malware detectors and defences against attacks masked by these samples;
- an extensive discussion on open issues, challenges and opportunities to assist the researchers, practitioners and professionals who work at the crossroads of mobile systems security and AML.

In light of the existing survey studies, we also identify other potential surveys that could bridge a gap in a Wireless and Mobile Network setting besides the contents introduced in this survey. For instance, surveys on AML in the physical layer are not commonly seen in the literature. Thus, AML techniques particularly focusing on the physical layer can be comprehensively summarized in the future. Moreover, for NIDS or malware detection, benchmark datasets are available such as NSL-KDD, whereas many works in the physical layer generate their own datasets making it difficult to compare their results. Therefore, a future survey that brings together the AML techniques and physical layer datasets will be helpful for the community.

This article is structured as shown in Fig. 2. Section II describes the challenges faced by the adopters of AML towards detecting attacks that leverage offensive intelligence. Section III presents the components of AML ranging from the adversarial generation techniques/algorithms, such as FGSM [15] and JSMA [18] to generative models, such as Generative Adversarial Networks [16]. Section IV details the latest studies conducted on applying AML to signals, IDSs and mobile malware detection systems. Section V provides a list of open issues and challenges, as well as the potential future directions that are identified out of the lessons learned and can

be taken by the researchers. Finally, Section VI summarizes the content in this survey.

II. Challenges in Adversarial Design in Mobile and Wireless Networks

Wireless and mobile networks employ various physical channels and specify the transmission of data and signals. Fig. 1 depicts the digital signal transmission diagrams, including source coding, encryption/authentication, channel coding, multiplexing, pulse modulation, bandpass modulation, frequency spreading, and multiple access. Driven by the desire for massive network traffic, faster speed, and spectrum efficiency, machine learning, particularly deep-learning, exhibits exceptional potential to enhance wireless and mobile network performance. Sagduyu *et al.* describe end-to-end communications as auto-encoders, considering the channel influence in their model, optimizing the entire transmission system.

In 5G or heterogeneous networks, to accommodate diverse channel conditions and improve the network utilization, deep learning plays an essential role in Adaptive Modulation Classification (AMC), which takes signals as input and determines the optimal modulation format, such as BPSK, QPSK, 8PSK, QAM16, QAM64, CPFSK, GFSK, PAM4, WBFM, AM-SSB, and AM-DSB. As the progress in wireless communication technologies continues, the available frequencies are being exhausted; thereby, wireless and mobile networks demand cognitive radio, especially Dynamic Spectrum Access (DSA), to reuse the previously allocated commercial spectrum without interfering with existing wireless system standards and primary users. For instance, when radar signal is not available in some regions, the 5G base stations (gNodeBs) can reuse part of the radar signal band (3550-3700 MHz); otherwise, the base stations should quit using those bands once radar signal is recognized. In such scenarios, deep learning can be used for spectrum sensing to decide whether to occupy the commercial spectrum without manually design radio sensing algorithms. Moreover, for security reasons, due to the broadcast nature of wireless networks, signal authentication is widely utilized in 5G and IoT networks. Deep learning can recognize user signal patterns, and avoids spoofing and replay attacks.

While machine learning brings benefits to wireless and mobile networks, the field of AML has uncovered the vulnerabilities of ML-based classifiers. With this in mind, the applications mentioned above (i.e., AMC, DSA, and signal authentication) could be exploited by adversaries using AML. Even if AML techniques or models (e.g., FSGM, JSMA, and GAN) developed in other fields (e.g., computer vision) can be utilized as core algorithms, these AML methods should be adapted to the limitations in wireless and mobile network environments, such as channel conditions, power consumption, insufficient feedbacks, hardware. Among those challenges, the most significant one is the channel condition.

As illustrated in Fig. 1, since the adversary and the receiver may not share the exact locations, the channel between the adversary and the transmitter and the channel between the transmitter and the receiver can be distinctive because of their different multipath effects, fading, and additive noise, not to mention hardware differences leading to various waveforms. Thus, the adversary system and the target system cannot share the same inputs (i.e., signals), and the receiver's channel information is almost impossible to be obtained by the attackers. A severer challenge caused by channel condition is to consider the mobility of receivers. Attacks, such as spoofing, jamming, and Trojan, require time to train the adversarial model to learn the distribution of the target system's signal, whereas, if the target receiver constantly moves, the adversarial model may need to be adjusted to the new distribution or even be retrained to continue the attacks. Additionally, unlike in the computer vision field, where pixel values can be increased or decreased, the channel between the transmitter and the receiver is already established in wireless and mobile networks. Attackers can only attach their signals to the transmitter's signals instead of manipulating the target system's inputs.

Furthermore, even though wireless networks provide attackers with opportunities to eavesdrop on receiver reactions (such as the ACK replying to DSA), the attackers cannot obtain the target classifier output labels but infer from the signals, which may lead to insufficient feedback to train the adversarial methods. Besides, the power should also be considered during an attack. With tiny power, the signal may not be received by the target system, while excessive power calculated by the AML methods may raise alerts and beyond the adversary's transmission ability. Hence, to avoid being discovered, attacks aim to minimize the power consumption, subject to a successful attack. Data poisoning and Trojan attacks demand injecting or modifying examples in the training process; then, such behavior may reduce the target receiver's performance, resulting in a low Bit Error Rate (BER), even influencing the adversary's transmission. Accordingly, the BER is an additional factor to be analyzed when evaluating those attacks.

The studies in [32], [37], [38] indicate that ML-based classifiers are prone to white box, black box or gray box attacks [39] (these attacks are explained in detail in Section III). However, there are challenges faced by AML, specifically in the security of mobile networks, that can be narrowed down to the generation of adversarial samples [8], [40]–[42]. In order to not disrupt the network traffic flow, adversarial samples aim to have perturbations only in non-functional features [41]. Attackers do not have access to the intrusion and malware detector architecture, parameters and gradients and, therefore, black box sample generation techniques have to ensure that the generated samples mislead the detection in a realistic setting [42], [43]. By improving the adversarial sample generation capability, researchers can gain more insights on the most vulnerable features exploited by such techniques [40], [44]. Here, we list down the challenges faced by researchers in generating adversarial samples for intrusion and malware detectors.

Similar to the computer vision field, the application of AML on intrusion and malware detectors in mobile systems rely on the availability of features. However, unlike the computer vision field, where perturbations are added to the pixel of an image, which is its most important feature, it is quite

difficult to alter features without affecting the functionality of the network due to their large scale deployment and operation [8], [41].

GANs need to have access to all features, i.e., functional and non-functional, of the samples. Recently, there have been attempts [41] for using GANs specifically with the non-functional features to prevent the network traffic flow from getting disrupted while evading detection from an ML based IDS. However, the generation of samples through GANs face the issue of being close to the distribution of the original samples as they are generated from noise. Sweet *et al.* [40] present the use of mutual information constrained generator to reduce the difference between the generated and original distributions and achieve a maximum similarity of 83%.

A black box attack does not have access to the target classifier gradients or the training dataset that the target classifier is trained on [45]–[47]. Hence, it builds on learning to generate samples from the labels predicted by a trained classifier [48], [49]. Thus, the search for perturbations to be added to the adversarial samples has to be optimized, then it can be tailored to work well with network attacks. A possible search strategy is proposed in [42] which alters the boundary-based search of such perturbations. As noted by the researchers, the boundary-based approach uses random-walk [50] and, thus, it does not guarantee convergence while also requiring a large number of queries to probe the target classifier. It is also pointed out by the researchers that search methods, such as Opt-Attack [51], which require less queries, could not distinguish continuous features from discrete ones. Since any normal network traffic sample consists of discrete, continuous and immutable features, Opt-Attack is reported not be an effective technique for adding perturbations. Thus, an improved search process for such samples could help researchers improve adversarial sample generation specifically for network attacks, like DoS.

MalGAN [52] is one of the first approaches to provide a black box adversarial example generation technique that generates samples without access to the parameters, gradients and the structure of the target classifier. However, when the target classifier is equipped with an adversarial example detector (or firewall as mentioned in [43]), the samples provided by MalGAN can be detected with a success rate up to 90%. In order to generate samples that could avoid detection from such adversarial detectors, Li *et al.* [43] propose an improvement to the GAN architecture.

Public datasets, such as NSL-KDD [53] and Drebin [54], used for training ML based IDS and MD consist of labeled samples. These labels denote whether the sample is an attack or a normal sample. Supervised learning is a type of ML approach which uses labeled samples to train the ML models, such as Random Forest, to identify such samples in the future. Reinforcement Learning is a sub-type of ML which self-labels such samples during training by actively observing a simulated environment [55], [56] without any supervision. For instance, AE-RL [55] proposed a reinforcement learning approach to generate adversarial samples against IDS where the model learns these labels through positive and negative reinforcement with rewards and penalties, respectively. Ma and Shi [56] enhance this approach to generate more refined adversarial samples. The major challenge faced by adversarial reinforcement learning-based generation techniques is the determination of the reward and penalty to be assigned to the generating model to refine its generation capabilities [55], [56].

III. Revisiting Adversarial ML Methods: An Outlook on Wireless and Mobile Systems

Adversarial Machine Learning (AML) studies the effects of adversarial samples generated by attacks on ML models by preventing them from functioning as expected. Kurakin *et al.* [9] describe the AML concept as follows. Given a classification model *M* and a clean sample *X* which is given as an input to *M*, the model correctly predicts the label of that sample, i.e., $M(X) = Y_{True}$. An adversarial sample *A* can be created by injecting noise/perturbations to *X*, which makes *A* perceptually indistinguishable from *X*. However, when *A* is provided to the same model, it leads to an incorrect prediction, i.e., $M(A) \neq Y_{True}$ [9]. In FGSM [15], an adversarial sample generation technique (described in Section III-A), Goodfellow *et al.* use the sign of the loss function gradients to change image pixels' intensity in an effort to generate perturbation/noise for an image. In this case, perturbations are imperceptible changes applied to an image with some magnitude (ϵ) to generate an adversarial sample. In case FGSM is applied to wireless and mobile systems, perturbations are added to original samples by changing their network characteristics or anti-malware behaviors. Goodfellow *et al.* [16] present the potential of generating such samples using GANs. Many adversarial sample generation techniques and models, such as FGSM and GAN, respectively, are developed in the computer vision field for the image classification task [37]. The classification task, however, is of paramount importance to other fields such as mobile systems, where, for instance, under the umbrella of wireless and mobile systems security, the intrusion [57]–[59] and malware [60] detection are performed as classification tasks. Similar to the models for classification in the computer vision field, the models in wireless and mobile networks are vulnerable to attacks caused by adversarial samples [21]. The impact of adversarial samples on network ML-based IDS/firewall includes but is not limited to:

- *Influence:* describes the capability of an adversary to fool the model, i.e., convincing the model that its generated instances are not adversarial but normal. In WMNs, adversaries send adversarial signals, packets, or malicious binary files to target users without being discovered by the physical layer classifier, NIDS or malware detection mechanisms respectively. This could be done as a *causative* influence where the adversary can access the training data of the target model. An alternative could be an *exploratory* influence where the adversary compromises the target model by probing it without explicit knowledge about its training data or parameters.
- *Specificity:* The attack targets one specific class as the output of the inputs to the misled classifier, i.e., *targeted attack*. If the attack does not restrict the output to a specific class but the only aim is to produce a result

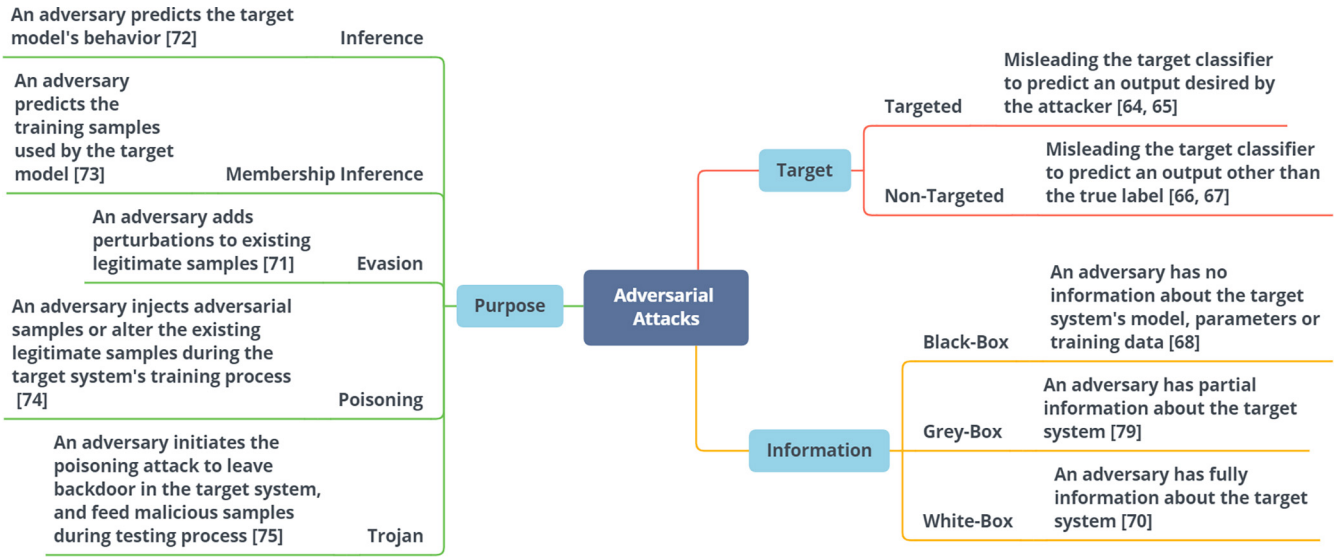

Fig. 3. Categorical presentation of attacks using Adversarial Machine Learning: purpose, targeted versus non-targeted, and the information possessed by attackers.

other than the original expected output of the model, it is called an *indiscriminate/untargeted* attack. For adversarial signals, physical layer authentication is the target, targeted attacks are commonly used to mimic legitimate users, while other targets such as Adaptive Modulation Classifier or Dynamic Spectrum Access can be untargeted since the goal of the attackers is to degrade the quality of service. Regarding adversarial network packets or malware, targeted attacks are frequently used since the objective is mostly set to misleading the target classifiers to classify malicious signals/packets/malware as benign samples.

- *Security violation:* An attack on the *integrity* of the model could deceive it so to classify a malicious sample as normal. An *availability* attack can cause the model to produce many false predictions (i.e., false positives and false negatives) that deem it ineffective. A *privacy* attack gains information about the creator or owner of the model, and thus, intrudes on their private / sensitive information.

It is worth to revisit the common terminology of AML in light of the study in [37] and incorporate this terminology into the context of wireless and mobile systems.

- *Adversarial Machine Learning Attacks:* Machine learning algorithms designed by intruders to resist security challenges or bypass defense mechanisms [61].
- *Adversarial/malicious sample:* Humans may not be sensitive to some perturbations appended to original samples, while machine learning and/or deep learning models are able to detect those patterns [62]. Those perturbations affect certain features of the sample that can lead the model to misclassifying the sample. In the physical layer, adversarial samples are wireless signals crafted to mislead authentication systems or interfere communications. Adversarial samples of a WSN intrusion can be seen as reformatted original malicious samples/flows (e.g., features as the size of packets, inter-arrival time). Regarding mobile malware, random or normal behaviors (e.g., system calls, strings) are appended to original APK or executable files to evade the target IDS.
- *Adversarial training:* The use of adversarial samples in the training set, along with original samples, to train the models and make them robust to both samples [63]. For physical layer security, researchers use various methods to augment signals in the training set such as appending Gaussian noise to signals or adding adversarial samples from GAN. During the IDS training, the generated adversarial network flows are mixed with the original flows and fed into IDS. The trained IDS can resist an adversarial attack and increase the original attack detection performance.
- *Perturbation:* Noise or random values (drawn from a specific distribution) injected to a sample with the intention of misleading the ML/DL model. When perturbations are generated by adversarial sample generation techniques, they are attached to the features of the original samples so to deceive the decision of the target IDS. For instance, Wang *et al.* [30] use various AML techniques to generate perturbations which will be added to network flow features, resulting in high evasion rate of attacks.

We further categorize the types of attacks that involve AML in Fig. 3, and visualize the attacks classified according to purpose as shown in Fig. 4. These attack types are elaborated under their corresponding categories below:

- *According to target:*
  - *Untargeted attack:* The adversary's intention is to mislead the model to predict an output other than the one it was naturally supposed to predict [64], [65]. The only control that the attacker has in this situation is to ensure that the actual output differs from the expected output. An ML-based IDS serves as a multi-class classifier that further recognizes and

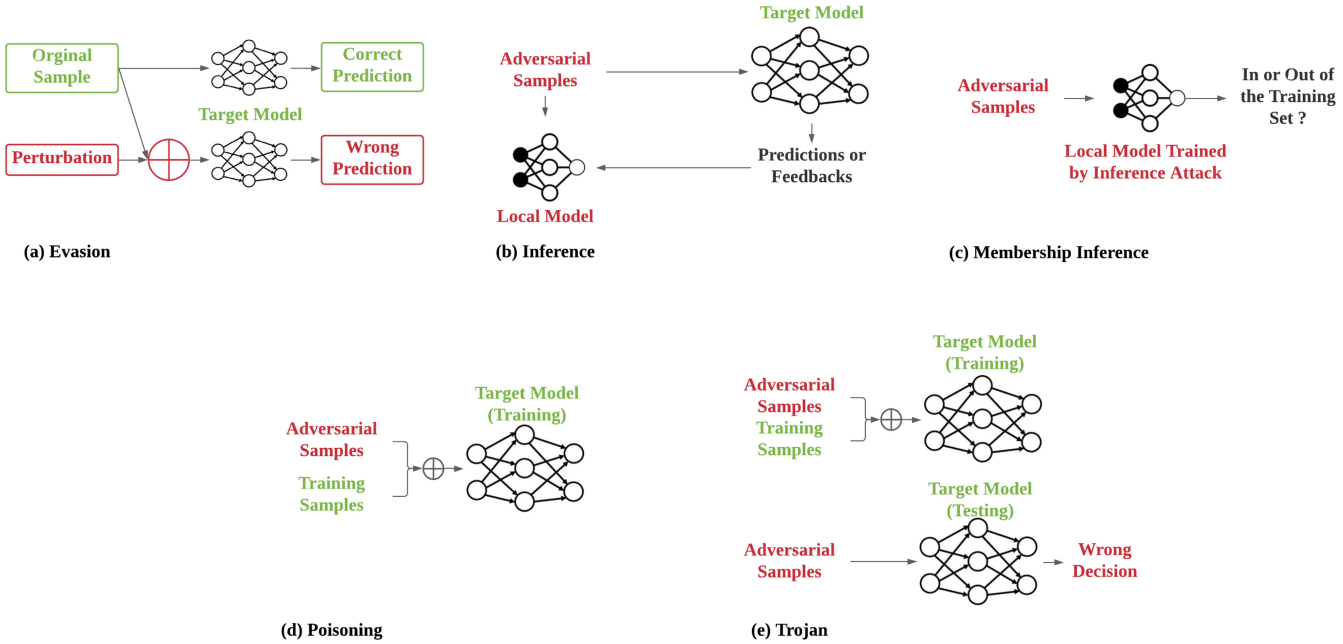

Fig. 4. Adversarial Attacks Categorized According to Purpose.

characterizes attack types, such as, DoS, Probe, User-to-Root and Remote-to-Local. Besides misleading the target IDS to classify malicious packets/flows as legitimate, misclassifying attack types and reducing output confidence are also among the goals of untargeted attacks.

- *Targeted attack:* An attack where the model is deceived to predict an output desired by the attacker [66], [67]. Thus, the output of the model that has been exposed to such an attack will still be different than the one it is naturally expected to produce. In this case it will produce an output that the attacker would like the model to produce.

- *According to the information assumed by attacker:*
  - *Black box attack:* Detailed information about the target model is not perceivable by attackers, including the parameters or weights, the training procedure, training data or architecture [68]. These attacks are developed after a trained model has been deployed since there exists no prior knowledge about the model. For instance, the model presented in [69] attacks the target IDS by only using the feedback (success/failed) to train the generator without taking any other information from the target IDS.
  - *White box attack:* If the target system is considered as white box, the attacker is aware of the model parameters or weights. In some instances, the attacker is also aware of the training procedure or training data and could thus deceive the model on the data that is known to it [70]. In WSN settings, in addition to the details about the target IDS such as, preprocessing procedure, ML-model and hyper-parameters, a white box attack also requires normal traffic of the target networks.

- *According to attack purpose:*
  - *Evasion Attack:* Evasion attacks produce adversarial samples by adding perturbations to existing samples, aiming to bypass the target classifiers [71]. In the physical layer, adversaries can fool the AMC classifiers, and make them unable to identify the modulation formats correctly, which will result in reduced throughput. For network intrusions, adversaries can alter the malicious packet network characteristics to evade the target IDS. Likewise, via mixing legitimate strings or instructions, malware can also confuse the malware detector to accept them as normal ones.
  - *Inference Attack:* The purpose of inference attacks is to learn or predict target classifier behavior by developing a local model with similar functions as those in target systems [72]. Attackers commonly apply inference attacks in smart jamming or other DoS-like attacks. An example would be a target system that employs Dynamic Spectrum Access (DSA) to enhance spectrum efficiency. In such a system, a transmitter requests the receiver enabling DSA, and the receiver replies with an ACK if the radar signal or primary user signal is not detected. An adversary can train a deep learning model to estimate when the receiver will reply with an ACK and jam the corresponding channel, reducing the target system's throughput. With inference attacks, jamming attacks can be more energy-efficient and concealed. Moreover, various goals can be achieved combining inference attacks with other types of attacks.
  - *Membership Inference Attack:* Built on inference attacks, membership inference attacks not only explore the target systems' models but determine whether a sample is used in the training process of

target systems [73]. With a surrogate model inferred using the inference attack, an adversary calculates the probability of a sample belonging to the target classifier's training samples and initiates replay attacks or bypasses the target classifier.

- *Poisoning Attack:* Poisoning attacks demand attackers accessing the target system's training process, including injecting synthetic adversarial samples to the training dataset and adding perturbations to the existing training samples [74]. Even though poisoning attacks can disguise more than the attacks that require probing in the testing process, they are difficult to implement since the target models are usually trained privately or have the data cleaning process. Furthermore, to achieve a specific goal, the number of adversarial samples that would be injected or altered may not be too scarce. Nonetheless, the target system's performance should not be degraded because of the adversarial samples.
- *Trojan Attack:* injects elaborately designed samples in the target system training process, leaving backdoors; and triggers the backdoors during the testing time by feeding the target system with samples with particular features [75]. Trojan attacks are stealthier in comparison to poisoning attacks since they inject fewer adversarial samples into the training set, while it also acquires that adversaries can access training sets. In wireless networks, take Dynamic Spectrum Access (DSA) as an example which allocates spectrum dynamically by sensing environment signals. Attackers send malicious signals when the DSA classifier of a target is in training, and mislead the target classifier in the testing process.

When non-random perturbations generated by techniques such as FGSM are added to an input the new input can persuade a trained classification model to predict a label other than the expected label. Similarly, the adversarial samples generated to attack IDS and MD systems aim to evade detection from such systems. For instance, DoS attack is a well-known attack that targets an IDS. An adversarial sample of DoS has some non-random permutation, such as the samples generated in [42], which will persuade the ML-based IDS to classify the sample as normal. Once it bypasses the IDS, it can perform the DoS attack. The rest of this section details various techniques and ML-based models that generate adversarial samples.

Besides the three categorization methods (i.e., purpose, targeted, and attacker information), as Fig. 5 shows, we further introduce a model-based classification that consists of the generative model-based techniques and non-generative techniques. The former represents the generative ML/DL-based adversarial models whereas the latter can generate noise or perturbations using optimization algorithms without leveraging deep learning models.

A. Non-Generative Adversarial Sample Generation Techniques

This subsection presents the types of attacks / adversarial sample generation techniques that can generate noise / perturbations without the use of an ML model, unlike the approaches described in Section III-B. The adversarial samples,[1] created by attaching perturbations to original samples, aim to mislead the model into classifying them under different labels than their original ones. The attacks discussed below address each other's shortcomings with the aim of producing adversarial samples more efficiently. Their efficiency is measured in terms of whether the sample gets detected as an adversarial sample or if the ML model predicts the original label instead of the intended target label. It is worth pointing out that the one common aspect among these techniques is the use of distance metrics such as L_2 (Euclidean) and L_∞ (Chebyshev distance) to measure similarity and find images similar to input image. Those similar images have different labels than the label of the input image. The following non-ML algorithms explore noise amplitude boundary that can mislead target classifiers without being discovered and the locations / features of noise that contribute most to misguiding the discriminators.

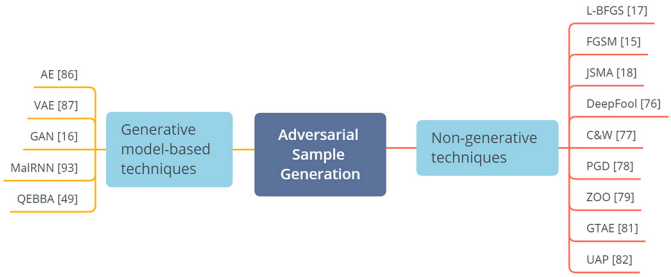

Fig. 5. Adversarial Sample Generation Methods. Non-generative techniques means methods are not based on ML or DL models, while others utilize generative ML models.

L-BFGS: Szegedy *et al.* [17] proposed the first non-generative adversarial sample generation technique after identifying certain blind spots in ML classifiers and more specifically in neural networks. The individual units of a network are activated when the unit is exposed to some features of an input. Thus, Szegedy *et al.* propose that under L_2 distance (i.e., Euclidean) a sample $x + r$ could be found. However, despite this sample being similar to x (i.e., the image), it has a different label (l) than x. Thus, the minimum amount of changes that separate the two can be learned and added to x.

These changes/perturbations are denoted as r and when added to x, a new sample $x + r$ is obtained, which is classified as l by the ML algorithm where l denotes a label. Cross entropy loss ($loss_f(x + r, l)$) measures the difference between model's outputs and the actual labels. A line search is performed for constant c, which denotes the magnitude of perturbations r. Thus, the search for an adversarial sample is transformed into an optimization problem where gradient based L-BFGS can be used to find the minimum c and r values to generate an adversarial sample as in Eq. (1).

$$\text{Minimize} \quad c|r| + loss_f(x + r, l) \tag{1}$$

This AML technique is proposed in the computer vision field, while it has been adopted in WMN settings, for example, in

[1]The terms "attack" and "adversarial sample generation technique" are used interchangeably when describing these techniques.

TABLE II
NON-GENERATIVE ADVERSARIAL SAMPLE GENERATION TECHNIQUES

Generation Technique	Description
L-BFGS [17]	Uses L_2 distance to find a similar sample with a different label. Uses constant c to alter the magnitude of perturbations r to the input x to generate adversarial sample.
FGSM [15]	Considers the gradient of the loss function of the classifier and changes the intensity of all pixels in an image based on the sign of the gradients. Fastest method of generation but also the least likely to be used by attackers as it changes all features. Similar images are found under L_∞ distance.
JSMA [18]	Saliency map of the features is generated and perturbations are added only to a select number of features which have the highest saliency values. Although this is more computationally expensive it is a more realistic approach of generative adversarial samples as it alters least number of features. Similar images are found under L_0 distance.
DeepFool [76]	Exploits the non-linear functionality of neural networks to find the closest samples to the original samples and adds least amount of perturbations needed to push the input sample onto the most-likely sample's hyperplane. More effective than FGSM and JSMA but more computationally expensive than both. Similar images are found under L_2 distance.
C&W [77]	Improves on the objective function of L-BFGS to effectively generate better adversarial samples but is more computationally expensive than all four methods mentioned above. Can be used under L_0, L_2 and L_∞ distance metrics.
PGD [78]	Randomly initializing a sample within the L_∞ ball around original input taking a gradient step to improve the quality of the adversarial sample such that it will generate the highest loss for the ML model.
ZOO [79]	A black box adversarial sample generation technique that utilizes zeroth order coordinate based Stochastic Gradient Descent to get an approximate notion of the target models gradient. Additional steps such as importance sampling and hierarchical attacks are incorporated to improve the sample generation time.
GTAE [80]	Using a DNN verification technique such as Ruplex [81] which also acts as defensive technique can be utilized to generate adversarial samples that are much closer to the ground truth and can yet mislead the model.
UAP [82]	Iteratively developing a perturbation from a sample set of training data that can impact the performance of multiple DNN architectures as well as can be applied on multiple inputs of the same type, for instance images.

WMN, signals are among possible samples. By using L-BFGS, attackers can deceive physical layer classifiers without costing excessive energy, since the amplitude *c* is minimized. In reality, the lower the energy is, the stealthier and more intelligent the attack is.

FGSM: Unlike L-BFGS, Fast Gradient Sign Method (FGSM) [15] does not search for an adversarial sample that is similar to the input, which produces a different label. Instead Goodfellow *et al.* [15] proposed a faster approach to generate adversarial samples by observing the sign of the gradient of the loss function. Based on the sign of the loss function gradient, the value of the mobile network / malware features can be increased or decreased. All features could be altered to generate an adversarial sample. Unlike L-BFGS attack, which aims to find adversarial samples under L_2 distance, FGSM optimizes under L_∞ distance to find the sample. The generation of samples is significantly faster with FGSM. All network features are at risk of being altered, which makes the generated samples more likely to be detected; for instance, some discrete features, like TCP flags, would become continuous values, or the network speed beyond the WSN normal speed.

JSMA: An alternative to generate adversarial samples is to optimize them subject to the L_0 distance that stands for the number of non-zero elements in the difference vector between the input and the adversarial sample [18]. Certain features of an input provide more semantic details about the image than other features. If the input is an image, the features would be the pixels of the image. If the input is network traffic flow, then certain functional and non-functional features of traffic provide such semantic details. The Jacobian Saliency Map Attack (JSMA) proposed in [18] generates a saliency map to denote the contribution of each feature towards classification of the input. These saliency maps are used by ML classifiers to classify the input under its actual label (y). JSMA ranks the saliency values of each feature in these maps in descending order to target the major contributors and adds some perturbations to these features with the aim of changing the input so that it can be classified under target label *l*, i.e., generating an adversarial sample from the input which had an actual label (y). The generation of saliency maps are computationally expensive; hence, JSMA is much slower than FGSM. However, JSMA adds perturbations to the lowest number of features as compared to FGSM which makes it more appealing to the generation of adversarial samples. Using JSMA to generate adversarial samples can avoid altering the WSN or malware features which are hard to be changed. However, due to its slow speed, JSMA may lead to altering the target IDS.

DeepFool: A network intrusion classification model projects a sample input *x* into *n* dimensions where a classification model can find a hyper plane to classify it under a label (class) that differs from other inputs. The final predictions for network flows or mobile malware will have one target (actual) class that has the highest probability and multiple runner-up classes which have lower probabilities. Deepfool [76] is described as an attack that keeps track of these runner-up classes whose samples are close to the actual inputs under L_2 norm but have different labels. It identifies the gradient differences between similar images and the input samples, as well as the difference in labels. By using these differences, it generates the minimum number of perturbations *r* subject to the following constraints. When these perturbations are applied to *x*, adversarial samples ($x' = x + r$) are generated, which are projected into the hyper plane of another class. Thus, the ML model is persuaded that the input x' has a different label other than its actual label y. An advantage of DeepFool over the previous methods is the lower number of perturbations it needs to generate adversarial samples to achieve the same adversarial attack outcome in comparison to FGSM. For NIDS or malware detection, adversarial training processes rely on the target system security feedback and long time probing process will raise

a system security alert. Therefore, with less probing process (i.e., adversarial training or tuning), Deepfool is safer (less probing time) and more suitable than FGSM and JSMA, even though it costs more computation.

C&W: Carlini and Wagner [77] proposed an improvement over L-BFGS algorithm by changing the loss function from cross entropy to other loss functions, such as hinge loss. Unlike L-BFGS, FGSM and JSMA, C&W attack can be used under all three distance metrics (L_0, L_2 and L_∞) with some modifications. For instance, under the L_2 distance a similar example to input *x* is found which has a target label *t*. Like L-BFGS, C&W attack continues to use constant *c* to find such an example. The changes made over L-BFGS by C&W are in terms of the objective function *f*, which accepts a confidence threshold, then one can generate highly confident adversarial samples. C&W uses minimum perturbations to generate an adversarial sample x' which is very similar to input *x* so to make the sample less likely to be detected. Hence, with a high confidence score, network intrusion or malware can even evade anomaly-based intrusion detection systems which apply confidence score to determine potential attacks. C&W has proven to produce effective samples that can evade detection from models resistant to adversarial attack such as defensive distillation [83]. While it is more effective than L-BFGS, it has higher computational overhead in comparison to all of the four techniques mentioned above.

Projected Gradient Descent (PGD): While FGSM is a single step process, Kurakin *et al.* [9] proposed an iterative FGSM process also known as Basic Iterative Method (BIM) or Iterative - FGSM (IFGSM) to further improve the quality of the adversarial samples. The values of input's features would be incrementally increased by 1 for each iteration, and the number of iteration is determined heuristically. It is critical to ensure that the improved input will remain within the ϵ-neighbourhood of the original input. Projected Gradient Descent (PGD) builds on the previous work conducted on BIM to further enhance the adversarial sample generation process. PGD aims to maximize the perturbations that can achieve significant loss while keeping the size of the perturbations within ϵ [78]. The optimal value for ϵ is considered to be L_∞. Unlike BIM, PGD starts with a randomly initialized sample within the L_∞ proximity to the original sample. It then takes a gradient step in the direction of the greatest loss and continues to move in this direction until it converges. PGD can maximize the target security system's loss at the expense of longer training time than FGSM, resulting in more adversarial training process. Therefore, it is not quite applicable in the anticipatory studies on adversarial attacks against WMNs.

Zeroth Order Optimization (ZOO): Chen *et al.* [79] propose a black box adversarial sample generation technique only has access to the input data and confidence scores of the predictions provided by a classification model. ZOO adversarial technique [79] does not require access to the structure of the model or the gradients. Instead ZOO utilizes zeroth order coordinate-based Stochastic Gradient Descent (SGD) to estimate the gradient descent of the target model. Furthermore, it does not require a substitute model to replicate the gradient descent and back propagation process of the target model in order to develop adversarial samples for the target model. However, this process requires more training to develop the samples. Hence, Chen *et al.* [79] propose additional steps such as importance sampling, hierarchical attacks and dimension reduction in the attack space to reduce the time taken to generate samples. As ZOO provides a black box solution, it has an irreplaceable role in the context of WMNs. Yang *et al.* [84] and Peng *et al.* [42] present high misclassification rates when the target classifier is a DNN-based NIDS. However, since NIDS features are also statistical features extracted from network flows, and confidence scores are difficult to obtain in a wireless network, ZOO builds on the assumption that adversaries possess partial information about the target system.

Ground Truth Adversarial Example (GTAE): Deep Neural Networks (DNN) have a high generalization capability but this also presents a drawback in terms of verifying that the DNN model will provide the expected results when it is faced with an unknown input. Thus, to verify the functionality of a small DNN model, Katz *et al.* [81] developed the Ruplex algorithm to verify the functionality of the model. The Ruplex algorithm also acts as a defence mechanism to help the developers of such DNN models identify which samples will result in an unexpected output on the DNN models. Carlini *et al.* [80] propose utilizing the satisfiability modulo theory (SMT) solver to generate adversarial samples that are closer to the ground truth than any other non-generative adversarial sample generation technique. This process can only be applied on a DNN model with a few hundred nodes. Thus, while the generated adversarial samples are unique and able to help developers understand the issues with the DNN model, they cannot be applied to large scale DNN models. Even though GTAE [80] is only suitable for small scale models, in wireless networks, constrained devices such as IoT devices may not have enough memory or computational power for large scale DNNs. Therefore, application of such methods are not visible in WMN research although it is worth to be researched.

Universal Adversarial Perturbations (UAP): Unlike previous methods where the type of perturbations added to an input is dependent on the input UAP provides a common perturbation that could be applied to all inputs belonging to a particular dataset or type of dataset. Moosavi-Dezfooli *et al.* [82] present the UAP adversarial sample generation technique as an iterative process that learns universally adoptable perturbations from a sample set of inputs. Moosavi-Dezfooli *et al.* [82] also display incorrect classification results provided by multiple DNN architectures when they are exposed to inputs with universal perturbations. UAP is shown to exploit the geometrical correlation that exist in the decision boundaries of such architectures and hence it can be universally applied to deceive similar DNN architectures. Furthermore Moosavi-Dezfooli *et al.* [82] also display that the perturbations also have significant negative impact tasks such as classifying inputs of the same type for instance, images of multiple datasets. Due to the broadcast nature of wireless environments, a generalized adversarial sample can maximize the impacts of adversarial attacks. For instance, other adversarial methods have a special target, while UAP can affect all the wireless neighbors within an adversarial transmitter's range.

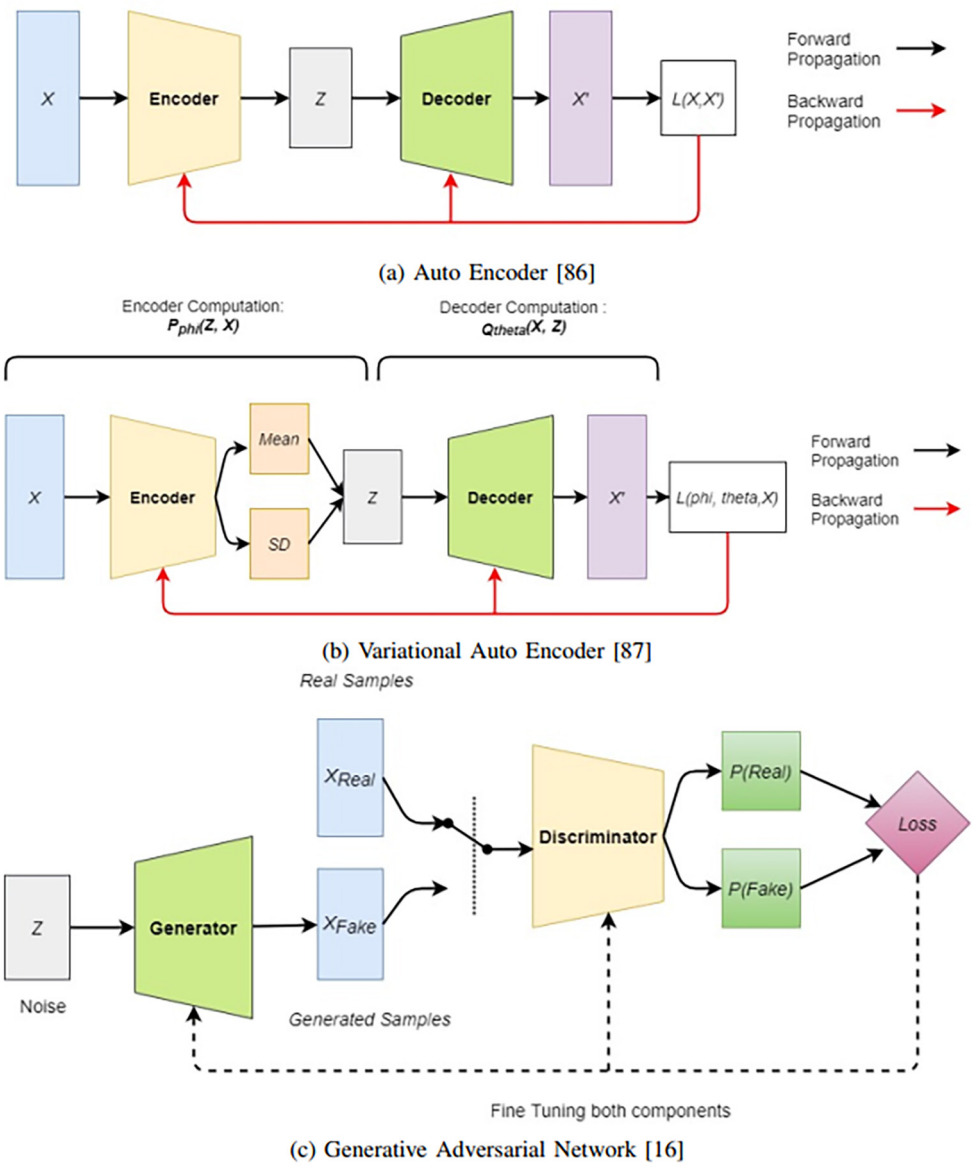

Fig. 6. Adversarial Sample Generation Models.

Research conducted by Kim *et al.* [85] has shown that UAP can reduce multiple target receivers' classifiers performance effectively.

B. Adversarial Sample Generation Models

This subsection describes the machine learning model-based generative methods including AE, VAE and GAN. The attacks described above can be made more robust if they do not use hand crafted features but rather search for the right features in the feature vector space of a sample which can mislead the classifier when altered. The advantage of such generative models is the ability to create variations of inputs/samples that could train the model in an adversarial training environment and make it more robust to such attacks. Thus, fields, such as IDS for mobile networks and systems where the number of available datasets is quite small, can benefit from such generative models. The disadvantage is that attackers could use such models to develop new forms of attacks hard to detect by the ML-based models that have not been trained on such adversarial samples.

Auto Encoder [86] is a pioneering generative model that generates latent variables/features of the input with an encoder network. The encoder network encapsulates the essential features of the input with the aim of classification/regression as an output.The latent representation of the input Z (Fig. 6(a)) can be used as an input by a decoder network. The decoder then provides a compressed version of the original input that could be used as a new sample. There are many variations of this architecture to generate samples as per the user requirement. To evaluate the quality of the output from the auto encoder, one could use mean squared error loss as in Fig. 6(a) and Eq. (2).

$$L(X, X') = ||X' - X||^2 \tag{2}$$

Variational Auto Encoder (VAE) [87] adds a probability sampling technique to pick latent variables from the input to an encoder input processing algorithm. By learning the mean μ and standard deviation σ of the input, the latent representations of the input can be restricted to more precise features. This helps to reconstruct the sample by the decoder. Thus, VAEs can produce better quality samples then an Auto Encoder as they are able to remove additional noise, that is generated without using this probabilistic sample of latent variables. The loss for a VAE consists of two parts: 1) Reconstruction loss and 2) Re-paramaterization loss.

Generative Adversarial Network (GAN): [16] generates samples from a uniform distribution (called noise in this case) rather than from the distribution of the real sample/data unlike the auto encoders. A GAN consists of a Generator *G* and a Discriminator *D*. The generator uses the noise or uniform distribution to generate samples, which are provided to the discriminator. The discriminator receives these generated samples along with real samples and has to decide whether the sample is fake or real. Initially, the generated samples are significantly different from the real samples. Both components are trained independently without one knowing the existence of the other, creating a two player non-cooperative game scenario where both players aim to win, achieving a Nash Equilibrium [89].

GANs were initially proposed in [16]. Several variations have been proposed towards improving certain aspects of them [90]. The underlying architecture of the GAN variants remains the same (as described above) with changes made to the loss functions and inputs to the generator. The functionality of GAN lies in generating a distribution of data such that it is similar to the actual data distribution [91]. However, there are certain issues, such as mode collapse [19] and vanishing gradient [92], that can prevent a GAN from converging. A mode collapse occurs when the generator successfully identifies a set of features that can mislead a discriminator and apply variations of these features in the generated samples. This reduces the overall variation in the generated samples. There are several solutions to these problems, as in [19], [20], [92]. This article limits its focus to the use of Wasserstein distance to improve the convergence of a GAN. Indeed, among many existing ones, Wasserstein distance is an effective metric scoped to measuring the closeness of probability distributions.

Additional generative models: In addition to generative models such as GANs many generative models have been developed to generate adversarial samples in specific domains. For instance multiple works have been developed for generating adversarial samples for API calls. While these approaches are not specific to WMNs, they exhibit a high generalization potential, and hence they can be applied to WMNs as well. In the following section, we describe additional generative models.

- *MalRNN* MalRNN, proposed by Ebrahimi *et al.* [93] as a black box generative model, generates adversarial samples without accessing the target model's architecture or confidence scores of the results presented by the target model. In the design of MalRNN, Ebrahimi *et al.* [93] utilize executable binary files as inputs to the proposed system. MalRNN [93] comprises of a language model which is a sequence-to-sequence Recurrent Neural Network (RNN) that is trained on binary files to develop binary sequences that appear benign. MalRNN starts off with feeding an anti-malware Deep Learning (DL) based model on a new sample generated by appending the input binary sequence to a malware sequence. MalRNN continues to iteratively improve this new sample until the anti-malware detector considers this sample to be benign.
- *Query-Efficient Black Box Attack:* A query based adversarial sample generation model for API malware detector is presented by Rosenberg *et al.* [49]. The presented approach requires limited knowledge about the detectors and significantly reduces the number of queries sent to the detector. Furthermore, in the same study by Rosenberg *et al.* [49], it is shown that considering the confidence scores of the predictions from the detectors, can improve the success rate of the generated adversarial samples up to 98%. By bridging a GAN and a self adaptive evolutionary algorithm, adversarial samples are developed for sequential as well as non sequential inputs. Additionally, the samples of the presented model are among the first to evade detection from not only CNNs, but also RNNs.
- *AC-GAN:* Odean *et al.* [94] propose an Auxiliary Classifier GANs (AC-GAN) as a generative model that develop images at higher resolution and with discriminative features. By utilizing this AC-GAN architecture, Song *et al.* [95] develop a new type of adversarial sample that is not bounded by any norm-bound perturbations. This work concludes that an AC-GAN can learn the distribution of a class-conditional dataset, and be utilized to search additional samples which are harder to be identified than samples generated by norm-bound perturbations based methods. Song *et al.* [95] define the generated samples as unrestricted adversarial samples and indicate that this process can be applied to any field where the distribution of the dataset is available.

IV. Adversarial Machine Learning Applications to Mobile Networks and Systems

Application of Machine Learning (ML) is widely accepted as an alternative to rule-based algorithms to solve various tasks in mobile networks and systems. Two sub-fields of mobile systems that have greatly benefited from using ML are intrusion [26], [96] and malware detection [54], [97]. However, Szedegy *et al.* [17] note that by injecting a certain level of noise / perturbations to the input, a trained classifier could be misled into predicting a different output than the expected output. Although the input used by the study in [17] is an image, specific adversarial samples can be generated for mobile networks and systems [43], [98], [99]. Thus, ensuring the robustness of ML models against adversarial attacks is a challenging research problem in many fields [100]–[102]. These adversarial samples are the original attack samples with perturbations injected to them so that they can avoid being

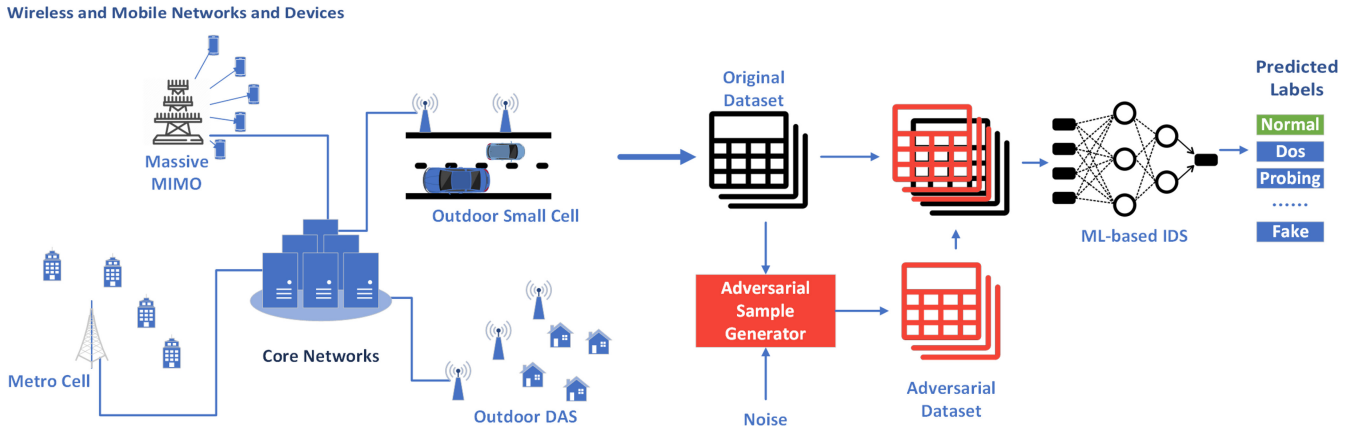

Fig. 7. General Framework of AML in WMNs. WMN part of this figure is reproduced from [88].

detected by ML-IDS and ML-MMD. Fig. 7 depicts a high-level framework of AML in WMN. In general, the first step of utilizing AML is to collect data from the target layers, i.e., sensed signals (physical layer), flow-based features generated from the network traffic (networking layer), and binary application files (APK files, application layer). According to different attack types such as black box, grey box, and white box attacks, entire or partial dataset labels and features will be used. Following upon acquiring the dataset, it is fed to adversarial machine learning algorithms to craft adversarial samples so to confuse victims. Victims or defenders receive a blend of original and adversarial samples, and their detection mechanisms (including physical layer security, Network-based Intrusion Detection System (NIDS), and Host-based Intrusion Detection System (HIDS)) should be able to identify the adversarial samples or classify samples correctly without being misled by adversarial samples.

Since an adversarial process introduces generated malicious examples into a dataset, besides traditional ML evaluation metrics, such as, accuracy, precision, and recall, many researchers propose various evaluation metrics to measure the performance of adversarial sample generator and AML/ML-IDS. It is worth to introduce the evaluation metrics in AML schemes prior to reviewing the existing studies on AML in mobile systems. To this end, we further unify and list the formulations of the evaluation metrics (e.g., confusion matrix, accuracy, and f1-score) of adversarial machine learning in Table III.

In the following subsections, we break down the AML in MWN into three categories: physical layer, networking layer, and application layer. Furthermore, for each aspect, we further review the related works from offensive and defensive standpoints, respectively. As shown in Fig. 8, for each layer, adversaries utilize different kinds of data formats to achieve their goals. In physical layer, adversaries transmit signals to harm the target receivers; attackers send well-designed malicious network flows to bypass network security mechanisms; and binary files altered by AML methods are used to evade malware detection and damage the target systems.

A. Attacks and Defenses in Physical Layer

This section introduce various adversarial and defense approaches and models within the physical layer.

Fig. 9 depicts a general adversarial scenario in the physical layer, where an adversary continuously monitors surrounding signals (i.e., signals of transmitters/base stations, receivers/targets, and background) and transmits adversarial signals or perturbations once its training process completes. Due to their different locations, the channels between the transmitter and the adversary, and the channels between the transmitter and the adversary are distinctive; as a result, channel effects constitute one of the greatest challenges of adversarial machine learning in physical layer [103]. Additionally, on account of the broadcast nature of wireless signals, an adversarial signal can affect multiple neighbor devices/targets; thereby, researchers also work on universal adversarial signals that can compromise multiple target machine learning models simultaneously [85].

We further summarise those schemes in Table IV and Table V from offensive and defensive perspective, separately.

1) Offensive Perspective: As wireless and mobile communications develop dramatically, spectrum scarcity is inevitable; therefore, many proposed solutions aim to alleviate such a problem, including exploring unlicensed spectrum, multiplexing, and Multiple-Input Multiple-Output (MIMO). Nonetheless, due to their fixed allocation scheme, spectrum, a non-renewable resource, will be exhausted eventually [110]. On the other hand, by enabling dynamically access and reuse of the licensed frequency, cognitive radio is one of the most promising technologies to solve the problem effectively [111]. To reuse the licensed band without affecting primary users (i.e., licensed users), cognitive radio follows four steps which are spectrum sensing, spectrum allocation, spectrum access, and spectrum handoff. Since machine learning can significantly accelerate the spectrum sensing, attackers introduce various approaches to compromise the sensing and accessing systems, degrading the communication service quality and spectrum utility [112].

One of the most common attacks is to broadcast particular frequency noise to jam the target channels continuously whereas such an attack can be quickly discovered and prevented via legal or technical means. Hence, adversaries demand more advanced and stealthier attacks. Shi *et al.* [107] propose an inference-like attack to achieve an intelligent jamming scheme. The authors abstract the communication system

TABLE III
PERFORMANCE METRICS USED IN THE EVALUATION OF AML TECHNIQUES

Evaluation Metric	Description	Simple formulation for a generalist
Confusion Matrix	A 2*2 matrix reporting the number of True Positives (TP), False Negatives (FN), False Positives (FP), and True Negatives (TN)	-
N_{All}, N_{TP}, N_{TN}, N_{FP}, N_{FN}	The number of samples, TP, TN, FP, and FN	$N_{All} = N_{TP} + N_{TN} + N_{FP} + N_{FN}$
True Positive Rate (TPR)	The ratio of TP to all positive class	$TPR = \frac{N_{TP}}{N_{TP}+N_{FN}}$
False Positive Rate (FPR)	The ratio of FP to all negative class	$FPR = \frac{N_{FP}}{N_{FP}+N_{TN}}$
True Negative Rate (TNR)	The ratio of TN to all negative class	$TNR = \frac{N_{TN}}{N_{TN}+N_{FP}}$
False Negative Rate / False Alarm Rate (FNR)	The ratio of FN to all positive class	$FNR = \frac{N_{FN}}{N_{FN}+N_{TP}}$
Accuracy	The ratio of correctly predicted samples to all samples	$Accuracy = \frac{N_{TP}+N_{TN}}{N_{All}}$
Precision	The ratio of TP to TP and NP	$Precision = \frac{N_{TP}}{N_{TP}+N_{FP}}$
Recall	Same as true positive rate	$Recall = \frac{N_{TP}}{N_{TP}+N_{FN}}$
F1-score	The harmonic mean of precision and recall	$F_1 = \frac{2N_{TP}}{2N_{TP}+N_{FP}+N_{FN}}$
AUC	Area Under Curve (Receiver Operating Characteristic Curve)	$AUC = P(P_{positive} > P_{negtive})$
Matthews Correlation Coefficient (MCC)	A balanced measure which represents the confusion matrix comprehensively	$MCC = \frac{N_{TP}N_{TN}-N_{FP}N_{FN}}{\sqrt{(N_{TP}+N_{FP})(N_{TP}+N_{FN})(N_{TN}+N_{FP})(N_{TN}+N_{FN})}}$
G-Mean	Useful for imbalanced datasets	$G-Mean = \sqrt{TPR * TNR}$
Attack Severity	The less attack samples being detected, the higher severity of attack	$AS = 1 - \frac{Recall_{AfterAttack}}{Recall_{BeforeAttack}}$
Average Confidence (AC)	X's average prediction confidence of the k class	$AC_k(X) = \frac{1}{N_X}\sum_{i=1}^{N_X} P(k\|x_i)$
Average Confidence of Adversarial Class (ACAC)	Misclassified samples' average prediction confidence of the k class	$ACAC_k = AC_k(X_{misclassified})$
Average Confidence of True Class (ACTC)	Correctly classified (true) samples' average prediction confidence of the k class	$ACTC_k = AC_k(X_{CorrectlyClassified})$
Noise Tolerance Estimation (NTE)	The difference between the misclassified class (k) and the max probability of other classes	$NTE_k(x) = P(k\|x) - \max_{i\neq k} P(i\|x)$
Adversarial Attack Success Rate	The ratio of misclassified adversarial samples and total adversarial samples	$ASR = FPR(X_{adversarial})$
Original Detection Rate (ODR)	The classifier's detection ability for original samples	$ODR = TNR(X_{original})$
Adversarial Detection Rate (ADR)	The classifier's detection ability for adversarial samples	$ADR = TNR(X_{adversarial})$
Total Time Cost (TTC)	Training and testing time consumption	$TTC = T_{end}^{Train} - T_{start}^{Train} + T_{end}^{Test} - T_{start}^{Test}$
Evasion Increase Rate (EIR)	The rate of efficiency of adversarial sample generator	$EIR = 1 - \frac{ADR}{ODR}$
Weighted Normalized Conditional Entropy	Used for examining the generator performance	$\hat{H} = \sum_{i=1}^{N_X} \frac{\|w_i\|}{\|w\|} \sum_{j=1}^{N_{class}} P(x_i\|j) \log \frac{1}{P(x_i\|j)}$, where $\|w_i\|$ is the weight of a sample
Effective Generation Rate (EGR), Evasion Rate (ER), Misclassification Rate (MR)	Adversarial examples' FPR	$FPR(X_{adversarial})$
Crafting Time	Training Time Consumption	$CT = T_{end}^{Train} - T_{start}^{Train}$

as a transmitter that senses the environment via a deep learning model (e.g., CNTK), discovers spectrum holes (i.e., unused licensed spectrum), and occupies the spectrum sending data if the primary user is inactive; meanwhile, a receiver replies with ACK to confirm successful data transmission. The adversarial scheme develops a local deep learning model (different from the one in the transmitter system), which takes sensed signals as the input and the receiver's ACK as labels (assuming that adversaries can obtain the receiver's signals). Once the adversary predicts an ACK, it initiates a jamming attack, blocking the transmission. Compared to an authentic inference attack, the proposed scheme does not infer the transmitter's deep learning model; instead, the adversarial model aims to infer whether transmission will succeed. Additionally, the inferred model and the transmitter's model turn out to be significantly different due to their different locations and labels (i.e., ACK/NACK for the adversarial model; IDLE/BUSY for the transmitter's model). This scheme has also been shown to be applicable in an environment with multiple transmitters and receivers.

Even though an intelligent jamming attack interferes with the communication during the data transmission, it can potentially trigger alerts and cost considerable energy. As a result, the study in [103] anticipates improved intelligent jamming by introducing an evasion attack to compromise the target sensing and accessing system, aiming to mislead the target system via false estimation of the channel status. To tamper with the target system's sensing process, the adversary needs to speculate the target system's spectrum sensing time slot via sampling and processing multiple ACK time intervals. Once the sensing time slot is determined, the adversary can easily mimic a primary user's behavior by sending signals and occupying the

Fig. 8. Application of Adversarial Machine Learning to Mobile Networks and Systems.

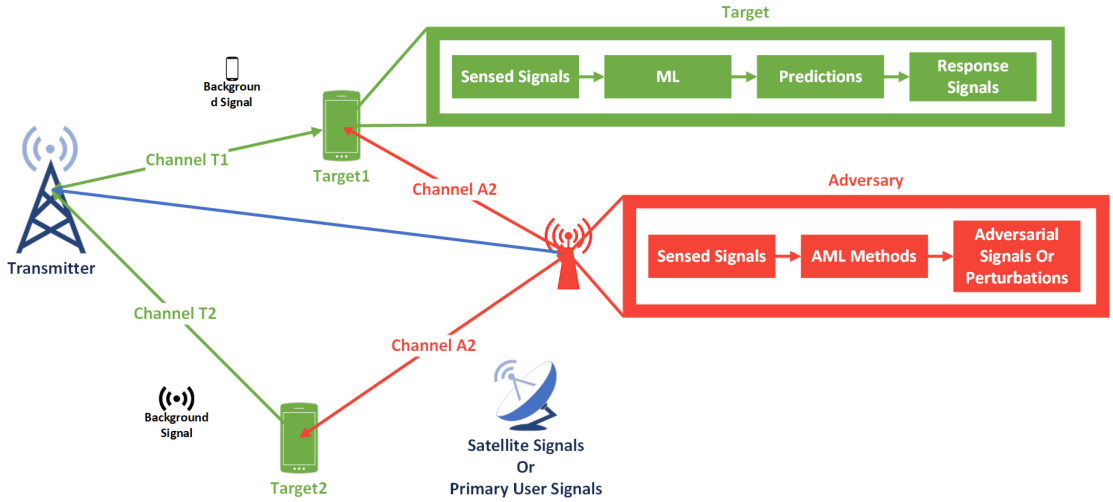

Fig. 9. Attacks and Defenses in Physical Layer. Color codes are chosen to represent the following relations. Green: Target receiver and the channels between the transmitter and targets; Red: the adversary and the channels between adversary and targets; Blue: the transmitter and the channels where the adversary use it to eavesdrop signals.

target channel. Instead of brute-forcing to jam the data transmission, the evasion attack can achieve a higher success rate with less energy budget.

As devices and background noise may change constantly, a sensing system requires to retrain its classification model periodically to adjust to the dynamic environment. On the other hand, this results in an attack surface for poisoning attacks, which pursue an alternative approach to jeopardize the target system's spectrum sensing mechanism. The initial stage of a poisoning attack is to estimate when the target system will be retrained by continuously monitoring the receiver's ACK since a poisoning attack aims to inject samples into the

TABLE IV
ADVERSARIAL SIGNAL GENERATION SCHEMES [SIGNALS – OFFENSIVE]

Title	AML Schemes	Target ML Classifiers	Targeted	Attacker Info	Result	Evaluation Metrics
Davaslioglu et al. [104]	Signal rotation	CNN	No	White Box	Higher than 80 attack success rate with 10 poisoned samples	Attack success rate, Accuracy
Karunaratne et al. [105]	Reinforcement learning	CNN	No	Black Box	99 fooling rate when the SNR is larger then 15 dB	Fooling rate, Convergence time
Kim et al. [106]	Channel Inversion, MMSE, MRPP, UAP	DNN	Yes	White and Black Box	UAP can reduce the target system accuracy with highest rate under high PNR	Accuracy
Kim et al. [85]	Channel Inversion, MMSE, MRPP, UAP, IDBA, JDBA	DNN	No	White Box	IDBA and JDBA can successfully influnce multiple neighbor receivers	Accuracy
Shi et al. [107]	Inference based jamming	CNTK	Yes	Black Box	Reduce the target system success ratio from 73.79 to 2.91	Througput, Success ratio
Sagduyu et al. [103]	Time slot sensing, inference, evasion and causitive based jamming	CNTK	Yes	Black Box	With the combination if causative and jamming reducing the target system success rate from 96.94 to 25.95	Througput, Success ratio, Energy cost
Shi et al. [108]	Replay attack, GAN	DNN	Yes	Black Box	GAN-based spoofing can achieve 76.2 attack success probability	Success probability
Shi et al. [109]	GAN with consideration of locations, channels and MIMO	DNN	Yes	Black Box	Success probability: 98.6	Success probability

TABLE V
ADVERSARIAL SIGNAL DETECTION SCHEMES [SIGNALS – DEFENSIVE]

Title	Defense Approaches	Adversarial Mechanisms	Targeted	Attacker Info	Result	Evaluation Metrics
Davaslioglu et al. [104]	Augmentation with rotated signal, Clustering	Signal rotation	No	White Box	Accuracy larger than 98	Accuracy
Flowers et al. [118]	FGSM	Eavesdropping	No	White Box	Lowest BER is achieved under 12 dB SPR	Accuracy, BER
Kim et al. [85]	Randomized smoothing	Channel Inversion, MMSE, MRPP, UAP, IDBA, JDBA	No	White Box	The study demonstrates that the accuracy changes according to PNR and parameters	Accuracy
Merchant et al. [119]	Adding adversarial samples in training set	GAN	Yes	Black Box	The classifier's performance will be improved more than 35	ROC-AUC
Roy et al. [120]	CNN, DNN, LSTM, GRU	GAN	Yes	Black Box	Achieve 97.85 Accuracy by using GRU	Accuracy
Shi et al. [107]	Intentionally wrong decision	Inference jamming attack	Yes	Black Box	20.79 misdetection rate with 10 wrong decision rate	Throughput, Success Ratio, Misdetection Rate, False Alarm
Sagduyu et al. [103]	OptDefense	Inference, evasion, causative, and combined jamming attack	Yes	Black Box	15.63 Throughput under 10 wrong decision rate	Throughput, Success Ratio, Misdetection Rate, False Alarm

target system's training data. If the retraining is in progress, the adversary pursues a poisoning attack by injecting crafted adversarial signals into training data. The adversarial signal generation can be either white box (i.e., using the target classifier) or black box (i.e., building a local model via the inference attack). It has been shown in [103] that the poisoning attack further reduces the energy cost in comparison to the evasion attack. With the combination of evasion attack or jamming attack, the poisoning attack can cause even more damage to the target system.

As one of the vital functions in cognitive radio systems, AMC can maximize the spectrum utilization without network protocol overhead by automatically and adaptively identifying the modulation format of signals [113]. As deep learning can directly take signals as inputs, it has been extensively researched and applied in the AMC field, resulting in the most promising results. Although deep learning-based AMC approaches have been deployed in military and civilian communication systems, adversaries are exploring the attacks targeting the deep learning-based classifiers in AMC [114].

Kim *et al.* [106] comprehensively take channel influence, transmission power, and the broadcast nature of wireless and mobile networks into consideration in their work, and propose three adversarial algorithms for targeted and non-targeted attacks to fool the AMC classifiers misclassifying the format of signals: naive attack, Maximum Received Perturbation Power (MRPP), and Minimum Mean Square Error (MMSE) attacks. The naive attack is the simplified version of the MRPP attack without analyzing channel conditions' influences. For

each class, the naive targeted attack calculates the normalized perturbations by straight using the targeted Fast Gradient Method (FGM) and determines the minimum power via binary search. At last, the class with minimum power is selected as the target class. The non-targeted naive attack calculates the normalized perturbation by non-targeted FGM, and gradient descent is applied to determine the amplitude of the perturbation. To take the channel effect (e.g., multipath, fading, and pathloss) into account, MRPP attacks multiple the normalized perturbation with the channel matrix between the adversary and the receiver. MMSE considers the channel effects differently, which minimizes the difference between the received perturbations and the naive perturbations; thus, the problem is formulated with respect to quadratic programming. Among these three attacks, MRPP achieves the best performance when the Rayleigh channel is applied in the experiments. As the channel information cannot always be well-known by attackers, attackers can sample the channel information and calculate and stack the perturbation for each sampled information. Finally, PCA is utilized to obtain the ultimate perturbations.

To relax the restriction of white box attacks, in [85], Kim *et al.* investigate the possibility of using the transferability of adversarial learning in wireless and mobile networks. With pre-collected data and a substitute local deep learning model, the adversary can attack other DNNs with different layers and neurons to some extent. Due to wireless communication's broadcast nature, adversaries can maximize the impact of adversarial attacks through broadcasting the perturbations to attackers' neighbor devices. With these in mind, the work prooposes a scheme which estimates each neighbor's perturbation according to the introduced adversarial above and weighted summarizes them to form a broadcast perturbation.

In addition to the evasion attacks introduced above, Trojan attacks are also applicable in wireless and mobile environments. As discussed in Section III, Trojan attacks involve altering or crafting data during the training process to leave backdoors that are to be triggered in the testing time. Davaslioglu and Sagduyu [104] propose an algorithm that poisons original signals via rotating them by θ degrees in the constellation diagram with different labels from the original signals. In the testing process, the target classifier cannot identify the correct modulation type via rotating the signal with the same degrees. As the research suggests, with slight disturbance and the tiny number of poisoned samples, the classifier can be deceived.

Current authentication of 5G or other mobile and wireless networks rely on upper-layer protocols (e.g., Authentication and Key Agreement (AKA)); whereas, not only is the additional cost required, such as delay, computation, and energy, but also the distribution and management of cipher-key are becoming a burden for the system [115]. Therefore, one of the promising and thriving solutions is physical layer authentication via utilizing the transmitter hardware's imperfection [116]. For instance, In-phase and Quadrature (IQ) modulation is extensively applied in wireless communication, which contains various hardware, including but not limited to Digital to Analog Converters (DAC), filters, and Power Amplifier (PA). Every piece of hardware introduces signal distortion (e.g., IQ imbalance, phase noise, and non-linear PA distortion), forming each transmitter's unique fingerprint and making physical layer authentication possible [117]. Thus, a physical layer authentication system can be considered as a classification model (e.g., CNN) that takes signals within a time window as input and identifies which user sends these signals. Even though both Physical Layer Authentication (PLA) and AMC are both classification tasks, inference and evasion attacks are not commonly seen in PLA since an adversary must craft its own signals to be accepted by a transmitter. With this in mind, the spoofing attack is a viable way for an adversary to bypass the authentication procedure [103]. Replay attacks can be an effective way to compromise the target authentication system, which eavesdrop on the transmitter's physical signals and sends them back to the receiver afterward. Nevertheless, as many techniques can defend against replay attacks, even the authentication system can identify most replay signals, Shi *et al.* [108] introduce a GAN-based spoofing attack. The adversary overhears the transmitter's samples and feeds them into a GAN model, assuming a black box target system (i.e., the GAN's discriminator differs from the classification model used in the receiver). The research demonstrates that the GAN-based spoofing attack outperforms the replay attacks and random signal attacks. Additionally, even the adversary's location is changed, the GAN-based attack can still compromise the target authentication system.

As discussed earlier, the signals received by the adversary and the receiver are distinctive due to channel influence, whereas, even if the introduced spoofing attack achieves a high bypass rate, the wireless channel impact is entirely neglected; thereby, Shi *et al.* [109] integrate the channel information into the GAN model. Not only are the eavesdropped transmitter signals (which are fed into the discriminator) filtered by channel, but an additional layer is injected between the GAN's generator and discriminator to simulate the channel effects (e.g., Gaussian channel, Rayleigh fading channel, and Rician fading channel). Simulated results report that the GAN-based spoofing attack with channel information can have even higher performance than [108]. Moreover, in the same study, implementation on a real-life scenario via FPGA shows the availability and the fast convergence of the adversarial model.

In addition to GAN-based spoofing attacks, Karunaratne *et al.* [105] adopt a reinforcement learning approach to bypass the PLA mechanism and capture the temporal relationship among signals. A typical Markov Decision Process contains four elements, i.e., states, actions, transition probabilities, and a reward function, and a policy that maps the current state to an optimal action. The proposed scheme in [105] treats the sensed signals and channel as states, the generated adversarial signals as actions, the channelized discriminator's outputs as rewards, and a deep learning model (i.e., generator) as the policy function. The generator is initialized by Mean Square Error (MSE) locally to reduce the probing (i.e., training in the environment) time. The proposed reinforcement learning-based spoofing attack is proven to be effective in various environment settings with respect to SNR, channels, and hyperparameters.

2) Defensive Perspective: Offensive approaches assist researchers to propose defense mechanisms to mitigate those adversarial attacks designing robust and secure systems. As discussed in Section IV-A1, a black box attack targeting DSA requires collecting enough feedback as labels (e.g., physical layer ACK) from wireless environments to train adversarial models, such as GAN or AE. Since adversaries also utilize machine learning or deep learning models which rely on defenders' response, Shi *et al.* [107] propose that defenders can actively initiate a poisoning attack to confuse attackers' models by intentionally making wrong decisions. In detail, a transmitter will deliberately flip the decisions predicted by its machine learning model (i.e., transmitting when the channel is busy and vice versa) with a slight possibility. The predictions with high confidence scores are expected to be flipped to maximize the effectiveness of the defending schemes against the adversaries. Inevitably, communication performance of the transmitters, such as spectrum utility and packet loss rate, will be reduced via deploying such a mechanism; whereas the results in [107] demonstrate that, with a small false decision score (10%), the misdetection rate of adversarial models can be increased remarkably (more than 20%). Sagduyu *et al.* [103] further improve their model by formulating an optimization problem. The OptDefense proposed by Sagduyu *et al.* maximizes the inaccuracy of the adversarial model ($1-$*accuracy*) while the adversarial model's misdetection rate and false-positive rate and the wrong decision rates are constrained by the inaccuracy and a threshold, respectively. Their results prove that the OptDefense can significantly increase the system throughput without significant sacrifice in the adversarial model's misdetection rate compared with the vanilla wrong decision scheme.

Without adversarial training, AMC's deep learning models are prone to various attacks and distorted signals (such as rotating wireless signals on constellation diagrams with small angles); thus, Davaslioglu and Sagduyu [104] propose using signal rotation to augment the training set to mitigate those attacks. The results show that the proposed approach can significantly reduce the attackers' success rate; nevertheless, it can barely deal with other adversarial attacks. As a result, the authors further introduce a clustering algorithm (i.e., t-SNE) to identify potential risks. Even though the accuracy is high, its false alert rate can be further improved.

To defend against adversarial attacks targeting AMC, Kim *et al.* [85] propose augmenting training sets via randomized smoothing, which crafts additional samples by adding Gaussian perturbation to original samples. In comparison to adversarial training, which assumes defenders know attackers' adversarial schemes in advance and applying the same scheme to augment the training set, randomized smoothing does not need such assumption and can generate isotropic new samples avoiding biasing to one or some of the adversarial methods. The authors' experiments demonstrate that, with proper parameter settings (i.e., the number of additional samples and Gaussian variance), randomized smoothing can make machine learning models robust against adversarial attacks.

In addition to robustifying AMC, researchers also pay attention to adversarial attacks targeting physical authentication. Roy *et al.* [120] test the performance of CNN, DNN, LSTM, and GRU-based RF fingerprint classification systems when adversarial samples are included in datasets. Assuming that adversarial samples are generated from GAN, the authors find that, due to I/Q samples' lack of notable spatial correlation, CNN can hardly extract useful features or information, resulting in unsatisfied performance. On the other hand, DNN or Fully Connected Layers achieves higher accuracy than CNN. If the time-series property can be considered via LSTM or GRU, accuracy can be further boosted to more than 97%. Merchant and Nousain [119] propose adding adversarial samples generated from GAN into the training set to resist adversarial attacks. The authors assume attackers are using GANs to generate adversarial RF signals, and the target classifier (CNN-LSTM)'s performance is reduced to 62% in terms of the Area Under the Curve (AUC) for Receiver Operating Characteristics (ROC). In contrast, if the target classifier is trained with adversarial samples (generated from another GAN which is different from the one used for attacking), the classifier can achieve more than 99% ROC-AUC.

Besides active attacks which aim to jeopardize the target system, due to the broadcast nature of wireless environments, passive attacks (e.g., eavesdropping and traffic analysis) are also worth to be researched to guarantee user privacy. Unlike active attacks, as eavesdroppers will not transmit any signals, most reactive approaches such as detection systems, are not viable under such scenarios. On the other hand, proactive approaches (e.g., encryption and authorization) are more commonly seen in wireless settings. Instead of introducing additional overhead to upper layers, Flowers *et al.* [118] propose adding perturbations at transmitter signals to fool the eavesdropper's AMC classifier without influencing the legitimate receivers. FGSM is utilized with power constrain to generate the perturbations. Experiments and analysis under Additive White Gaussian Noise (AGWN) channels show the efficiency of this approach.

B. Attacks and Defenses in Networking Layer

Intrusion Detection Systems (IDSs) offer attack detection on networks or hosts. According to the type of the monitored data, an IDS can be classified as Network-based IDS (NIDS), Host-based IDS (HIDS), or Collaborative IDS (CIDS) [121]. NIDS observes communication/traffic in a network environment to discover the misbehaved packets or flows [122], while HIDS checks the system behaviour of a host including but not limited to system events, system calls and shell logs so to capture unauthorized or malicious actions performed by software [123], [124]. Neither NIDS nor HIDS is sufficient to recognize all novel threats. For instance, NIDSs are well-known for their real-time capability whereas encrypted malicious payload can be difficult for NIDSs to deal with [125]. HIDSs excel in the detection of attacks missed by NIDS or network firewall, whereas advance persistence attacks can be handled by HIDS, not to mention many of them target the HIDS itself [126]. Therefore, CIDSs (i.e., a combination of NIDS, HIDS and other techniques) emerge to protect the whole system holistically [121]. At the network layer, this survey focuses on

ML-based NIDS and the threats that a network can encounter. Recent focus in IDS has shifted to using Machine Learning and Deep Learning with the latter looking prominent in the current literature [26], [96], [127]. Since Deep Learning methods take away the need for manual feature extraction that is required by Machine Learning models, many of the latest IDS have begun to use Deep Learning methods ranging from Variational Auto Encoders [128], [129] to Generative Adversarial Network (GAN)-based architectures [130], [131]. As Deep Learning (DL) primarily leverages neural networks, it is a branch of ML. Hence, we refer to both DL and ML based IDS as ML-IDS throughout this article.

ML-IDSs are susceptible to adversarial attacks that mask the input / sample as normal traffic although the sample turns out to be malicious [132]. Using AML, an adversary can find such loopholes in the target classifiers and can exploit these loopholes by sending inputs that are inherently network traffic attacks, such as DoS [42]. Attackers can use different generation techniques to build such malicious samples. Generative models, such as Auto Encoders [133], Variational Auto Encoders [129] and GANs [44], [84], [134], can generate such samples or help / act as ML-IDS. Adversarial sample generation techniques [30], [98], [135], such as FGSM and JSMA, can generate adversarial samples. Thus, with the growing dependence on ML-IDS, there is an urgent need to develop more robust IDSs that can tackle detection, such samples, and prevent the attacks they can cause, such as DoS. This section describes the latest works that have applied AML to ML-IDS to generate adversarial samples and defend against them.

In order to cover all areas of network intrusion detection, we include both network- and host-based ML-IDS in this section. We divide the content into three subsections for network traffic samples. First, we describe adversarial samples generated by techniques such as FGSM and JSMA. Second, adversarial samples generated by adversarial models, such as GANs, are presented where we summarize the state-of-the-art approaches to apply generative models towards the creation of such adversarial samples. Lastly, we present the adversarial models, as IDS, which are more robust and capable of detecting novel attacks when compared to the traditional ML-IDS by drawing a more clear boundary of regular traffic through learning from the adversarial samples. This subsection presents the application of AML in WMN's network layer from attackers' and defenders' perspective. We further summarize and compare the related research in Table VI, Table VII, and Table VIII.

1) Offensive Perspective: Rigaki and Elragal [98] are among the first researchers to point out the importance of adversarial training for ML-IDS. When adversarial samples are generated through FGSM and JSMA, both methods require access to the predicted class of a sample by a model to understand which features have to be changed. Hence, a pre-trained MLP is used as a substitute classifier to test these methods and generate adversarial sample. In the above mentioned work, these samples reduce the classification accuracy of both models by 16% and 27%, respectively. Both models fall short in the detection of certain samples which are attacks that are falsely classified as normal traffic. Based on these, adversarial training is needed to improve the detection of such samples as none of the tested ML models provide the optimal results following upon the addition of the adversarial samples to the test set.

Unlike Rigaki and Elragal [98], Wang *et al.* [30] test the efficiency of applying other adversarial sample generation techniques on an MLP. The train-test set split transfers the imbalance of the original dataset and, hence, the F1-score performance in the detection of DoS attacks is significantly higher than that in the detection of R2L and U2R attacks. When three variants of FGSM are employed to generate samples, they are affected in a similar way. The variants are: untargeted (i.e., no prior setting of target label for sample), least-likely targeted (i.e., from the final prediction, pick the class with least probability as target) and random-target (i.e., randomly pick a target class). Applying JSMA, only nine features are needed to be changed to result in 100% misclassification. It should be noted that since it is crucial to change only a very small subset of features, JSMA is also a recommended method by Wang [30].

Martins *et al.* [135] focus on intrusion systems against DoS attacks and provide an analysis on adding adversarial samples to the NSL-KDD and CICIDS-2017 datasets. The study presented by Martins *et al.* build on three variants of these datasets: (*i*) Original-Original (i.e., training and test sets are original data); (*ii*) Original-Adversarial (i.e., training set is original while test set has adversarial samples) and (*iii*) Adversarial-Adversarial (i.e., both training and test sets contain mutually exclusive adversarial samples to avoid overfitting). The ML-IDS techniques employed in this study are DT, RF, SVM, NB, DNN and DA, and they lead to similar results as in [98] with demonstrating the best performance on the Original-Original set. However, on the Original-Adversarial set, it is reported that Denoising Autoencoders perform the best as they add noise to the training samples while being trained. Under the Adversarial-Adversarial set all models experience performance drops whereas RF stands out to be a strong classifier due to its ensemble nature and randomness factors, which can regularize the model and prevent over-fitting, even if there are adversarial samples in the training set. It is also reported that JSMA is a preferable choice for the generation of adversarial samples for intrusion detection. Although it does not outperform FGSM and DeepFool, it alters fewer features.

Apruzzese *et al.* [136] note that once the attackers exploit the vulnerabilities of ML-IDS, traditional ML-IDS need improvements to defend against such adversaries. One can use botnet detectors as an example to present the motivation for these improvements. A bot is an end device that is infected by malware to provide details to the attacker about the internal network, and performs various attacks [146]. A number of bots connected via Internet form a botnet to perform various types of attacks including Distributed DoS (DDoS), spamming, and data theft. Thus, a botnet detector is an NIDS variant aiming to detect an attack that is generated by a botnet [147]. These detectors assume that the bot is already planted in the internal network. Unlike the attacks described earlier, this can be caused by changing the network flow values such as exchanged_bytes, duration, total_packets.

TABLE VI
EXISTING AML-BASED RESEARCH TO ANTICIPATE ATTACKS ON ML-IDS [PACKETS – OFFENSIVE]

Title	AML Schemes	ML / DL Classifier	Dataset	Result	Evaluation Metrics
Rigaki et al. [98]	FGSM, JSMA	RF, Linear SVM	NSL-KDD	Reduced detection performance of both by 16 and 27. JSMA is preferred since it altered only 6.14 of the network traffic features.	Accuracy, F1-score, AUC
Wang [30]	FGSM, JSMA, DeepFool and C&W	MLP	NSL-KDD	100 misclassification with samples generated by JSMA which also changed only 9 features. DeepFool and C&W changed on average 65 and 115 features per sample. JSMA is preferred again.	Accuracy, Precision, Recall, False Alarm, F1-score
Martins et al. [135]	FGSM, JSMA, DeepFool and C&W	DT, RF, SVM, NB, DNN and Denoising Autoencoder (DA)	NSL-KDD, CICIDS-2017	Three combinations of train-test splits with and without adversarial samples were created to see the impact. DAE performed the best on Original train - Adversarial test set. RF performed best on Adversarial train - Adversarial test set. JSMA was preferred again.	F1-score, AUC
Apruzzese et al. [136]	Incrementally adding noise to 3 features	RF, MLP, KNN	CTU-13	RF recall reduces from 96 to 34. MLP achieved 27 recall after data poisoning. RF and MLP perform worse after adding adversarial samples to training set	Accuracy, Precision, Recall, F1-score, Attack Severity
Apruzze et al. [137]	Incrementally adding noise to 4 features	RF, MLP, AB, BT, KNN	CTU-13, CICIDS-2017, UNB-CA	On CTU-13 RF, AB and BT achieve the highest recall. However recall drops on average by 55	Precision, Recall, F1-score
Peng et al. [42]	ZOO	ANN	KDDcup99, CICIDS-2017	Adversarial samples of Neptune and Smurf (which are sub-classes of DoS samples) are tested and resulted in 100 and 99 misclassification.	Accuracy, Average Confidence, Average Confidence of Adversarial Class, Average Confidence of True Class, Noise Tolerance Estimation
Zhang et al. [138]	Monte Carlo Tree Search (MCTS) algorithm	Discriminator of GAN, LSTM and XGB	CICIDS-2017	Only XSS attack samples are used. Lower TPR indicates generation technique is promising. Higher TPR indicates detector is promising. MCTS generated samples produce 6 lower TPR than samples generated through FGSM. GANs trained on original and original + adversarial train-test highest TPR.	True Positive, Precision
Ayub et al. [139]	JSMA	MLP	CICIDS-2017, TrabID 2017	MLP performance reduced to 22.52 and 29.87 on each dataset respectively.	Precision, Recall, F1-score
Piplai et al. [140]	GAN + FGSM	Discriminator of GAN	IEEE BigData 2019 Cup: Suspicious Network Event Recognition challenge dataset	Discriminator is trained on samples generated by generator of GAN. On applying FGSM to these samples success rate of avoiding detection is at 96 and 60 when all and top-two features are not used respectively.	Attack Success Rate, Precision, Recall, F1-score
Zhang et al. [69]	Brute force search which uses only predicted labels for measuring improvement	LR, DT, RF, MLP, NB	NSL-KDD	ADR of BFAM prove to be preferable over the samples generated by GANs on Dos, U2R and R2L adversarial samples. Lowest ADR achieved by BFAM is 0.13.	Original Detection Rate, Adversarial Detection Rate, Total Time Cost,
Ibitoye et al. [141]	FGSM, BIM, PGD,	FNN, SNN	BOT-IOT	SNN is more resilient to adversarial samples and has 9 higher detection accuracy over FNN. However, both models perform well under 50 for adversarial samples with and without normalized features. BIM generated samples have the most impact in reducing detection accuray of both models within the range of $15 - 25$	Accuracy, Precision, Recall, F1-score, Cohen Cappa Score, MC Coefficient

Through further analysis under CICIDS-2018, IDS-2017 and UNB-CA Botnet along with CTU-13, Apruzzese *et al.* test the efficiency of ML-IDS, such as RF, MLP, AB, BT and KNN, as botnet detectors against adversarial attacks [137]. Under the CTU-13 dataset, ensemble methods, such as RF, AdaBoost and Bagging Trees stand out in non-adversarial

TABLE VII
ADVERSARIAL SAMPLES GENERATED BY GENERATIVE MODELS TO ATTACK ML-IDS [PACKETS – OFFENSIVE]

Title	AML Schemes	ML / DL Classifier	Dataset	Result	Evaluation Metrics
Yang et al. [84]	C&W, ZOO, WGAN	DNN	NSL-KDD	ZOO algorithm and WGAN produce adversarial samples that drop the detection accuracy from 89 to 53.7 and 56 : 7 respectively.	Accuracy, Precision, Recall, F1-score, False Alarm
Yan et al. [44]	WGAN-GP	CNN	KDDcup99	Detection accuracy decreases from 97.3 to 47.6 after adversarial samples were introduced in the test set	Accuracy
Wu et al. [142]	Q-learning	DT, CNN	Botnet [143] + original [144] samples	Manual feature extraction reduces the evasion rate of adversarial samples in DT to 10. Automatic feature extraction in CNN develops an evasion rate of adversarial samples between 35 and 50.	Evasion Rate (FNR of adversarial samples)
Ma et al. [56]	AESMOTE in reinforcement learning architecture	Q-function	NSL-KDD	No evasion rate performance provided. Instead F1 score of ML-IDS on using AESMOTE generated samples is provided. F1 score on samples generated by AESMOTE is 83 while those generated by AE-RL is 78	Accuracy, Precision, Recall, F1-score, Generation Time
Sweet et al. [40]	WGAN-GP, WGAN-GPMI	-	2017 National Collegiate Penetration Testing Competition (CPTC)	The samples generated by the models are studied to view if distribution of these samples match original samples distribution. WGAN-GPMI produces samples which were 83 similar to original sample distribution. Additionally they are studied to understand what features are targeted by these models to generate the samples.	Conditional Entropy
Usama et al. [41]	GAN	DNN, RF, DT, LR, KNN, SVM, LR, NB	KDDcup99	DNN achieves a detection accuracy of 89 on original samples. After the adversarial samples were added to the test set, the detection results drop down to 56.5. After adversarial training, model detection accuracy ramps up to 84.3.	Accuracy, Precision, Recall, F1-score
Lin et al. [8]	IDSGAN	MLP, RF, DT, LR, KNN, SVM, LR, NB	NSL-KDD	The adversarial DoS samples achieve 82.7 and 1.56 DR and ADR respectively on an MLP. The EIR for these samples when tested on MLP are 98.11.	Original Detection Rate, Adversarial Detection Rate, Evasion Increase Rate
Shu et al. [145]	GAN-AAL	Gradient based DT	CICIDS 2017	The generated samples have a 99.73 success rate of evading detection	Adversarial Attack Success Rate

cases whereas in the presence of adversarial samples, these models suffer from low recall performance, which is on average 55%. Furthermore, as reported by this study, even feature removal as a defensive strategy will result in recall degradation on an average of 14% under adversarial samples.

Peng *et al.* [42] improve the boundary-based approach to the generation of adversarial samples by applying Zeroth Order Optimization (ZOO) algorithm [79] to continuous features of the network and improving the search process for such samples through random walk to search for perturbations for discrete features. By limiting the focus to DoS samples in public datasets such as KDDcup99 and CICIDS-2017, unlike FGSM and JSMA, a black box attack predicts the outputs to recalculate the number of features and their corresponding values to mislead the detector in considering this adversarial sample as normal traffic. Experimental results reported by Peng *et al.* demonstrate over 99% misclassification impact.

XSS attacks are commonly known in the Web development environment, and they add HTML and JavaScript code in certain input areas on the website to extract user information. Zhang *et al.* leverage a Monte Carlo Tree Search (MCTS) algorithm as a black box attack to generate adversarial samples for an XSS attack [138]. A set of bypassing rules are provided to ensure that the original samples retain their functionality after adding the perturbations. Some of these rules are as follows but not limited to: hexadecimal encoding, URL encoding and invalid character insertion. Under the CICIDS-2017 dataset, only the XSS attack and normal instances are considered for training and testing the generators and detectors. The detector used in these experiments is the discriminator network of a GAN. The MCTS algorithm assigns a reward for every sample that is undetected by the detector. MCTS proves to improve the effectiveness of FGSM with 6% lower True-Positive rate (TPR) in deceiving LSTM and XGBoost.

Ayub *et al.* [139] use an MLP network as an IDS and present the effectiveness of building an IDS with neural networks under the CICIDS-2017 and TrabID 2017 datasets. Similar to the previous studies, the study uses the JSMA technique to build adversarial samples. The samples are generated based on the model's parameters alone without accessing the model

TABLE VIII
ADVERSARIAL MODELS AS ML-IDS [PACKETS – DEFENSIVE]

Title	ML-IDS	Dataset	Result	Evaluation Metrics
Wang et al. [130]	WGAN-GP	KDDcup99	The trained discriminator of WGAN-GP achieves 88.2 and 80.8 detection accuracy respectively for binary and multi-class classification of all samples	Accuracy, FPR, Detection Rate
Mohammadi et al. [131]	GAN	NSL-KDD	The discriminator of the GAN which is trained on generated samples alone is able to accurately predict labels of all samples in the KDDTest+ set with 91.3 accuracy.	Accuracy
Ferdowsi et al. [152]	GAN	Daily activity recognition	Distributed GAN based IDS has achieves a 15 and 20 increase when compared to a centralized GAN IDS and a standalone IDS on each IOTD respectively	Accuracy, FPR, Precision
Xia et al. [153]	Deep deterministic policy agent algorithm	NSL-KDD	The proposed algorithm is to be tested on the NSL-KDD dataset as a future work	Accuracy, Reward
Yang et al. [129]	SAVER-DNN	NSL-KDD, UNSW-NB15	Achieve the best classification performance on KDDTest+ and KDDTest-21 with 89.36 and 80.3 accuracy respectively. On the UNSW-NB15 test the detector achieves the top accuracy with 93.	Accuracy, Precision, Recall. F1-score, FPR, TNR, G-Mean, ROC, AUC
Hara et al. [154]	Adversarial Auto Encoder	NSL-KDD	By using more than 1 of the labeled data the model can achieve better results than a trained DNN. It achieves 82.78 while DNN achieves only 75 when both models are trained on 10 of the data.	Accuracy, FNR, FPR

structure or the training set. These samples develop an evasion attack that misleads the MLP-based IDS, which in turn reduces the IDS detection capabilities between $20 - 30\%$.

Piplai *et al.* [140] also employ the detection component of a trained GAN as an ML-IDS, similar to [138] and evaluate its performance of generating adversarial samples against the detection capabilities of the discriminator. Thus, by training both models asynchronously, a generative model is formed alongside a discriminator that can detect such adversarial samples. This detection model is trained on adversarial samples along with the original attack and non-attack samples on the test split of the IEEE BigData 2019 Cup:Suspicious Network Event Recognition challenge [148]. However, when FGSM is applied to inject more perturbations to original data to further mislead the trained discriminator, the performance of the trained discriminator decreases. Even after removing the top 2 features, which are required by adversarial samples, the attack leads to a 60% success rate, and if all features are altered (as is normally done by FGSM), 96% success rate is met. Thus, even GAN-based IDS is susceptible to adversarial attacks.

Brute Force Attack Method (BFAM) is an improved technique that employs a black box approach to develop adversarial samples, and can evade the detection by ML-IDS [69]. Instead of using a neural network such as a GAN to generate samples, Zhang *et al.* search the list of features and keep adding perturbations to each feature with some magnitude of α. A confidence score is used to determine whether the perturbations need to be added to the other features. If the confidence score has increased for this target label, the adversarial sample is deemed ideal for testing, otherwise the process continues with the next feature. Having Adversarial Detection Rate (ADR) and Original Detection Rate (ODR) as the performance metrics, BFAM achieves the lowest ADR on DoS, R2L and U2R attacks for all models dropping as low as 0.13% whereas this technique is outperformed by GAN in terms of ADR for Probe attacks.

Ibitoye *et al.* [141] compare Feedforward Neural Network (FNN) and Self Normalizing Neural Network (SNN) to build IDS for an IOT network under the BOT-IOT dataset [149]. Obtaining the adversarial samples under generation techniques such as FGSM, Basic Iteration Method (BIM) [9] and Projected Gradient Descent (PGD) [78], it is concluded that an FNN outperforms an SNN on original samples. However, when exposed to adversarial samples, the accuracy of SNN improves, significantly. Normalizing the features of the samples is shown to increase the performance of both models with original samples whereas the performance for both models under adversarial samples is well under 50% detection accuracy.

Yang *et al.* [84] emphasize the viability of assuming black box attacks instead of white box ones (e.g., FGSM) which require a priori knowledge of the target system and access to its gradients. By using a DNN as an IDS, adversarial samples here are generated by either of the three approaches: 1) A substitute model (similar to target model but not used as an IDS) and samples are generated by the C&W technique, 2) A Zeroth order optimization (ZOO) [79] algorithm, and 3) WGAN.

Both ZOO and WGAN models are black box approaches, and have been proven to be effective black box approaches in the generation of adversarial samples. However, both models require intensive computation and a large number of queries

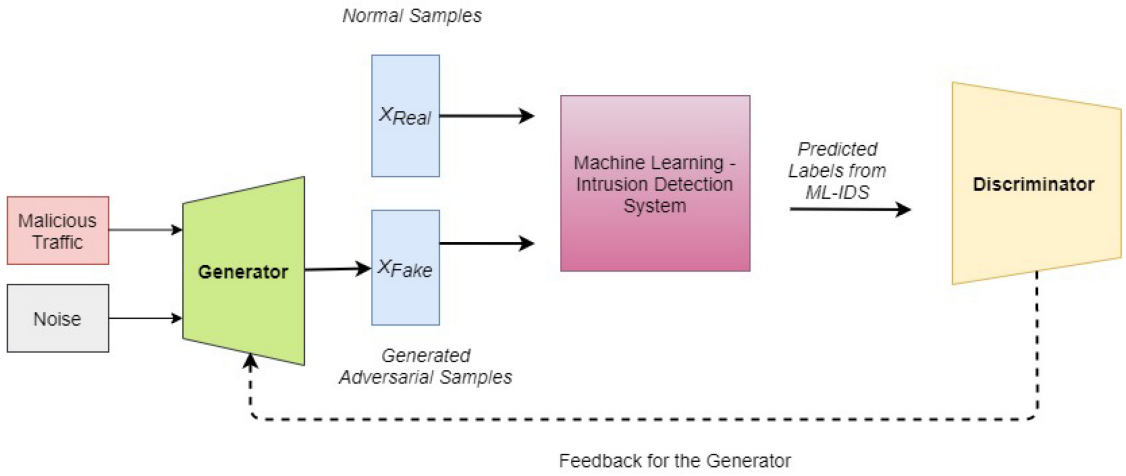

Fig. 10. IDSGAN architecture for adversarial intrusion attack sample generation proposed by Lin *et al.* [8]. It has a similar structure to the architecture proposed by Usama *et al.* [41] where an ML-IDS predictions are used as well to improve generation capabilities.

to generate the optimal adversarial samples which may not be feasible in real time.

As described earlier, using a generative model to generate adversarial samples instead of an adversarial technique is also an effective strategy. Yan *et al.* [44] use a WGAN with gradient penalty [20] to generate such adversarial samples for DoS attacks, i.e., DoS-WGAN-GP. The model can generate adversarial samples from the original samples with such efficiency that it can reduce the detection performance of a CNN-based IDS from 97.3% to 47.6%. The authors suggest that using such a generative model can help to further understand the features that are most commonly attacked by such models in the pursuit of generating adversarial samples. Sweet *et al.* [40] use WGAN-GP to generate adversarial samples based on NIDS attack samples with the intention of understanding the features that are altered by GAN-based models.

A GAN is trained similarly to the training of a Q-learning model which builds on rewards and penalties. Thus, Wu *et al.* [142] use a Q-learning based architecture to alter the network traffic and generate adversarial samples for a botnet flow attack as a black box attack. With botnet and normal samples extracted from Malware Capture Facility Project [143] and IOST 2010 dataset [144], respectively, the rewards / penalty is assigned solely based on the output of two types of ML-IDSs: a DT and a CNN. The DT gets trained on manually selected network features, thus the generated adversarial samples only lead to an 10% evasion rate with it. However, the CNN automatically picks the features, after the attack, the adversarial sample evasion rate is between 35% and 50%.

The architecture proposed by Ma and Shi [56] functionalities. First, it produces adversarial samples to evade detection of ML-IDS. Second, Ma and Shi [56] propose a reinforcement learning-based approach towards not only generating adversarial samples but also training the ML-IDS. Here, the Adversarial Reinforcement Learning (AE-RL) [55] architecture is used with an anomaly detector (Q-function in this case) that can detect malicious samples among normal traffic. The sampling process is referred to as AESMOTE as it generates adversarial samples of attacks based on the SMOTE algorithm [150], and picks the important features to learn from so that the generated samples can capture the main attributes of the original samples. AE-RL could achieve an F1-score of 78% but with AESMOTE the performance of the system rises to 82%.

Usama *et al.* [41] study altering the non-functional features of network traffic to generate adversarial samples, thus maintaining the functional behaviour of the network. They provide two contributions. When a GAN-based approach adds perturbations to the non-functional features of the network traffic, the perturbations added to the non-functional features do not hamper the functional behaviour of the network and ensure that adversarial sample is not detected as an attack. When a GAN-based adversarial training adds the adversarial samples to the training set of the ML-IDS, the detection accuracy of these trained models (DNN, RF and SVM) increases significantly.

The developers of IDSGAN (whose architecture is shown as Fig. 10) [8] also propose a WGAN-based solution to generate adversarial samples for intrusion attacks on ML-IDS. In the WGAN-based solution, the discriminator receives adversarial attack samples and network samples along with predicted labels from an ML-IDS. Thus, the discriminator can mimic an ML-IDS and improve the adversarial generation capability of the generator to be tailored specifically for that particular ML-IDS. To test the adversarial samples, Detection Rate (DR), Adversarial Detection Rate (ADR) and Evasion Increase Rate (EIR) metrics are used in the original work. It is worth to note that one should aim for low DR and ADR and a high EIR. Under the NSL-KDD dataset, it has been shown that MLP and SVM yield to more than 80% detection rates against DoS, and less than 5% detection rates against grouped U2R and R2L attacks. The corresponding ADR performances of the classifiers are significantly less than 2% for all of these attacks.

Shu *et al.* [145] use a GAN architecture, which is referred to as Gen-AAL, to construct adversarial samples in an active

learning environment. Under the CICIDS 2017 dataset and partially labelled training data [151], labeled training samples are provided to a model in order for it to learn important features that can help to classify the samples under a class. Iteratively, portions of the unlabeled set are then sent to the trained classifier to be classified and assigned a label. These samples are then merged into the labeled dataset to form a new dataset, based on which, the classifier is retrained, and the process continues accordingly. To train the Gen-AAL model in an active learning environment, the generator of the GAN is replaced by a VAE network. The discriminator is a Fully Connected Neural Network (FNN) that acts as a substitute IDS (S-IDS) during training. Once the VAE and S-IDS are trained, the S-IDS is replaced by an ML-IDS for testing, which in this case is a gradient-based DT. Research reports that the use of an S-IDS coupled with an active learning approach to train the generator model improves the capacity of the generator to generate effective adversarial samples.

2) Defensive Perspective: Research on intrusion detection is not limited to using GANs to generate adversarial samples but instead extends its usage towards building classification models as alternatives for ML-IDS algorithms such as RF. Wang and Wang [130] use a variant of GAN called Wasserstein GAN (WGAN-GP) as an IDS and prove that it is a viable alternative to the previously used ML-IDS.

Mohammadi and Sabokrou [131] also use a GAN to act as an IDS. However, unlike Wang and Wang [130], a semi-supervised approach is followed to generate adversarial samples and train the discriminator component of the GAN. To train the GAN in a semi-supervisied setting, the generator is provided with normal samples from the NSL-KDD dataset to generate adversarial samples which are labeled as attack samples. These generated attack samples along with the normal traffic flow from the dataset are provided to the discriminator for training.

Ferdowsi and Saad [152] propose a distributed GAN approach for each IoT device in an IoT network to act as an IDS. In order to maintain users privacy and ensure a robust IDS is available for the network, a centralized generator feeds malicious data to the distributed discriminators that act as IDS on each IoT device. With this architecture each discriminator on the IoT device can be trained to detect internal (on the device) and external (on neighbouring device) attacks by providing the correct loss to the discriminator that will enable to generate samples for multiple internal and external malicious samples.

Similar to GAN, an agent of deep reinforcement learning (an attack generator) can play an adversarial game against the environment, where the reward is proportional to the number of misclassified malicious samples. With this in mind, Xia *et al.* [153] use a reinforcement learning-based approach to generate adversarial samples and train a detection system: A generator agent which is a Deep Q-learning-based adversarial sample generator, and a detection agent which is a deep deterministic policy agent algorithm that acts as a detection model.

SAVAER-DNN is an improvement to the current ML-IDS concept as per [129]. It shows significant improvement in detection of malicious traffic that is known, unknown as well as ones that have low frequency. During the training phase, an Encoder (GAN generator), Decoder (VAE Decoder) and Discriminator are trained. Furthermore, a WGAN-GP is used instead of a GAN to generate fake or malicious traffic. The encoder learns about the features that adversarial samples target by generating samples that try to deceive the discriminator. The actual classes are also added to the training of the decoder and discriminator to ensure robustness against known and unknown attacks via supervised learning. These generated malicious samples along with their labels are fed to the decoder. The decoder is used to generate augmented samples which are combined with original samples to create a synthetic dataset. During the inference phase, the weights of the encoder are used to initialize a DNN to act as the new and improved IDS.

Hara and Shiomoto [154] test the efficiency of generative models, such as Auto Encoders in a semi-supervised training environment using an Adversarial Auto Encoder to build an IDS. The encoder of the Auto Encoder model acts as a generator (similar to a GAN) and feeds encoded samples to two discriminators, as well as the decoder of the Auto Encoder. It provides two outputs each of which is fed to a discriminator and the decoder. The discriminators used to help the encoder generate adversarial samples. One discriminator is for categorical data (labels) whereas the other is for discrete data (network features). The two discriminators, the decoder and the encoder, are all trained together. Similar to [153], the encoder is used as an IDS but the output for the categorical discriminator serves to evaluate its performance on the NSL-KDD dataset. Upon training the model on different number of labeled data, the conclusion is that simply by using more than 1% of the labeled data, an Adversarial Auto Encoder can outperform a trained DNN.

C. Attacks and Defenses in Application Layer

Mobile usage around the world continues to grow at a rapid pace. Most mobile and tablet devices use the Android OS. Since a large number of Android devices are used around the world, most adversaries are keen on targeting such devices [155], [156] to inflict maximum damage not just to the targeted device but to the network that the device is connected onto as well as to other devices on the network. Machine Learning techniques have proven to be effective solutions to detect such attacks [54], [97], [157]–[159]. The current literature is developed around generating attacks and defences for the Android operating system largely due to the availability of Android malware datasets [54], [97] and the sheer number of users of Android devices (74% as of July 2020).[2] We refer to these Machine Learning detectors as ML-MMD (Machine Learning - Mobile Malware Detector), and consider Deep Learning based approaches which comprise of neural network architectures as part of the larger ML community [160]. Hence, we denote the detection models built with Deep Learning as ML-MMD as well.

Drebin is one of the earliest benchmark datasets for ML-MD. In the research presented in [54], the application of a

[2]Available at https://gs.statcounter.com/os-market-share/mobile/worldwide

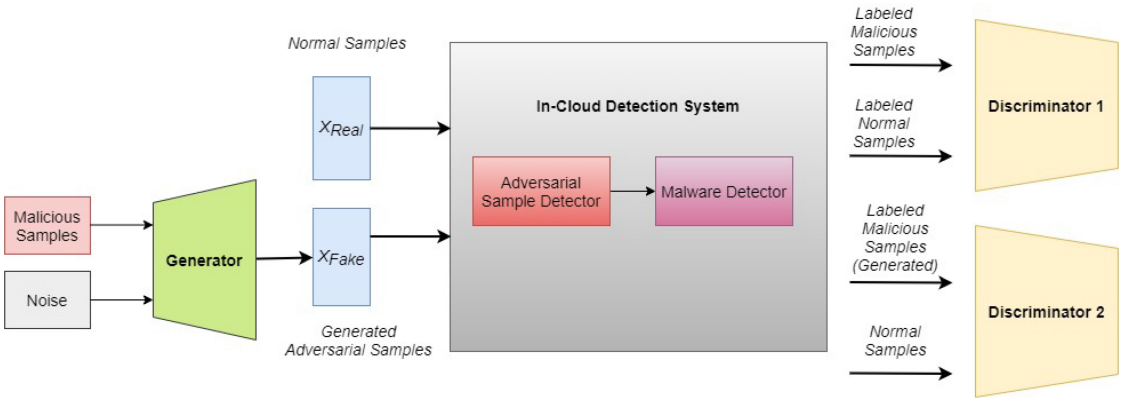

Fig. 11. E-MalGAN architecture for adversarial malware sample generation proposed by Li *et al.* [43].

Linear SVM shows the efficiency of ML algorithms towards detecting malware inputs. Additionally, Markov chains are also used to track the API calls for all samples which are provided to Random Forest (RF) and K-Nearest Neighbour (KNN) algorithms to detect malicious samples [97]. However, both approaches do not deal with malware samples that are generated by adversarial techniques or models with the aim of misleading the classifier so to detect the sample as benign. These solutions provide remarkable performance under benchmark datasets (e.g., Drebin). However, they are unable to detect malware samples that have undergone some small changes at the feature level. Thus, pressing the need for explainable models that can display the important features that are prone to be targeted by attackers [161]. By being unable to detect such samples, these ML-MMD solutions show fragility against patterns generated by adversarial techniques and models. Thus, in this section, we present an overview of the latest works conducted in the field of Mobile Malware detection but focus on the impact of Adversarial Machine Learning in this field. Furthermore, the offensive approaches and defensive methodologies are categorised in different subsections and listed in Table IX and Table X respectively.

1) Offensive Perspective: Chen *et al.* [157] is one of the first to generate adversarial samples by producing a data poisoning attack by manually altering certain features of the Android application. Data poisoning attacks lead to some aimed errors in the ML models used in prediction or detection [167]–[170]. Alternatively, Hu and Tan [52] use the MalGAN model to generate samples and attack Android MDs through a black box attack. Both studies use SVM, Random Forest and KNN to detect such malware samples but are unable to achieve results as promising as those achieve on non-adversarial malware samples. Similarly, in [171], the authors use JSMA to create adversarial samples on Drebin dataset and use a DNN as a substitute model to alter the adversarial samples.

Shahpasand *et al.* [165] use a GAN (not MalGAN) to generate adversarial malware samples from the Drebin dataset, and test the impact of using different number of parameters in the GAN architecture by setting a threshold for the number of features to which perturbations can be added to generate the adversarial samples. Further tests on the performance of RF, SVM, DNN and LR classifiers on this dataset show that these models achieve on average $97-98\%$ accuracy on non-adversarial samples. Besides, the effectiveness of generating adversarial samples in terms of evasion rate, i.e., number of malware samples that went undetected over total number of malware samples show that 99% evasion rate on DNN and LR can be achieved. RF is proved to be the most effective since its highest evasion rate remains at 52%.

Li *et al.* [163] review mobile malware attacks and attack detection systems, and report that Android MDs are targeted by either static, dynamic or adversarial attacks. Static and dynamic attacks go hand in hand and are used as a combination to make the attack more effective. They also need prior knowledge of the features to alter (such as application's API calls). Adversarial attacks on the other hand can be evasion attacks (black box) or data poisoning attacks (white box). The ML-MMD proposed in the study builds on SVM based upon on its performance on the Drebin dataset as shown in [54].

According to Li *et al.* [43], firewall systems used in cloud based environments may have two different detectors: one for adversarial samples and one for Android malware. Most generation models generate adversarial samples in a black box attack format as they assume to have no prior knowledge of the target system but only its predictions. Thus, current adversarial sample generators for Android malware such as MalGAN can avoid detection from MMD but not adversarial malware detector. To this end, E-MalGAN, is a bi-objective GAN that aims to deceive both adversarial samples and malware detectors which are part of one system called in-cloud detection system. Unlike other GANs, it has two discriminators (one representing the adversarial sample detector and the other is the malware detector) where one of the two generates samples in a black box attack format. Similar to MalGAN, the generator receives data from noise as well as malware samples. The combination of CNN as adversarial detector and AdaBoost as malware detector prove to be the most robust with a lower Effective Generation Rate (EGR).

Inspired by the success of utilizing SVM on the Drebin [54] dataset and using Markov chains to represent API calls on the MaMaDroid dataset, Chen *et al.* [99] apply JSMA and C&W techniques to generate adversarial malware samples for both datasets. With the addition of samples from VirusShare and

TABLE IX
ADVERSARIAL MOBILE MALWARE GENERATION [MALWARE – OFFENSIVE]

Title	AML Schemes	Target ML / DL Classifiers	Dataset	Result	Evaluation Metrics
Abusnaina et al. [162]	C&W, Deep Fool, ElasticNet, FGSM, JSMA, MIM, PGD, VAM, GEA	CNN	Android and IoT malware mixed with OpenWRT's firmware	C&W can mislead CNN 100 and GEA can maintain malware lethality	Accuracy, FNR, FPR, Misclassification Rate, Average Number of Changed Features, Crafting Time
Ceshin et al. [159]	Adding benign binaries' strings, Changing malicious binary headers	LightGBM, Non-Negative MalConv, MalConv	Malware Binaries Provided by Microsoft	Reduce detection rate by 20, where the classifiers are extremely biased to malware	Accuracy, Confidence
Chen et al. [157]	Manually Altering Features	SVM, RF, KNN	Pwnzen Infotech Inc., COntagio, Zhou et al., Arp et al.	FNR: 80.05 for Off-the-shelf detection tools.	Accuracy, FPR
Chen et al. [99]	JSMA and C&W	RF, SVM, DNN	Drebin and MaMaDroid with samples from VirusShare	Evasion rate : C&W on Drebin black box attack: $99-100$, C&W on MaMadroid black box attack: 100	F1-score, Evasion Rate, Distortion
Li et al. [163]	Evasion and Data Poisoning	SVM	Drebin [54]	-	-
Li et al. [43]	E-MalGAN	Multiple Scenarios of ML-MMD	Benign: Tencent MyApp and AndroZoo, Malware: VirusShare and Contagio	Average Effective Generative Rate of E-MalGAN: 95.02, MalGAN: 27.02	Effective Generation Rate
Rosenberg et al. [49]	Score based and Decision Based Attack method	LSTM, Deep LSTM, GRU, CNN, LR, RF, SVM, GB Tree	Malware dataset generated in [164]	Attack effectivenesses are ranging from 41.47 to 63.63, higher than previous works.	Attack Effectiveness, Additional API Calls
Shahpasand et al. [165]	GAN	RF,SVM, DNN, LR	Drebin	Evasion Rate of 99 on DNN and LR	Attack Success Rate
Suciu et al. [166]	FGM, Slack FGM	MalConv	VirusTotal, Reversing Labs, and FireEye	Evasion Success Rate of 71 using FGM	Evasion Success Rate

TABLE X
ADVERSARIAL MOBILE MALWARE DETECTION [MALWARE – DEFENSIVE]

Title	ML / DL Defense Approaches	AML Schemes	Dataset	Result	Evaluation Metrics
Khoda et al. [176]	DNN	JSMA	Drebin	Accuracy after retraining on samples picked by 1) Cluster : 82 2) SVM RBF kernel : 86	Accuracy, Precision, Recall, TNR, G-Mean
Abusnaina et al. [162]	CNN	C&W, Deep Fool, ElasticNet, FGSM, JSMA, MIM, PGD, VAM, GEA	Android and IoT malware mixed with OpenWRT's firmware	C&W can mislead CNN 100 and GEA can maintain malware lethality	Accuracy, FNR, FPR, Misclassification Rate, Average Number of Changed Features, Crafting Time
Wang et al. [160]	DNN, DNN with dropout, Adversarial Trainign, Randomized Feature Nullification	JSMA	Windows Audit Logs Collected by the authors, (MNIST, CIFAR-10)	Adversarial Training with RFN achieves 94.81 accuracy and 68.77 Resistance score on the malware dataset	Accuracy, Resistance
Chen et al. [156]	SecCLS, SecENS	Brute-Force, Anonymous, Well-Crafted	Apps from Comodo Cloud Security Center	Detection accuracy can maintain 81.06 to 91.68	Accuracy, TPR, F1-score

APKPure datasets to the Drebin dataset, an original and surrogate version of Drebin is formed. A similar step is taken for the MaMaDroid dataset. This step of combining datasets to create original and surrogate versions is to ensure the number of samples remain the same for the following attack scenarios: 1) F: adversaries comprehend the way to extract features from APK files; 2) FB: based on scenario F, adversaries can further query the target classifier obtaining the prediction of adversarial samples; 3) FT: based on scenario F, adversarial can also acquire the training set; 4) FTB: adversaries have full knowledge of the target classifier (i.e., feature set, training set, and the prediction of adversarial samples). The "B" in these notations is to denote a black box attack. The same study achieves 100% evasion rate for RF, SVM and DNN with the C&W attack technique and FB and FTB attack scenarios under MaMaDroid dataset. For Drebin dataset the C&W technique achieves $99-100\%$ evasion rate for all four classifiers with FB and FTB attack scenarios. JSMA technique

perform well in attack scenarios when the training data is accessible.

There are also studies that apply the Brute Force Attack Method (BFAM) explained in Section IV-B1 and test the generated samples against LR, DT, RF, MLP and NB classifiers. Zhang *et al.* [69] observe that the Adversarial Detection Rate (ADR) for BFAM-generated samples are much lower than those generated by a GAN and the original malware samples of the dataset.

Abusnaina *et al.* [162] perform a comprehensive study on the adversarial sample generation of IoT malware, experimenting with almost all existing adversarial sample generation techniques and propose Graph Embedding and Augmentation (GEA), which guarantees that the outputs are realistic and able to perform original tasks on target host machines. By adopting the IoT binaries collected by Alasmary *et al.* [172] as malware and combining benign firmware from OpenWRT, when the dataset is fed into off-the-shelf adversarial sample generation techniques, including C&W, Deep Fool, Elastic Net, FGSM, JSMA, Momentum Iterative Method (MIM), Projected Gradient Descent (PGD), and Virtual Adversarial Method (VAM), C&W can achieve 100% in terms of misleading the target classifier (CNN). Although the existing methods successfully evade the target IDS (CNN), it is possible for them to generate adversarial samples that do not function well on target machines. With this in mind, the GEA method emerges to overcome such an issue by introducing the condition that only executes the applicable codes of the original samples and skips the codes of the generated samples.

The adversarial sample generation generally involves probing, leading the security policies to be more aggressive; thereby, Rosenberg *et al.* [49] propose a black box adversarial scheme intending to minimize the number of probing times by slicing the system calls of a malware. For each segment, it repeatedly inserts the perturbations (benign system calls) trained by SeqGAN until the target detector misclassifies the malware and eliminates redundant perturbations to maintain the malware size. The proposed method can maintain O(log n) time complexity whereas the linear search can be up to O(n). As the knowledge / information obtained from target systems is different, Rosenberg *et al.* further develop the decision-based attack using success / failure as feedback and score-based attack exploiting the confidence of the target system.

Practical research by Ceschin *et al.* [159] on malware evasion involve bypassing ML-based malware detection schemes (i.e., Light Gradient Boosting Machine (LightGBM) [173], NonNegMalConv [174], MalConv [175]) without breaking the original functionalities of malware. Suciu *et al.* [166] apply FGM to generate adversarial noise which will be appended to Portable Executable (PE) files. This strategy achieves 71% Evasion Success Rate (ESR) when MalConv is used as a detection classifier while appending random noise and benign content only achieve 0% and 1% ESR respectively.

2) Defensive Perspective: Khoda *et al.* [176] propose adversarial training as an instrument to defend against adversarial Android malware attacks through a DNN-based binary classifier against JSMA-generated adversarial samples under the Drebin [54] dataset. Without the adversarial samples, the DNN can achieve an accuracy of 98%. With he addition of adversarial samples, the performance drops to 71%. Two retraining alternatives are proposed to choose adversarial samples to train the network to boost the adversarial attack detection of this model: 1) Cluster-based approach where n samples are chosen with respect to the minimum distance to cluster center, 2) An SVM with RBF kernel is trained on the dataset to learn the probability distribution of the samples, and n adversarial samples with the least probability are chosen.

To build robust malware detectors, one needs to be aware of the type of attacks that such detectors face in mobile networks. Abusnaina *et al.* investigate the deep learning networks and control flow graph-based approaches in malware detection, and report that the graph embedding and augmentation method can achieve total misclassification of malware attacks on IoT devices [162]. Yuan *et al.* [177] however propose a different strategy where each malware detector constitutes a lightweight unique model that is trained on device based on broad_learning [178]. These models consist of two types of DNN: 1) On-device fully trained broad learning training and detection algorithm (BL-TDA) and 2) On-device incremental learning training and detection algorithm (IL-TDA). They work together as one system named Broad Learning Android Malware Detector (BL-AMD). The IL-TDA is retrained when the overall detection performance drops by a certain percentage. Under a joint dataset that is comprised of benign samples from Tencent MyApp and AndroZoo [179], [180], and malicious samples from VirusShare and Contagio, BL-AMD outperforms the off-the-shelf algorithms ML-MMD algorithms such as Logistic Regression and SVM on both non-adversarial and adversarial sample detection. However, BL-AMD does not perform better than CNN-based MLP.

Wang *et al.* [160] propose replacing the input layer of a DNN with a nullification layer, which randomly drops the input features according to Gaussian distribution. Compared to other defense mechanisms, such as adding dropout layers and adversarial training, the random feature nullification (RFN) method slightly reduces the accuracy. To have further improved performance, adversarial training is combined with RFN. Such a combination not only increases the resistance score but also enhances the detection accuracy.

Chen *et al.* [156] introduce the SecureDroid framework comprised of two components, i.e., SecCLS and SecENS. Instead of randomly dropping out features, SecCLS takes feature importance and altering difficulty into consideration. SecENS is an ensemble method while each base estimator has a distinct feature set determined in the training process. With the test data obtained from Comodo Anti-Malware Database [181], and Brute-force, Anonymous, and Well-crafted methods as the adversarial sample generators, the SecureDroid framework leads to a decrease of 5% to 15% in the accuracy of an attack-free scenario under various settings while also outperforming other defense techniques (i.e., feature evenness [182], classifier retraining [183], and feature reduction [184]).

TABLE XI
OPEN ISSUES AND CHALLENGES

Perspective	Open Issues and Challenges	Long/Short Term
Offensive	Speed and Transferability	Short Term
	Effectiveness of Adversarial Samples	Short Term
	Better Representation of Reinforcement Learning	Long Term
	Difficulty of Training GANs	Long Term
Defensive	Datasets	Long Term
	Customizing GANs for Network IDS	Short Term

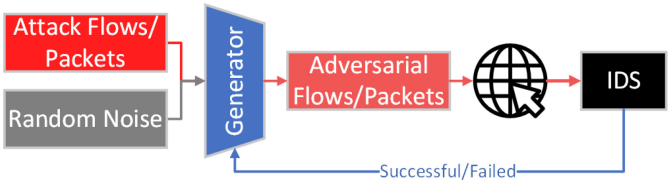

Fig. 12. The use of AML by adversaries.

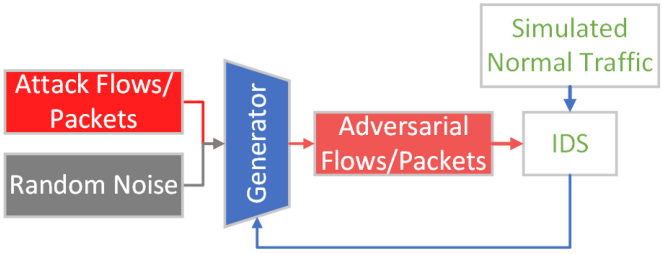

Fig. 13. Training process of generators.

V. LESSONS LEARNED, OPEN ISSUES AND CHALLENGES

The anticipation and defense strategies reviewed in Section IV provide an overview of the current impact of AML in mobile settings. The adversarial procedure of AML can be exploited by both attackers and defenders. Thus, this section discusses and summarizes the open issues and challenges from the standpoints of attack / threat anticipation and defence.

It is necessary to clarify the assumptions and tackle potential attack and defense scenarios. An adversary that has acquired the tools to effectively compromise target hosts / networks by also pursuing zero-day attacks, aims to bypass firewalls / IDSs and other security measures so to paralyze target network or acquire root privileges for the hosts [185]. Since IDSs are generally deployed at edges of networks, unless boundary routers or IDSs themselves are compromised, it is not realistic to assume that the regular traffic can be obtained by attackers, even if they control a host within the target domain [186]. Nonetheless, adversaries can perceive the status of their attacks (such as successful, failed, or falling into honeypot), which can be considered as a useful indicator to train their adversarial model [187]. On the other hand, a defender can conveniently access benign traffic but cannot enumerate all possible attacks while trying to protect essential assets from the known and impactful attacks. Consequently, it is intrinsically asymmetric for intruders and defenders to leverage AML. We summarize the open issues and challenges in Table XI and list whether they can be solved in short or long term; for instance, challenges related to improving adversarial machine learning or creating a novel dataset platforms will take long time to be explored, while challenges regarding application (e.g., customizing GAN for NIDS) perspective can be addressed more recently.

A. Attacker Perspectives

Fig. 12 illustrates an abstract view of the general use of AML by adversaries. An adversarial sample generator takes attack flows / packets produced by malicious tools and random noise as inputs, extracts features from intrusion flows, generates adversarial samples, reforms the flows / packets by controlling network characteristics (according to the adversarial samples), and finally, sends them via Internet towards target IDS which decides whether to pass the adversarial flows. For attackers, the goal is to train the generator to create regular-traffic-like flows according to IDS feedback and utilize the generator as a flow re-shaping unit. With these in mind, the following open issues call for further investigation:

- *Speed and Transferability:* IDS only observes and reports suspicious events without taking any defensive actions whereas IPS, firewall, and security experts are generally coupled with IDS, which could terminate hackers' network access once threats are observed [188]. Thus, if a generator is trained solely relying on the feedback from the target IDS, the long time probing will raise more significant alarms than regular attacks [189]. To the our best knowledge, a few adversarial sample studies discuss the convergent time of training and testing processes, which, nonetheless, is indispensable for attackers. One possible solution is to let generators start from nominal feature values such as low network speed and frequency and increase them gradually whereas the negative feedback from IDS is necessary for generators to learn. A more practical solution is to have a trained generator to bypass the target IDS. As Fig. 13 illustrates, a simulated network and a white box IDS can be established. An adversary can train a generator in such an environment and transfer it to intrude target networks. Szegedy *et al.* report that exploration of the impact of adversarial samples on different neural network structures uncovers the following phenomenon: although the adversarial samples are generated by one structure, they can also increase the error rate of other structures of neural networks from 20s% to the order of 98%. Papernot *et al.* [21] take one step further and measure this cross-technique transferability for DNN, SVM, LR, DT and ensemble models. As pointed out in the same study, SVM, LR and DT share similarities in being affected by the same adversarial samples. In mobile systems, however, the transferability of these concepts regarding which features of the network flow should be changed are quite unstable. Dealing with the transferability of generated samples, having the same impact on other DNN, had been studied in [30], [84]. Yang *et al.* [84] mention that this instability could arise from the substitute model (alternative to target model with close resemblance) not being able to achieve the same level of performance as the target model. Thus, in a black

box attack on different target models, the transferability of the samples, that proved to be effective on one model, cannot be guaranteed on the other model. Understanding the issue of transferability can help future research build more generalized solutions to AML attacks on ML-IDS.

- *Effectiveness of Adversarial Samples:* Several studies point out that adversarial samples can fool machine learning algorithms successfully [190] and evade the detection capability of IDSs [191]. Nevertheless, some restrictions should be affixed to the adversarial sample generation process to ensure the outcomes are realistic for attackers to generate and lethal enough to threaten the target hosts / networks. Peng *et al.* [42] plot the comparisons between the features of original attacks and adversarial samples, which uncover that the adversarial samples can result in a dramatic increase in the cost of computer and network resources. For instance, the adversarial samples require GoldenEye to send 10,000 packets per second and 35,000 bytes of payload per second, while the number of packets per second and bytes per second of original attacks is less than 100 and 3000, respectively. Therefore, the limited resources of attackers should be considered and restrict the adversarial samples from generators. Meanwhile, as Lee and Stolfo [192] show, different types of attacks have distinguished functional features. For example, DoS is well known for its flooding ability, thereby the time-based features should not be reduced substantially. Lin *et al.* [8] explore the feasibility of not tampering with functional features. Their results indicate that adversarial samples can still evade an IDS even with unchanged functional features. To further increase the evasion rate, functional features can be customized for each attack tool, e.g., slow HTTP test, as a DoS attack, can still maintain its function with low-speed connections. Hence, it is worth to study the balance between the evasion rate and lethality for each attack.
- *Better Representation of Reinforcement Learning:* Reinforcement learning (RL) is practical to be applied as a generator in Fig. 12. RL can be modeled as Markov Decision Process (MDP), consisting of a space of environmental states, action space, transition probability, and rewards. Various researches have distinct representations of actions, states, and rewards. As an example Wu *et al.* [142] describe actions as modifications of flow samples such as changing timestamp and length of packets and states as the features extracted from network flows. Thus, further comprehensive representations of actions, states, and rewards could lead to potential future studies, such as tuning network speed and packet generation frequency. Moreover, instead of using the existing datasets, creating a simulated or experimental network environment could lead to further compelling research in this field.
- *Difficulty of Training GANs:* GANs are notoriously difficult to train. Even WGANs are unstable and require prior knowledge on the distribution of noise to be provided to train the model. Furthermore, GANs also need access to all features of the network in order to perform effectively [69], [84]. As reported in [84], one of the most challenging issues encountered by GANs is the model collapse phenomenon, which is the consequence of the generator producing the same plausible adversarial samples while the discriminator is stuck on a local minimum and unable to distinguish them. In the occurrence of such a problem, the lazy generator can only bring out similar outputs regardless of the variance of its input noise; meanwhile, the discriminator can barely learn valuable information in the subsequent iterations due to being stuck at a local minimum. This challenge is brought upon because of the min-max game played by GANs. For instance, if the discriminator outperforms the generator in the early training process, the stable loss will cause gradient vanishing problem, and thus, the system will be unable to learn from the adversarial process. Due to GAN's high sensitivity to hyperparameters, training a GAN to generate adversarial attacks may take a long time and is unstable. Moreover, since the quality of a GAN's outputs also relies on its discriminator, partial information is assumed to be known by intruders (e.g., features used by defender, the approximate scale of defender's classifier). As a result, to anticipate the impact of adversarial samples, advanced GAN models need to be developed.

B. Defender Perspectives

- *Datasets:* As Hindy *et al.* [193] report, collecting reliable and representative data is essential for IDS since biased and skewed data only lead to an impractical model. Despite the existng datasets used by the state-of-the-art AML algorithms in multiple aspects, due to the lack of real-life and recent network traces, the IDS models that are trained on benchmarks, especially anomaly-based IDS, are reported as not a commonly good fit in production environments [194]. Moreover, the imbalance problem or the lack of certain types of attacks lead to overfitting or untrainable AML, and adversarial generation techniques can only be generalized to one attack at a time [42], [138]. They have yet to be tested on multiple network attacks, such as Probe, U2R, and R2L. One common reason for not expanding the application of generation techniques to other attacks is the lack of available data. In the NSL-KDD dataset, the DoS and Probe attacks have the highest number of samples whereas U2R and R2L have very few samples. Therefore, collecting real-life networks and increasing malicious records in training sets are still required for network intrusion datasets. For attackers, many sandbox platforms provide target machines to practice and improve their threat anticipation skills, such as HackMe, Vulhub, and HackBox. However, with the fast development of virtual technologies, a few platforms provide comprehensive experimental environments for defenders to collect, generate, and label datasets; thereby, building new datasets from scratch can be highly challenging. Furthermore, as a single group or project cannot cover significant types of attacks, standardized, modular, and extensible datasets are

demanded so that different datasets can be merged and reused by researchers [193]. Without reliable datasets, neither ML-IDS nor AML-IDS can be trusted and put into production environments.

- *Customizing GANs for Network IDS:* As we discuss in Section IV-B, currently researchers use the discriminator of a GAN as IDS, while attackers may apply different adversarial sample generation techniques or models other than the GAN's generator. Thus it is worthwhile to test the generalization ability of GAN discriminators facing different types of adversarial sample generation techniques and models. There could be a rise in temporal adversarial samples which have yet to be explored in wireless and mobile settings but it could be more challenging to detect as suggested by Sweet *et al.* [40]. As the same study suggests [40], Recurrent GAN architectures could be useful in response to network attack samples since most NIDS datasets are collected as a series of packets / flows. By introducing a recurrent architecture [195], a sequence of alerts can lead to an accumulation of features, which in turn facilitates the detection for NIDS. Thus, temporal adversarial samples can boost detectors in finding associations between samples, which has not been explored in the context of wireless and mobile networks yet. Developing a GAN specifically for wireless and mobile Networks can be more efficient than using pre-trained GANs from the Computer Vision domain since the samples in a network environment contain lower dimensions in comparison to the pixel map of an image. In addition, as suggested by [41], since network applications require discrete values, another direction is envisioned be customizing GAN for a discrete domain, such as the number of TCP flags, the number of packets, and the length of packets. These generated features should be realistic and obey network protocols; otherwise the generated flow cannot be received by targets properly.

VI. Summary and Conclusion

The impact on Machine Learning (ML)-based solutions continues to grow in various fields, such as Computer Vision, Natural Language Processing and Network Security. However, as research advances with more sophisticated and better performing models in these fields, it is critical to divert the attention toward their susceptibility to attacks that are meant to degrade their performance. This survey article has primarily focused on attacks towards ML-based solutions in wireless and mobile systems. As such attacks target intrusion and malware detection systems, the majority of this survey is centered around the network security aspect of mobile systems. While there is progress in detecting and preventing such attacks from affecting the network, further investigation is required in tackling attacks by Adversarial Machine Learning (AML) algorithms and models.

Most advancements in the field of AML have been applied and tested in Computer Vision. However, applying AML to network security has started to gain interest as presented in this article, which highlights the state-of-the-art applications of AML. These works have indicated not only the impact and relevance of AML to network security, but also a growing interest in its application to make networks more secure. The survey has focused on the applications of AML to physical, network and application layers of network security from offensive and defensive standpoints. As most of the research is targeted towards intrusion detection, we have been able to provide more in-depth analysis on three fronts: 1) Generating adversarial samples through adversarial techniques; 2) Generating adversarial samples through adversarial models; and 3) Using generative models to either help existing ML-IDS or to replace these ML-ILDSs.

This survey has also studied the Mobile Malware Detection (MMD) as the usage of smartphones continues to grow around the world. We have summarized the state-of-the-art approaches towards generating and defending against such malware attacks. It is expected that this work lays the groundwork for researchers to work on the open issues and challenges that network security systems face against attacks generated by adversarial techniques and algorithms.

References

[1] L. Chiticariu, Y. Li, and F. R. Reiss, "Rule-based information extraction is dead! Long live rule-based information extraction systems!" in *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Oct. 2013, pp. 827–832. [Online]. Available: https://www.aclweb.org/anthology/D13-1079

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[3] S. Mehtab and J. Sen, "A time series analysis-based stock price prediction using machine learning and deep learning models," 2020, *arXiv:2004.11697*.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[5] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2224–2287, 3rd Quart., 2019.

[6] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10778–10787.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[8] Z. Lin, Y. Shi, and Z. Xue, "IDSGAN: Generative adversarial networks for attack generation against intrusion detection," 2018, *arxiv:1809.02077*.

[9] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," 2016, *arxiv:1611.01236*.

[10] Y. Zhang, M. Simsek, and B. Kantarci, "Empowering self-organized feature maps for AI-enabled modelling of fake task submissions to mobile crowdsensing platforms," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1334–1346, Feb. 2021.

[11] Y. Zhang, M. Simsek, and B. Kantarci, "Self organizing feature map for fake task attack modelling in mobile crowdsensing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[12] X. Chen, M. Simsek, and B. Kantarci, "Locally reconfigurable self organizing feature map for high impact malicious tasks submission in mobile crowdsensing," *Internet Things*, vol. 12, Dec. 2020, Art. no. 100297.

[13] A. I. Newaz, N. I. Haque, A. K. Sikder, M. A. Rahman, and A. S. Uluagac, "Adversarial attacks to machine learning-based smart healthcare systems," in *Proc. IEEE Global Commun. Conf. (Globecom)*, 2020, pp. 1–6.

[14] P. Vidnerová and R. Neruda, "Vulnerability of classifiers to evolutionary generated adversarial examples," *Neural Netw.*, vol. 127, pp. 168–181, Jul. 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608020301350

[15] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–11.

[16] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, vol. 2, 2014, pp. 2672–2680.

[17] C. Szegedy *et al.*, "Intriguing properties of neural networks," 2014, *arXiv:1312.6199*.

[18] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Security Privacy (EuroS P)*, 2016, pp. 372–387.

[19] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 214–223.

[20] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein GANs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5769–5779.

[21] N. Papernot, P. D. McDaniel, and I. J. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," 2016, *arxiv:1605.07277*.

[22] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2224–2287, 3rd Quart., 2019.

[23] S. Krithivasan, S. Sen, and A. Raghunathan, "Sparsity turns adversarial: Energy and latency attacks on deep neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 11, pp. 4129–4141, Nov. 2020.

[24] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2805–2824, Sep. 2019.

[25] A. S. Chivukula and W. Liu, "Adversarial deep learning models with multiple adversaries," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1066–1079, Jun. 2019.

[26] S. Otoum, B. Kantarci, and H. T. Mouftah, "A comparative study of AI-based intrusion detection techniques in critical infrastructures," *ACM Trans. Internet Technol.*, vol. 21, no. 4, p. 81, 2021.

[27] A. I. Newaz, A. K. Sikder, M. A. Rahman, and A. S. Uluagac, "A survey on security and privacy issues in modern healthcare systems: Attacks and defenses," 2020, *arXiv:2005.07359*.

[28] J. Li, Y. Liu, T. Chen, Z. Xiao, Z. Li, and J. Wang, "Adversarial attacks and defenses on cyber–physical systems: A survey," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5103–5115, Jun. 2020.

[29] E. Alhajjar, P. Maxwell, and N. D. Bastian, "Adversarial machine learning in network intrusion detection systems," Apr. 2020, *arXiv:2004.11898*.

[30] Z. Wang, "Deep learning-based intrusion detection with adversaries," *IEEE Access*, vol. 6, pp. 38367–38384, 2018.

[31] Y. Xu, B. Du, and L. Zhang, "Assessing the threat of adversarial examples on deep neural networks for remote sensing scene classification: Attacks and defenses," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 2, pp. 1604–1617, Feb. 2021.

[32] N. Martins, J. M. Cruz, T. Cruz, and P. H. Abreu, "Adversarial machine learning applied to intrusion and malware scenarios: A systematic review," *IEEE Access*, vol. 8, pp. 35403–35419, 2020.

[33] D. J. Miller, Z. Xiang, and G. Kesidis, "Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks," *Proc. IEEE*, vol. 108, no. 3, pp. 402–433, Mar. 2020.

[34] F. Olowononi, D. B. Rawat, and C. Liu, "Resilient machine learning for networked cyber physical systems: A survey for machine learning security to securing machine learning for CPS," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 1, pp. 524–552, 1st Quart., 2021.

[35] D. Park and B. Yener, "A survey on practical adversarial examples for malware classifiers," in *Proc. Reversing Offensive-Oriented Trends Symp.*, Nov. 2020, pp. 23–35.

[36] K. Sadeghi, A. Banerjee, and S. K. S. Gupta, "A system-driven taxonomy of attacks and defenses in adversarial machine learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 4, pp. 450–467, Aug. 2020.

[37] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[38] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Inf. Fusion*, vol. 64, pp. 131–148, Dec. 2020. [Online]. Available: http://dx.doi.org/10.1016/j.inffus.2020.06.014

[39] K. Ren, T. Zheng, Z. Qin, and X. Liu, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346–360, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S209580991930503X

[40] C. Sweet, S. Moskal, and S. J. Yang, "Synthetic intrusion alert generation through generative adversarial networks," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, 2019, pp. 1–6.

[41] M. Usama, M. Asim, S. Latif, J. Qadir, and A. Al-Fuqaha, "Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2019, pp. 78–83.

[42] X. Peng, W. Huang, and Z. Shi, "Adversarial attack against DoS intrusion detection: An improved boundary-based method," in *Proc. IEEE 31st Int. Conf. Tools Artif. Intell. (ICTAI)*, 2019, pp. 1288–1295.

[43] H. Li, S. Zhou, W. Yuan, J. Li, and H. Leung, "Adversarial-example attacks toward android malware detection system," *IEEE Syst. J.*, vol. 14, no. 1, pp. 653–656, Mar. 2020.

[44] Q. Yan, M. Wang, W. Huang, X. Luo, and F. R. Yu, "Automatically synthesizing DoS attack traces using generative adversarial networks," *Int. J. Mach. Learn. Cybern.*, vol. 10, pp. 3387–3396, Feb. 2019.

[45] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Security*, 2017, pp. 506–519. [Online]. Available: https://doi.org/10.1145/3052973.3053009

[46] Y. Zhao, I. Shumailov, H. Cui, X. Gao, R. Mullins, and R. Anderson, "Blackbox attacks on reinforcement learning agents using approximated temporal information," in *Proc. 50th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, 2020, pp. 16–24.

[47] J. Hang, K. Han, H. Chen, and Y. Li, "Ensemble adversarial black-box attacks against deep learning systems," *Pattern Recognit.*, vol. 101, May 2020, Art. no. 107184. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320319304844

[48] A. Kuppa, S. Grzonkowski, M. R. Asghar, and N.-A. Le-Khac, "Black box attacks on deep anomaly detectors," in *Proc. 14th Int. Conf. Availability Rel. Security*, 2019, p. 21. [Online]. Available: https://doi.org/10.1145/3339252.3339266

[49] I. Rosenberg, A. Shabtai, Y. Elovici, and L. Rokach, "Query-efficient black-box attack against sequence-based malware classifiers," in *Proc. Annu. Comput. Security Appl. Conf.*, 2020, pp. 611–626. [Online]. Available: https://doi.org/10.1145/3427228.3427230

[50] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," 2017, *arXiv:1712.04248*.

[51] M. Cheng, T. Le, P. Chen, J. Yi, H. Zhang, and C. Hsieh, "Query-efficient hard-label black-box attack: An optimization-based approach," 2018, *arxiv:1807.04457*.

[52] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on GAN," 2017, *arxiv:1702.05983*.

[53] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD cup 99 data set," in *Proc. IEEE Symp. Comput. Intell. Security Defense Appl.*, 2009, pp. 1–6.

[54] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, "DREBIN: Effective and explainable detection of android malware in your pocket," in *Proc. NDSS*, 2014, pp. 1–15.

[55] G. Caminero, M. Lopez-Martin, and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Comput. Netw.*, vol. 159, pp. 96–109, Aug. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128618311216

[56] X. Ma and W. Shi, "AESMOTE: Adversarial reinforcement learning with smote for anomaly detection," *IEEE Trans. Netw. Sci. Eng.*, vol. 89, no. 2, pp. 943–956, Apr./Jun. 2021.

[57] P. Wu, H. Guo, and N. Moustafa, "Pelican: A deep residual network for network intrusion detection," in *Proc. 50th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, 2020, pp. 55–62.

[58] A. Halimaa and K. Sundarakantham, "Machine learning based intrusion detection system," in *Proc. 3rd Int. Conf. Trends Electron. Inform. (ICOEI)*, 2019, pp. 916–920.

[59] J. Liu, B. Kantarci, and C. Adams, "Machine learning-driven intrusion detection for contiki-NG-based IoT networks exposed to NSL-KDD dataset," in *Proc. 2nd ACM Workshop Wireless Security Mach. Learn.*, 2020, pp. 25–30.

[60] I. J. Sanz, M. A. Lopez, E. K. Viegas, and V. R. Sanches, "A lightweight network-based android malware detection system," in *Proc. IFIP Netw. Conf. (Networking)*, 2020, pp. 695–703.

[61] E. Tabassi, K. J. Burns, M. Hadjimichael, A. D. Molina-Markham, and J. T. Sexton, "A taxonomy and terminology of adversarial machine learning," Nat. Inst. Stand. Technol., Gaithersburg, MA, USA, Rep. NISTIR 8269, Oct. 2019. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8269-draft.pdf

[62] J. Wang, G. Dong, J. Sun, X. Wang, and P. Zhang, "Adversarial sample detection for deep neural network through model mutation testing," in *Proc. IEEE/ACM 41st Int. Conf. Softw. Eng. (ICSE)*, May 2019, pp. 1245–1256.

[63] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," Jan. 2020, *arXiv:2001.03994*.

[64] A. Wu, Y. Han, Q. Zhang, and X. Kuang, "Untargeted adversarial attack via expanding the semantic gap," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2019, pp. 514–519.

[65] H. Kwon, Y. Kim, H. Yoon, and D. Choi, "Selective untargeted evasion attack: An adversarial example that will not be classified as certain avoided classes," *IEEE Access*, vol. 7, pp. 73493–73503, 2019.

[66] H. Kwon, Y. Kim, K. Park, H. Yoon, and D. Choi, "Multi-targeted adversarial example in evasion attack on deep neural network," *IEEE Access*, vol. 6, pp. 46084–46096, 2018.

[67] P. Rathore, A. Basak, S. H. Nistala, and V. Runkana, "Untargeted, targeted and universal adversarial attacks and defenses on time series," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2020, pp. 1–8.

[68] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," Jul. 2018, *arXiv:1804.08598*.

[69] S. Zhang, X. Xie, and Y. Xu, "A brute-force black-box method to attack machine learning-based systems in cybersecurity," *IEEE Access*, vol. 8, pp. 128250–128263, 2020.

[70] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," Feb. 2019, *arXiv:1801.02610*.

[71] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Security Privacy (EuroSP)*, Mar. 2016, pp. 372–387.

[72] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," Mar. 2017, *arXiv:1602.02697*.

[73] A. Rahman, T. Rahman, R. Laganiere, N. Mohammed, and Y. Wang, "Membership inference attack against differentially private deep learning model," *Trans. Data Privacy* vol. 11, no. 1, pp. 61–79, 2018.

[74] S. Chen *et al.*, "Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach," *Comput. Security*, vol. 73, pp. 326–344, Mar. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404817302444

[75] J. Clements and Y. Lao, "Hardware trojan design on neural networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2019, pp. 1–5.

[76] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2574–2582. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2016.282

[77] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks," 2016, *arxiv:1608.04644*.

[78] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–28. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb

[79] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proc. 10th ACM Workshop Artif. Intell. Security*, 2017, pp. 15–26. [Online]. Available: http://dx.doi.org/10.1145/3128572.3140448

[80] N. Carlini, G. Katz, C. Barrett, and D. L. Dill, "Ground-truth adversarial examples," 2018, *arXiv:1709.10207v1*.

[81] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Proc. Int. Conf. Comput. Aided Verification*, 2017, pp. 97–117.

[82] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1765–1773.

[83] N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Security Privacy (SP)*, 2016, pp. 582–597.

[84] K. Yang, J. Liu, C. Zhang, and Y. Fang, "Adversarial examples against the deep learning based network intrusion detection systems," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, 2018, pp. 559–564.

[85] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus, "Channel-aware adversarial attacks against deep learning-based wireless signal classifiers," May 2020, *arXiv:2005.05321*.

[86] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proc. ICML Unsupervised Transf. Learn.*, 2012, pp. 1–14.

[87] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2014, *arXiv:1312.6114*.

[88] "Small Cell Networks and the Evolution of 5G—Qorvo." [Online]. Available: https://www.qorvo.com/design-hub/blog/small-cell-networks-and-the-evolution-of-5g (Accessed: May 17, 2017).

[89] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," 2016, *arXiv:1606.03498*.

[90] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Y. Zheng, "Recent progress on generative adversarial networks (GANs): A survey," *IEEE Access*, vol. 7, pp. 36322–36333, 2019.

[91] L. Weng, "From GAN to WGAN," 2019, *arxiv:1904.08994*.

[92] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," 2017, *arXiv:1701.04862*.

[93] M. Ebrahimi, N. Zhang, J. Hu, M. T. Raza, and H. Chen, "Binary black-box evasion attacks against deep learning-based static malware detectors with adversarial byte-level language model," 2020, *arXiv:2012.07994*.

[94] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.

[95] Y. Song, R. Shu, N. Kushman, and S. Ermon, "Constructing unrestricted adversarial examples with generative models," 2018, *arXiv:1805.07894*.

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

[97] L. Onwuzurike, E. Mariconti, P. Andriotis, E. D. Cristofaro, G. J. Ross, and G. Stringhini, "Mamadroid: Detecting android malware by building Markov chains of behavioral models (extended version)," 2017, *arxiv:1711.07477*.

[98] M. Rigaki and A. Elragal, "Adversarial deep learning against intrusion detection classifiers," in *Proc. NATO IST-152 Workshop Intell. Auton. Agents Cyber Defence Resilience*, 2017, pp. 1–14.

[99] X. Chen *et al.*, "Android HIV: A study of repackaging malware for evading machine-learning detection," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 987–1001, 2020.

[100] S. Sivaslioglu, F. Ö. Çatak, and E. Gül, "Incrementing adversarial robustness with autoencoding for machine learning model attacks," in *Proc. 27th Signal Process. Commun. Appl. Conf. (SIU)*, 2019, pp. 1–4.

[101] T. Chadza, K. G. Kyriakopoulos, and S. Lambotharan, "Learning to learn sequential network attacks using hidden Markov models," *IEEE Access*, vol. 8, pp. 134480–134497, 2020.

[102] M. Xue, C. Yuan, H. Wu, Y. Zhang, and W. Liu, "Machine learning security: Threats, countermeasures, and evaluations," *IEEE Access*, vol. 8, pp. 74720–74742, 2020.

[103] Y. E. Sagduyu, Y. Shi, and T. Erpek, "Adversarial deep learning for over-the-air spectrum poisoning attacks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 2, pp. 306–319, Feb. 2021.

[104] K. Davaslioglu and Y. E. Sagduyu, "Trojan attacks on wireless signal classification with adversarial machine learning," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Nov. 2019, pp. 1–6.

[105] S. Karunaratne, E. Krijestorac, and D. Cabric, "Penetrating RF fingerprinting-based authentication with a generative adversarial attack," Nov. 2020, *arXiv: 2011.01538*.

[106] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus, "Over-the-air adversarial attacks on deep learning based modulation classifier over wireless channels," in *Proc. 54th Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2020, pp. 1–6.

[107] Y. Shi, Y. E. Sagduyu, T. Erpek, K. Davaslioglu, Z. Lu, and J. H. Li, "Adversarial deep learning for cognitive radio security: Jamming attack and defense strategies," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2018, pp. 1–6.

[108] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu, "Generative adversarial network for wireless signal spoofing," in *Proc. ACM Workshop Wireless Security Mach. Learn.*, 2019, pp. 55–60. [Online]. Available: http://doi.org/10.1145/3324921.3329695

[109] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu, "Generative adversarial network in the air: Deep adversarial learning for wireless signal spoofing," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 1, pp. 294–303, Mar. 2021.

[110] Y. Arjoune and N. Kaabouch, "A Comprehensive survey on spectrum sensing in cognitive radio networks: Recent advances, new challenges, and future research directions," *Sensors*, vol. 19, no. 1, p. 126, Jan. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/1/126

[111] F. Hu, B. Chen, and K. Zhu, "Full spectrum sharing in cognitive radio networks toward 5G: A survey," *IEEE Access*, vol. 6, pp. 15754–15776, 2018.

[112] Y. Shi, T. Erpek, Y. E. Sagduyu, and J. H. Li, "Spectrum data poisoning with adversarial deep learning," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Oct. 2018, pp. 407–412.

[113] F. Meng, P. Chen, L. Wu, and X. Wang, "Automatic modulation classification: A deep learning enabled approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10760–10772, Nov. 2018.

[114] Y. Wang, J. Yang, M. Liu, and G. Gui, "LightAMC: Lightweight automatic modulation classification via deep learning and compressive sensing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3491–3495, Mar. 2020.

[115] Y. Wu, A. Khisti, C. Xiao, G. Caire, K. Wong, and X. Gao, "A survey of physical layer security techniques for 5G wireless networks and challenges ahead," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 4, pp. 679–695, Apr. 2018.

[116] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized radio classification through convolutional neural networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 370–378.

[117] N. Wang, T. Jiang, S. Lv, and L. Xiao, "Physical-layer authentication based on extreme learning machine," *IEEE Commun. Lett.*, vol. 21, no. 7, pp. 1557–1560, Jul. 2017.

[118] B. Flowers, R. M. Buehrer, and W. C. Headley, "Evaluating adversarial evasion attacks in the context of wireless communications," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1102–1113, 2020.

[119] K. Merchant and B. Nousain, "Securing IoT RF fingerprinting systems with generative adversarial networks," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Nov. 2019, pp. 584–589.

[120] D. Roy, T. Mukherjee, M. Chatterjee, E. Blasch, and E. Pasiliao, "RFAL: Adversarial learning for RF transmitter identification and classification," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 783–801, Jun. 2020.

[121] M. Liu, Z. Xue, X. Xu, C. Zhong, and J. Chen, "Host-based intrusion detection system with system calls: Review and future trends," *ACM Comput. Surveys*, vol. 51, no. 5, p. 98, Nov. 2018. [Online]. Available: http://doi.org/10.1145/3214304

[122] N. T. Van, T. N. Thinh, and L. T. Sach, "An anomaly-based network intrusion detection system using Deep learning," in *Proc. Int. Conf. Syst. Sci. Eng. (ICSSE)*, Jul. 2017, pp. 210–214.

[123] A. K. Saxena, S. Sinha, and P. Shukla, "General study of intrusion detection system and survey of agent based intrusion detection system," in *Proc. Int. Conf. Comput. Commun. Autom. (ICCCA)*, May 2017, pp. 471.

[124] M. Ozay, I. Esnaola, F. T. Y. Vural, S. R. Kulkarni, and H. V. Poor, "Machine learning methods for attack detection in the smart grid," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 8, pp. 1773–1786, Aug. 2016.

[125] S. Schupp, "Limitations of network intrusion detection," in *Global Information Assurance Certification Paper* p. 6, 2000.

[126] A. Chawla, B. Lee, S. Fallon, and P. Jacob, "Host based intrusion detection system with combined CNN/RNN model," in *Proc. ECML PKDD Workshops*, 2019, pp. 149–158.

[127] J. H. Joloudari, M. Haderbadi, A. Mashmool, M. Ghasemigol, S. S. Band, and A. Mosavi, "Early detection of the advanced persistent threat attack using performance analysis of deep learning," *IEEE Access*, vol. 8, pp. 186125–186137, 2020.

[128] M. L. Martín, B. Carro, A. Sánchez-Esguevillas, and J. L. Mauri, "Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in IoT," *Sensors*, vol. 17, no. 9, p. 1967, 2017.

[129] Y. Yang, K. Zheng, B. Wu, Y. Yang, and X. Wang, "Network intrusion detection based on supervised adversarial variational auto-encoder with regularization," *IEEE Access*, vol. 8, pp. 42169–42184, 2020.

[130] J. Wang and C. Wang, "High performance WGAN-GP based multiple-category network anomaly classification system," in *Proc. Int. Conf. Cyber Security Emerg. Technol. (CSET)*, 2019, pp. 1–7.

[131] B. Mohammadi and M. Sabokrou, "End-to-end adversarial learning for intrusion detection in computer networks," in *Proc. IEEE 44th Conf. Local Comput. Netw. (LCN)*, 2019, pp. 270–273.

[132] J. Clements, Y. Yang, A. A. Sharma, H. Hu, and Y. Lao, "Rallying adversarial techniques against deep learning for network security," 2019, *arxiv:1903.11688*.

[133] P. Madani and N. Vlajic, "Robustness of deep autoencoder in intrusion detection under adversarial contamination," in *Proc. 5th Annu. Symp. Bootcamp Hot Topics Sci. Security*, 2018, pp. 1–8. [Online]. Available: https://doi-org.proxy.bib.uottawa.ca/10.1145/3190619.3190637

[134] M. Salem, S. Taheri, and J. S. Yuan, "Anomaly generation using generative adversarial networks in host-based intrusion detection," in *Proc. 9th IEEE Annu. Ubiquitous Comput. Electron. Mobile Commun. Conf. (UEMCON)*, 2018, pp. 683–687.

[135] N. Martins, J. M. Cruz, T. Cruz, and P. H. Abreu, "Analyzing the footprint of classifiers in adversarial denial of service contexts," in *Progress in Artificial Intelligence*, P. Moura Oliveira, P. Novais, and L. P. Reis, Eds. Cham, Switzerland: Springer, 2019, pp. 256–267.

[136] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Addressing adversarial attacks against security systems based on machine learning," in *Proc. 11th Int. Conf. Cyber Conflict (CyCon)*, vol. 900, 2019, pp. 1–18.

[137] G. Apruzzese, M. Colajanni, and M. Marchetti, "Evaluating the effectiveness of adversarial attacks against botnet detectors," in *Proc. IEEE 18th Int. Symp. Netw. Comput. Appl. (NCA)*, 2019, pp. 1–8.

[138] X. Zhang, Y. Zhou, S. Pei, J. Zhuge, and J. Chen, "Adversarial examples detection for XSS attacks based on generative adversarial networks," *IEEE Access*, vol. 8, pp. 10989–10996, 2020.

[139] M. A. Ayub, W. A. Johnson, D. A. Talbert, and A. Siraj, "Model evasion attack on intrusion detection systems using adversarial machine learning," in *Proc. 54th Annu. Conf. Inf. Sci. Syst. (CISS)*, 2020, pp. 1–6.

[140] A. Piplai, S. S. L. Chukkapalli, and A. Joshi, "Nattack! Adversarial attacks to bypass a gan based classifier trained to detect network intrusion," in *Proc. IEEE 6th Int. Conf. Big Data Security Cloud (BigDataSecurity) IEEE Int. Conf. High Perform. Smart Comput. (HPSC) IEEE Int. Conf. Intell. Data Security (IDS)*, 2020, pp. 49–54.

[141] O. Ibitoye, O. Shafiq, and A. Matrawy, "Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[142] D. Wu, B. Fang, J. Wang, Q. Liu, and X. Cui, "Evading machine learning botnet detection models via deep reinforcement learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1–6.

[143] F. Haddadi, D. Phan, and A. N. Zincir-Heywood, "How to choose from different botnet detection systems?" in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp.*, 2016, pp. 1079–1084.

[144] D. Zhao *et al.*, "Botnet detection based on traffic behavior analysis and flow intervals," *Comput. Security*, vol. 39, pp. 2–16, Nov. 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404813000837

[145] D. Shu, N. O. Leslie, C. A. Kamhoua, and C. S. Tucker, "Generative adversarial attacks against intrusion detection systems using active learning," in *Proc. 2nd ACM Workshop Wireless Security Mach. Learn.*, 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1145/3395352.3402618

[146] M. Antonakakis *et al.*, "Understanding the Mirai botnet," in *Proc. 26th USENIX Security Symp.*, 2017, pp. 1093–1110. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[147] W. Wang, Y. Shang, Y. He, Y. Li, and J. Liu, "BotMark: Automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors," *Inf. Sci.*, vol. 511, pp. 284–296, Feb. 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025519308758

[148] A. Janusz, D. Kałuza, A. Chadzyńska-Krasowska, B. Konarski, J. Holland, and D. Slezak "IEEE BigData 2019 cup: Suspicious network event recognition," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, 2019, pp. 5881–5887.

[149] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, Nov. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X18327687

[150] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002.

[151] A. Mottaghi and S. Yeung, "Adversarial representation active learning," 2019, *arXiv:1912.09720*.

[152] A. Ferdowsi and W. Saad, "Generative adversarial networks for distributed intrusion detection in the Internet of Things," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[153] S. Xia, M. Qiu, and H. Jiang, "An adversarial reinforcement learning based system for cyber security," in *Proc. IEEE Int. Conf. Smart Cloud (SmartCloud)*, 2019, pp. 227–230.

[154] K. Hara and K. Shiomoto, "Intrusion detection system using semi-supervised learning with adversarial auto-encoder," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp.*, 2020, pp. 1–8.

[155] E. C. Bayazit, O. K. Sahingoz, and B. Dogan, "Malware detection in android systems with traditional machine learning models: A survey," in *Proc. Int. Congr. Human-Comput. Interact. Optim. Robot. Appl. (HORA)*, 2020, pp. 1–8.

[156] L. Chen, S. Hou, and Y. Ye, "Securedroid: Enhancing security of machine learning-based detection against adversarial android malware attacks," in *Proc. 33rd Annu. Comput. Security Appl. Conf.*, 2017, pp. 362–372. [Online]. Available: https://doi.org/10.1145/3134600.3134636

[157] S. Chen *et al.* "Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach," *Comput. Security*, vol. 73, pp. 326–344, Mar. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404817302444

[158] K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun, and H. Liu, "A review of android malware detection approaches based on machine learning," *IEEE Access*, vol. 8, pp. 124579–124607, 2020.

[159] F. Ceschin, M. Botacin, H. M. Gomes, L. S. Oliveira, and A. Grégio, "Shallow security: On the creation of adversarial variants to evade machine learning-based malware detectors," in *Proc. 3rd Reversing Offensive-Oriented Trends Symp.*, 2019, pp. 1–9. [Online]. Available: https://doi.org/10.1145/3375894.3375898

[160] Q. Wang *et al.*, "Adversary resistant deep neural networks with an application to malware detection," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 1145–1153. [Online]. Available: https://doi.org/10.1145/3097983.3098158

[161] M. Melis, D. Maiorca, B. Biggio, G. Giacinto, and F. Roli, "Explaining black-box android malware detection," in *Proc. 26th Eur. Signal Process. Conf. (EUSIPCO)*, 2018, pp. 524–528.

[162] A. Abusnaina, A. Khormali, H. Alasmary, J. Park, A. Anwar, and A. Mohaisen, "Adversarial learning attacks on graph-based IoT malware detection systems," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2019, pp. 1296–1305.

[163] W. Li, N. Bala, A. Ahmar, F. Tovar, A. Battu, and P. Bambarkar, "A robust malware detection approach for android system against adversarial example attacks," in *Proc. IEEE 5th Int. Conf. Collaboration Internet Comput. (CIC)*, 2019, pp. 360–365.

[164] I. Rosenberg, A. Shabtai, L. Rokach, and Y. Elovici, "Generic black-box end-to-end attack against state of the art API call based malware classifiers," Jun. 2018, *arXiv:1707.05970*.

[165] M. Shahpasand, L. Hamey, D. Vatsalan, and M. Xue, "Adversarial attacks on mobile malware detection," in *Proc. IEEE 1st Int. Workshop Artif. Intell. Mobile (AI4Mobile)*, 2019, pp. 17–20.

[166] O. Suciu, S. E. Coull, and J. Johns, "Exploring adversarial examples in malware detection," Apr. 2019, *arXiv:1810.08280*.

[167] M. Mozaffari-Kermani, S. Sur-Kolay, A. Raghunathan, and N. K. Jha, "Systematic poisoning attacks on and defenses for machine learning in healthcare," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 6, pp. 1893–1905, Nov. 2015.

[168] J. Shen, X. Zhu, and D. Ma, "Tensorclog: An imperceptible poisoning attack on deep neural network applications," *IEEE Access*, vol. 7, pp. 41498–41506, 2019.

[169] Y. Chen, Y. Mao, H. Liang, S. Yu, Y. Wei, and S. Leng, "Data poison detection schemes for distributed machine learning," *IEEE Access*, vol. 8, pp. 7442–7454, 2020.

[170] J. Zhang, B. Chen, X. Cheng, H. T. T. Binh, and S. Yu, "PoisonGAN: Generative poisoning attacks against federated learning in edge computing systems," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3310–3322, Mar. 2021.

[171] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. D. McDaniel, "Adversarial examples for malware detection," in *Proc. ESORICS*, 2017, pp. 62–79.

[172] H. Alasmary, A. Anwar, J. Park, J. Choi, D. Nyang, and A. Mohaisen, "Graph-based comparison of IoT and android malware," in *Computational Data and Social Networks*, (Lecture Notes in Computer Science 11280), X. Chen, A. Sen, W. W. Li, and M. T. Thai, Eds. Cham, Switzerland: Springer, pp. 259–272. [Online]. Available: http://link.springer.com/10.1007/978-3-030-04648-4_22

[173] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, vol. 30. Red Hook, NY, USA: Curran Assoc., 2017, pp. 3146–3154. [Online]. Available: https://papers.nips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

[174] J. Chorowski and J. M. Zurada, "Learning understandable neural networks with nonnegative weight constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 1, pp. 62–69, Jan. 2015. [Online]. Available: http://ieeexplore.ieee.org/document/6783731/

[175] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas, "Malware detection by eating a whole EXE," Oct. 2017, *arXiv:1710.09435*.

[176] M. Khoda, T. Imam, J. Kamruzzaman, I. Gondal, and A. Rahman, "Selective adversarial learning for mobile malware," in *Proc. 18th IEEE Int. Conf. Trust Security Privacy Comput. Commun. 13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, 2019, pp. 272–279.

[177] W. Yuan, Y. Jiang, H. Li, and M. Cai, "A lightweight on-device detection method for android malware," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 9, pp. 5600–5611, Sep. 2021.

[178] C. L. P. Chen and Z. Liu, "Broad learning system: A new learning paradigm and system without going deep," in *Proc. 32nd Youth Acad. Annu. Conf. Chin. Assoc. Autom. (YAC)*, 2017, pp. 1271–1276.

[179] K. Allix, T. F. Bissyandé, J. Klein, and Y. L. Traon, "Androzoo: Collecting millions of android apps for the research community," in *Proc. IEEE/ACM 13th Work. Conf. Min. Softw. Repositories (MSR)*, 2016, pp. 468–471.

[180] L. Li *et al.*, "Androzoo++: Collecting millions of android apps and their metadata for the research community," 2017, *arxiv:1709.05281*.

[181] "Strategic Partners." [Online]. Available: https://www.comodo.com/ (Accessed: Nov. 30, 2021).

[182] A. Kołcz and C. H. Teo, "Feature weighting for improved classifier robustness," p. 8. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.163.5184&rep=rep1&type=pd

[183] F. Wang, W. Liu, and S. Chawla, "On sparse feature attacks in adversarial learning," in *Proc. IEEE Int. Conf. Data Min.*, Dec. 2014, pp. 1013–1018.

[184] F. Zhang, P. P. K. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 766–777, Mar. 2016.

[185] H. Kiliç, N. S. Katal, and A. A. Selçuk, "Evasion techniques efficiency over the IPS/IDS technology," in *Proc. 4th Int. Conf. Comput. Sci. Eng. (UBMK)*, Sep. 2019, pp. 542–547.

[186] S. Raponi, M. Caprolu, and R. D. Pietro, "Intrusion detection at the network edge: Solutions, limitations, and future directions," in *Edge Computing (EDGE)* (Lecture Notes in Computer Science), T. Zhang, J. Wei, and L.-J. Zhang, Eds. Cham, Switzerland: Springer, 2019, pp. 59–75.

[187] M. Tsikerdekis, S. Zeadally, A. Schlesener, and N. Sklavos, "Approaches for preventing honeypot detection and compromise," in *Proc. Global Inf. Infrastruct. Netw. Symp. (GIIS)*, Oct. 2018, pp. 1–6.

[188] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *J. Netw. Comput. Appl.*, vol. 84, pp. 25–37, Apr. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804517300802

[189] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[190] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2805–2824, Sep. 2019.

[191] A. Warzyński and G. Kołaczek, "Intrusion detection systems vulnerability on adversarial examples," in *Proc. Innov. Intell. Syst. Appl. (INISTA)*, Jul. 2018, pp. 1–4.

[192] W. Lee and S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," *ACM Trans. Inf. Syst. Security*, vol. 3, no. 4, pp. 227–261, Nov. 2000. [Online]. Available: https://dl.acm.org/doi/10.1145/382912.382914

[193] H. Hindy *et al.*, "A Taxonomy of network threats and the effect of current datasets on intrusion detection systems," *IEEE Access*, vol. 8, pp. 104650–104675, 2020.

[194] E. K. Viegas, A. O. Santin, and L. S. Oliveira, "Toward a reliable anomaly-based intrusion detection in real-world environments," *Comput. Netw.*, vol. 127, pp. 200–216, Nov. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128617303225

[195] O. Press, A. Bar, B. Bogin, J. Berant, and L. Wolf, "Language generation with recurrent generative adversarial networks without pre-training," Dec. 2017, *arXiv:1706.01399*.

Jinxin Liu (Student Member, IEEE) received the B.Sc. degree from the College of Communication Engineering, Jilin University, China. He is currently pursuing the Ph.D. degree in electrical and computer engineering from the University of Ottawa, where he is a Member of the Next Generation Communications and Computing Networks (NEXTCON) Research Lab. His current research interests include the Internet of Things (IoT), machine learning, and network security.

Michele Nogueira (Senior Member, IEEE) received the Ph.D. degree in computer science from Sorbonne University–UPMC, LIP6, France, in 2009. She is currently an Associate Professor with the Computer Science Department, Federal University of Minas Gerais, Brazil. Her research interests include wireless networks, network security, and dependability. Her works search to provide resilience to self-organized, cognitive and wireless networks by adaptive and opportunistic approaches. She is the Director of the Center for Computational Security sCience (CCSC) Research Lab. She serves as the Chair for the IEEE ComSoc Internet Technical Committee. She served as an Associate Technical Editor for the *IEEE Communications Magazine* and the *Journal of Network and Systems Management*.

Johan Fernandes received the BCS degree (Hons.) in applied computing and the B.Sc. degree in I.T. He is currently pursuing the Master of Computer Science (MCS) degree in Applied Artificial Intelligence with the University of Ottawa. He works as a Research Assistant with the Next Generation Communications and Computing Networks (NEXTCON) Research Lab, University of Ottawa, where his primary field of research is object detection and Image classification using deep learning. He has worked part-time with CIBC Wood Gundy, Windsor, from 2018 to 2019. He has three years of work experience as an SAP Fiori and ABAP developer with Bristlecone India Ltd., from 2014 to 2017.

Burak Kantarci (Senior Member, IEEE) received the Ph.D. degree in computer engineering. He is an Associate Professor and the Founding Director of the Smart Connected Vehicles Innovation Centre (SCVIC), and the Founding Director of the Next Generation Communications and Computing Networks (NEXTCON) Research Lab, University of Ottawa. He is a globally recognized researcher particularly in mobile cloud computing, mobile crowdsensing, and AI-driven solutions for secure and trustworthy cyberspace. He has coauthored over 200 publications in established journals and conferences, and 13 book chapters. He is well known for his contributions to the quantification of data trustworthiness in mobile crowd-sensing (MCS) systems, and game theoretic incentives to promote user participation in MCS campaigns with high value data; as well as AI-backed access control, authentication and machine learning-backed intrusion detection solutions in sensing environments. He served as the Chair of IEEE Communications Systems Integration and Modeling Technical Committee, and has served as the Technical Program Co-Chair/Symposium Co-chair of more than twenty international conferences/symposia/workshops, including IEEE Global Communications Conference (GLOBECOM)—Communications Systems QoS, Reliability and Modeling (CQRM) symposium. In 2021, he has been elected as the new Secretary of IEEE Social Networks Technical Committee. He is an Editor of the IEEE COMMUNICATIONS SURVEYS & TUTORIALS, IEEE INTERNET OF THINGS, *Vehicular Communications* (Elsevier), and an Associate Editor for IEEE NETWORKING LETTERS, and *Journal of Cybersecurity and Privacy*. He is a Distinguished Speaker of the Association for Computing Machinery (ACM) and a Senior Member of ACM.