Physical Layer Authentication and Security Design in the Machine Learning Era

Tiep M. Hoang, Alireza Vahid, Hoang Duong Tuan, and Lajos Hanzo

***Abstract*—Security at the physical layer (PHY) is a salient research topic in wireless systems, and machine learning (ML) is emerging as a powerful tool for providing new data-driven security solutions. Therefore, the application of ML techniques to the PHY security is of crucial importance in the landscape of more and more data-driven wireless services. In this context, we first summarize the family of bespoke ML algorithms that are eminently suitable for wireless security. Then, we review the recent progress in ML-aided PHY security, where the term "PHY security" is classified into two different types: i) PHY authentication and ii) secure PHY transmission. Moreover, we treat NNs as special types of ML and present how to deal with PHY security optimization problems using NNs. Finally, we identify some major challenges and opportunities in tackling PHY security challenges by applying carefully tailored ML tools.**

***Index terms*—**Physical Layer Security, Authentication, Secure Transmission, Machine Learning, Neural Network, Optimization.

I. Introduction

A. Physical Layer Security

Physical layer (PHY) security has become a new line of research independent of the security at higher layers [1]–[3]. While higher-layer security methods rely on cryptography, PHY security methods exploit the randomness of the propagation environment to enhance the security level of wireless systems. The theoretical foundation of PHY security were laid by Shannon in 1949 [4] and Wyner in 1975 [5]. These two pioneering contributions provide us with the understanding of wireless security from the information-theoretic perspective. Furthermore, they also allow us to quantify the security level of the system in terms of the maximum secret information rate. On this basis, a variety of conceptual and mathematical definitions of PHY security have been developed for a wide range of wireless systems over the years [6]. Accordingly, PHY security techniques have also been suggested for supplementing existing security solutions. In parallel to the evolution of wireless communications, PHY security has been investigated in a wide variety of communication systems. Many types of PHY security threats, as well as the corresponding security methods, have been studied over years. In the sequel, we respectively summarize common PHY security threats and methods.

1) PHY Security Threats: Wireless systems tend to encounter many different types of PHY security threats. The

TABLE I
List of abbreviations

Abbreviations	Meanings
AE	Auto-encoder
Complex-NN	Complex-valued neural network
Complex-OPT	Complex-valued optimization
CSI	Channel state information
DOA	Direction of arrival
GAN	Generative adversarial network
GD	Gradient descent
GMM	Gaussian mixture model
iForest	Isolation forest
IoT	Internet-of-Things
k-means	k-means clustering
k-NN	k-nearest neighbour
LDA	Linear discriminant analysis
LED	Light-emitting diode
LiFi	Light fidelity
MIMO	Multiple-input multiple-output
ML	Machine learning
MLP	Multiple layer perceptron
mmWave	Millimeter wave
MOO	Multiple-objective optimization
MUSIC	Multi-signal classification
NLOS	Non-line-of-sight
NN	Neural network
OC-SVM	One-class support vector machine
PGD	Projected gradient descent
PHY	Physical layer
PNN	Projection neural network
Real-NN	Real-valued neural network
Real-OPT	Real-valued optimization
RIS	Reconfigurable intelligent surface
RL	Reinforcement learning
RRN	Recurrent neural network
RSS	Received Signal Strength
RSSI	Received signal strength indicator
SGD	Stochastic gradient descent
SSL	Self-supervised learning
SVM	Support vector machine
TOA	Time of arrival
VB	Variational Bayes
VLC	Visible light communication
VAE	Variational auto-encoder
XAI	Explainable AI
MOEs	Mixture of experts
BST	Binary search tree

T. M. Hoang and A. Vahid were with the Department of Electrical Engineering, University of Colorado Denver, USA. They are now with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, NY 14623, USA (e-mails: tmheme@rit.edu; arveme@rit.edu).

H. D. Tuan is with the School of Electrical and Data Engineering, University of Technology Sydney, Broadway, NSW 2007, Australia (email:Tuan.Hoang@uts.edu.au).

L. Hanzo is with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (email: lh@soton.ac.uk).

The work of T. M. Hoang and A. Vahid was in part supported by NSF grants ECCS-2030285, CNS-2343959, CNS-2343964, and AST-2232482.

H. D. Tuan would like to acknowledge the financial support of the Australian Research Council's Discovery Projects under Grant DP190102501.

L. Hanzo would like to acknowledge the financial support of the Engineering and Physical Sciences Research Council projects EP/W016605/1, EP/X01228X/1 and EP/Y026721/1 as well as of the European Research Council's Advanced Fellow Grant QuantCom (Grant No. 789028).

categorization of threats usually depends on the methodology of the adversaries harnessed for degrading the security level of a specific wireless system. For example, Lee *et al.* [7] classified the PHY attacks inflicted upon the wireless smart grid into four types including eavesdropping, jamming, restricting access and injecting. Another example is the categorization of PHY attacks in [8], where Shu *et al.* also classified eavesdropping and jamming as common PHY attacks applied to cognitive radio networks (CRNs), but additionally suggested other types of threats including primary user emulation and spectrum sensing data falsification. In these examples, the main difference in categorizing PHY attacks between [7] and [8] lies in wireless system modelling. As for CRNs, both primary and secondary channels co-exist, thus the associated aspects like primary users and spectrum sensing are also considered in the context of PHY security. In general, based on the actions of adversaries, the PHY threats on a generic wireless system can be categorized as follows [9]:

- *Passive eavesdroppers*: An adversary is considered as an eavesdropper if it simply listens to channels and collects the information leaked from transmitters to it.
- *Active attackers*: An adversary is considered to be active, when it actively transmits something to gain some benefits. This type of attack includes jamming and spoofing. As for jamming attacks, a jammer can send jamming signals to interfere with a legitimate system and hence imposes a degradation in security performance. By contrast, spoofing attacks tend to deceive a wireless system by impersonating a legitimate user in that system.

The detection of passive eavesdroppers seems to be at first sight impossible due to the fact that they do not transmit any signal. For this reason, passive eavesdroppers are rarely considered in PHY authentication. By contrast, both passive eavesdroppers and active attackers are considered in secure PHY transmissions, because the security-level of the transmission strategy/policy will directly affect on the likelihood of successfully extracting confidential information by the eavesdroppers.

2) PHY Security Methodologies: In PHY security, many different notions of security have been defined, such as ideal security, strong security, weak security and so on (see [6, Table I]). There is also a wide range of PHY security techniques and metrics. The most common ones rely on the channel characteristics and on the transceiver architectures. For example, channel characteristics like channel fading and noise are commonly used for the calculation of the secrecy rate and secrecy outage probability that are channel-based metrics. By contrast, transceiver architectures like hardware impairments can be used for authenticating devices. In general, PHY security solutions can be categorized into the following pair of areas.

- **PHY authentication**: The main objective of PHY authentication is to exploit the physical-layer datasets available at the receivers for authenticating the origin of the received signals [10]. Upon receiving a signal, authentication is needed to ascertain whether it comes from a trusted source or not. Through authentication, the origin of a signal can be tracked and the presence of an illegal user can be identified. In PHY authentication, channel uniqueness between a pair of devices can be exploited as a means of characterizing their identities and positions [2]. A pair of common attacks in PHY authentication are jamming and spoofing attacks. The purpose of jamming attacks is to contaminate the legitimate transmission. By contrast, the purpose of spoofing is to deceive the receiver into believing they come from a legitimate source. In contrast to passive eavesdropping, the jamming and spoofing attacks are generated by active attackers, who take action to proactively degrade the security of a communication system.
- **Secure PHY transmission**: This refers to the way we deal with security vulnerabilities through designing secure transmission strategies as well as systems at the physical layer. From a secure transmission design perspective, securing PHY transmission may incorporate quite different techniques to guard against eavesdropping, such as directional beamforming techniques [11], artificial noise [12], multiple antennas [13], and transmit antenna selection [14]. From a network design perspective, securing PHY transmission also refers to the inclusion, selection, and cooperation of network entities for enhancing the security, such as the cooperation of relay nodes to form cooperative beamforming [15] or the employment of an intermediate node as a friendly jammer [16]. Optimizing the performance of a secure system is another aspect of the secure PHY transmission [17]. The security can be optimized in a number of ways subject to a given set of optimization constraints. For example, a communication system can be designed for ensuring that the probability of information leakage is minimized within a fixed power budget or vice versa, and/or the power consumption is minimized while still guaranteeing the target security level.

To clarify the two above concepts a little further, let us consider a system consisting of an access point and a user in the presence of an adversary. If the access point wants to authenticate the user before transmitting anything, the user has to send something to the access point in the uplink for authentication. The adversary may be active during this authentication period and initiate a jamming attack for preventing the user from successful authentication. PHY security solutions conceived for dealing with this jamming attack belong to the family of **PHY authentication** techniques. After completing the authentication process, the access point may transmit confidential messages to the users in the downlink. At this stage, the adversary may decide to be passive as an eavesdropper to intercept the transmitted signals or continue to remain active by sending jamming to the user. PHY security solutions designed for dealing with the adversary during this downlink session may be classed as **secure PHY transmission** solutions, as exemplified by secure beamforming, which directs the beams of the access point towards the desired user (see [11] and [17]).

B. ML-aided PHY Security

The development of machine learning (ML) has spanned several decades and as a recent benefit, it has led to successful applications in different fields. Hence, the introduction of ML into wireless networks is expected to lead to advanced network architectures, which are capable of learning from wireless data to make data-driven decisions/predictions and perform adaptive network optimization [18]. As part of wireless communications, PHY security is also expected to beneficially exploit ML for further enhancing the security. Since security threats are likely to increase with the number of mobile users, new PHY security solutions should be able to harness a huge amount of data collected from the propagation environments to guarantee the safety and privacy for legitimate users. However, these challenging scenarios tend to result in large parameter spaces for exploration. Hence efficient learning-aided reduce-scope algorithms must be harnessed. In this content, ML algorithms are the appropriate candidates for new PHY security solutions in order to create dependable data-driven security services. Additionally, many ML algorithms are capable of carrying out complex tasks without requiring the availability of closed-form solutions; thus, ML-based security solutions can be supported by well-trained ML algorithms rather than calculating complex mathematical expressions. Although the exploitation of ML in PHY security has been studied (e.g., in [19]–[21]), it is still in its infancy compared to the extensive body of investigations of PHY security methods dispensing with ML. In fact, there are numerous knowledge gaps, which together form a new area of research, namely *machine-learning-aided physical-layer security* (ML-aided PHY security). As a dual pair of the two main branches of PHY security, ML-aided PHY security also encompasses two branches of research: i) ML-aided PHY authentication, and ii) ML-aided secure PHY transmission.

As for ML-aided PHY authentication, the goal is to build ML-based authenticators that learn from physical-layer datasets to distinguish between legitimate transmitters and illegitimate ones. Herein, illegitimate transmitters may imply jammers, spoofers or other types of attackers. Regardless of the attack types and their objectives (e.g., jamming attacks to confuse systems or spoofing attacks to intrude into systems), the authentication mechanism focuses on distinguishing normal signals from the abnormal ones; thus, both supervised and unsupervised classification algorithms (inclusive of neural-network-based classification algorithms) are appropriate candidates for performing the authentication task [20], [22]–[29]. Indeed, many of the ML-assisted techniques do not even have to set up a fixed threshold for distinguishing the related data, while traditional PHY authentication solutions critically rely on a threshold and appear to be sensitive to the change in its value upon making compromised/uncompromised decision [22], [26]. This suggests that the traditional methods, which typically rely on hypothesis testing, are vulnerable to non-stationary data. Thus, reinforcement learning–which is a salient branch of ML–may also be used for detecting active spoofers in dynamically fluctuating environments [19].

As for ML-aided secure PHY transmission, many studies show the applications of ML in enhancing the security of wireless systems. Among the most popular approaches are neural networks (NNs) and reinforcement learning [21], [30]–[34]. However, there are still some contributions considering other ML types to perform secure PHY transmission, for example, support vector machine (SVM) and Bayesian solutions [35]. Noticeably, wireless transmission is often geared towards optimal transmission designs, formulated as optimization problems. Accordingly, the process of designing secure communication systems lend themselves to ML-aided secure PHY transmission [21], [30]–[35].

In general, the aforementioned treatises on ML-aided PHY security demonstrate the applicability of ML to securing communication systems. However, compared to the mature conventional PHY security methods, the ML-based PHY security is still an immature area of research. Nowadays, in the context of data explosion, the ability to learn from data may allow ML to provide new PHY security solutions that adapt to the unpredictability in the environment. Indeed, along with the development of next-generation wireless systems, the field of ML itself also has its own advances in terms of new algorithms, which will create hitherto unexplored ML-aided PHY security solutions in the near future. Hence, there is an urgent need to critically appraise all to guide and inspire future research.

C. Prior Art

There is already substantial literature on the application of ML in general communication problems, e.g., [18], [36]–[39]; however, they do not focus on the topic of wireless PHY security. On the other hand, the recent surveys on PHY security do not delve into the application of ML, although they do allude to ML in the context of PHY security [40]–[42]. For example, Ruzomberka *et al.* [40], as well as Solaija *et al.* [42], present emerging wireless architectures and their associated security threats, while briefly touching upon the role of ML. In [41], Khalid *et al.* narrow down the PHY security to reconfigurable intelligent surface (RIS)-aided wireless designs and suggest ML as potential tools for RIS-aided secure systems. Only the authors of [37], [38] and [39] consider anomaly detection, but with no particular attention dedicated to PHY security. However, there is a paucity of literature on ML-aided PHY security, with the exception of [43]–[45].

In particular, Chen *et al.* [43] characterize the capability of deep learning to extract attack features in a cyber-physical transportation system. By contrast, Fang *et al.* [44] portray some challenges to be faced in ML-aided PHY authentication. However, these short magazine papers are constrained to an outline of the vast realm of PHY security. Similarly, Wang *et al.* [45] limit their scope of ML-aided PHY security to spectrum-sharing schemes. A more detailed PHY security landscape was portrayed by Baldini *et al.* in [46] and by Islam *et al.* in [47]. To elaborate, [46] focuses on exploiting the unique hardware fingerprints of physical devices as features and appraises the potential of some ML classification algorithms for PHY authentication. By contrast, Islam *et al.* [47] surveys PHY security vulnerabilities and countermeasures in the context of smart energy systems.

TABLE II
A COMPARISON BETWEEN THIS SURVEY AND OTHER PREVIOUS SURVEYS.

		Tutorial-and-survey papers										
		[39] 2016	[46] 2017	[36] 2018	[47] 2019	[48] 2020	[49] 2021	[50] 2022	[51] 2022	[52] 2022	[40] 2023	This survey
ML-aided PHY security			✓		✓	✓	✓	✓			✓	✓
ML-aided cyber security		✓		✓		✓			✓	✓		
ML-aided PHY Authentication			✓		✓	✓	✓	✓				✓
ML-aided PHY Secure Transm.					✓		✓					✓
Neural Network-Aided Security Optimization												✓
PHY-related features	TOA											✓
	DOA											✓
	RSS							✓		✓		✓
	Channel Gain							✓				✓
	Fingerprints		✓			✓		✓		✓		✓
Discussion of ML algorithms	k-NN	Detailed	Detailed			Detailed		Limited	Limited	Detailed		Detailed
	SVM	Detailed	Detailed	Limited		Detailed		Limited		Detailed		Detailed
	LDA		Limited									Detailed
	k-Means	Limited	Limited		Limited	Detailed		Limited	Limited	Limited		Detailed
	OC-SVM	Detailed										Detailed
	Random Forest	Detailed				Detailed		Limited		Detailed		Detailed
	iForest											Detailed
	Hierarchical Clustering	Limited	Limited									Detailed
	RL			Detailed		Detailed		Limited		Detailed		Limited
	NNs	Detailed	Detailed	Detailed	Limited	Detailed		Limited	Limited	Detailed		Detailed

A few years ago, Al-Garadi *et al.* [48] surveyed security of the Internet of Things and occasionally mentioned PHY authentication, but the prevalent security issues at the physical layer are not addressed. Nguyen *et al.* [49] outline the current challenges to be faced in the PHY security and cover a wide range of technical solutions. However, the ML techniques as well as ML-aided security solutions are not emphasized. Liu *et al.* [50] first focus on the PHY-layer and data-link-layer features of IoT device types, such as the features passively collected from radiometric fingerprints and network flow patterns, then review the ML-aided methods of identifying/authenticating IoT devices. However, since the topic of secure PHY transmission is not covered by [50], beamforming-aided security designs and PHY security optimization are not discussed. On the other hand, the pair of recent survey papers [51], [52] are inherently intended for ML-aided cyber-security, rather than ML-aided PHY security, although the authors briefly touch upon some of the PHY security risks.

Last but not least, PHY-security-related optimization problems are not presented in [18], [36]–[39], [43], [44], [46]–[52]. Given that NNs can be exploited for efficiently solving stochastic optimization problems, it is promising to use NNs for optimizing PHY security. This is a challenge to be tackled, but this opportunity was missed by the authors of [18], [36]–[39], [43], [44], [46]–[52]. By contrast, the subject of using NNs for handling PHY security optimization problems constitutes an important part of this survey. Table II highlights the major differences between this work and the related survey papers.

D. Contributions

When it comes to the employment of ML in PHY security, we consider the following pair of pivotal goals:

- **GOAL 1**: Harnessing ML algorithms for PHY authentication and for securing PHY transmission.
- **GOAL 2**: Invoking artificial NNs for PHY security with a focus on design optimization.

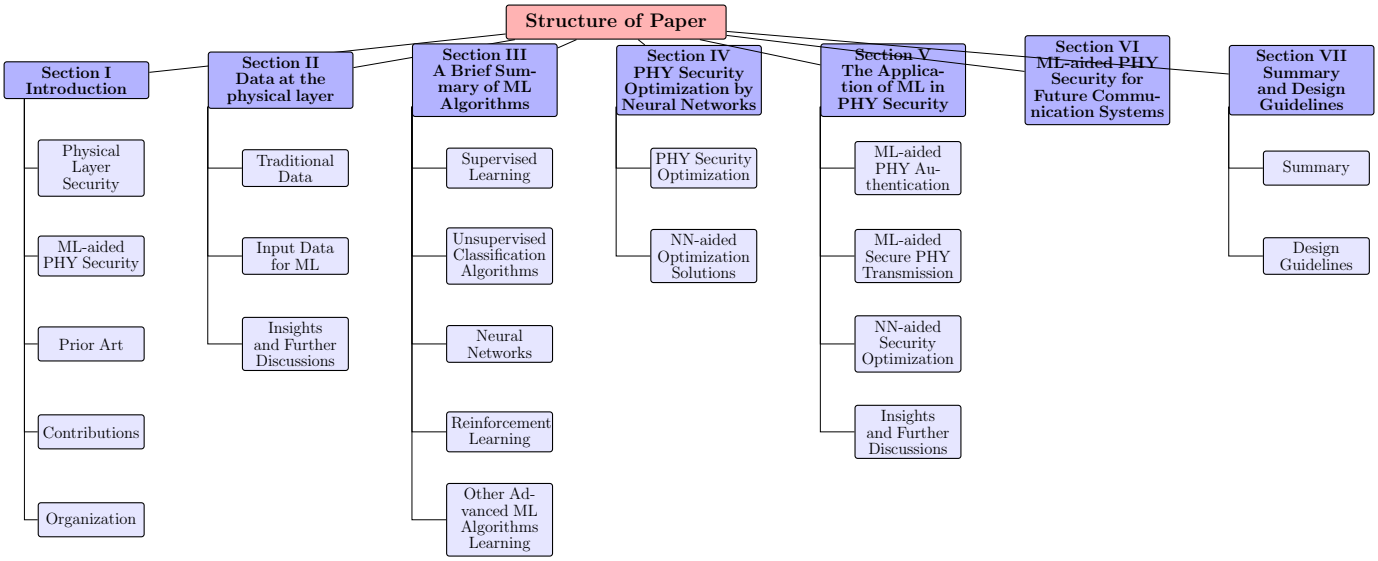


Fig. 1. The structure of the paper.

The aforementioned goals are challenging, hence there are gaps in the ML-aided PHY security literature, especially in terms of survey and tutorial papers. Note that the second goal has not been yet considered in any previous works as highlighted at a glance in Table II. Thus, we aim for achieving these two goals in this paper.

Our main contributions are summarized as follows:

- We commence by reviewing the typical primary data sources routinely employed in wireless communication systems for training ML algorithms.
- From the rich family of ML classification algorithms, we review four typical supervised learning algorithms, four typical unsupervised learning algorithms, and NNs that may belong either to the family of supervised or unsupervised learning. We dedicate particular attention to anomaly detection methods that have been ignored in previous surveys. Additionally, we discuss the application potential of ML classification in meeting **GOAL 1**.
- Regarding **GOAL 2**, we bridge the gap between PHY security optimization and the ability of NNs in dealing with stochastic optimization. Although the employment of NNs for solving optimization problems has found favour in ML research, it has not been popularized in secure communications. Hence, this is the first survey critically appraising the application potential of NNs in optimizing secure PHY transmission. Furthermore, we also reveal the ability of NNs to optimize communication system designs. In this vast field, securing PHY transmission constitutes a particularly important instance, because they must operate in the face of uncertainty, where conventional techniques fail.

E. Organization

The remainder of this survey is organized as follows. In Section II, we present the typical types of data at the physical layer that can be beneficially used by ML algorithms. Section III presents typical ML algorithms that can support PHY security. In Section IV, we present how NNs can deal with PHY security optimization problems. The application of ML in PHY security is reviewed in Section V. The potential of ML in securing different future wireless systems is discussed in Section VI. Finally, the summary of the paper and design guidelines are presented in Section VII. Fig. 1 shows the structure of this paper.

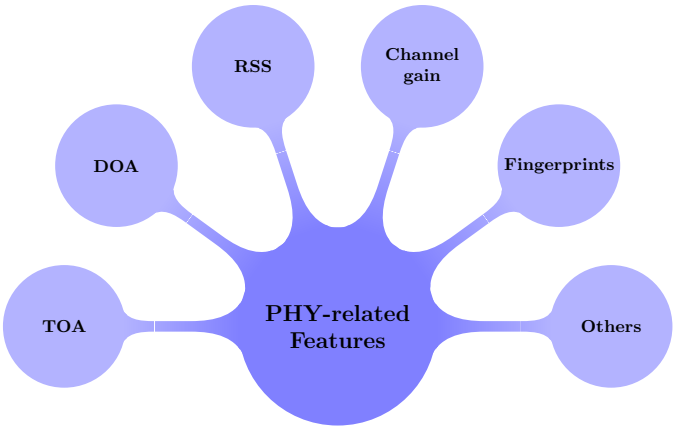


Fig. 2. Potential features for PHY security issues.

II. Data at the Physical Layer

In this section, we present the traditional types of data that are routinely considered at the physical layer followed by highlighting conversion of the traditional data into the input data to be fed into ML algorithms for their training. Unlike non-ML PHY security solutions, ML-aided ones are based on data. Naturally, the understanding of data will play an important role in analyzing data-driven security applications and services.

A. Traditional Data

This subsection presents several typical features of the data handled by communication systems. Fig. 2 provides an overview of the potential features evaluated for PHY authentication, including five specific types of data that are presented in this subsection.

1) Time of Arrival (TOA): The TOA is a common metric used for characterizing the time dispersion of a signal. We can express the ToA (in seconds) as follows:

$$t = t_0 + d/c + n + e_{\text{NLOS}}, \tag{1}$$

where t_0 (s) is the transmit time instant, d (m) is the distance between the transmitter and receiver, $c = 3 \times 10^8 (\text{m/s})$ is the speed of light, n is the noise, and $e_{\text{NLOS}} \geq 0$ is the measurement error mainly caused by non-line-of-sight (NLOS) propagation. In general, it is widely accepted that n is Gaussian distributed. For the NLOS error, several distributions have been used for characterizing the impact of NLOS conditions on time dispersion. For example, e_{NLOS} can be described by an exponentially distributed random variable [53]–[55], a Gaussian distributed random variable [56], or a uniformly distributed random variable [57]. Note that $e_{\text{NLOS}} = 0$, if there is no NLOS propagation.

2) Direction of Arrival (DOA): Another type of data that a receiver can attain is the DOA of a certain signal. The DOA estimation can be performed using direction-finding methods, such as the popular multi-signal classification (MUSIC) algorithm and Capon's minimum variance method [58], [59]. In general, the performance of the DOA estimation will much depend on the receiver antenna configuration. Let $\mathbf{a}$ be the array response in a direction θ and $\{\mathbf{y}_1, \ldots, \mathbf{y}_K\}$ be a set of received signals. Then, the sample covariance matrix is calculated as $\mathbf{\Sigma}_{\text{sample}} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{y}_k \mathbf{y}_k^H$, where $(\cdot)^H$ denotes the conjugate transpose operator. Using the singular value decomposition (SVD), $\mathbf{\Sigma}_{\text{sample}}$ can be partitioned into $\mathbf{\Sigma}_{\text{sample}} = \mathbf{U} \mathbf{\Sigma}_{\text{diag}} \mathbf{U}^H$, where $\mathbf{\Sigma}_{\text{diag}}$ is a rectangular diagonal matrix. The estimated value of θ can be found using the MUSIC algorithm as follows [60], [61]:

$$\theta = \arg\max \frac{1}{|\mathbf{a}(\theta)^H \mathbf{U}_{\text{noise}}|^2}, \tag{2}$$

where $\mathbf{U}_{\text{noise}}$ is extracted from $\mathbf{U}$ (see [60], [61] for more details).

The DOA estimation plays a pivotal role in many applications such as positioning and tracking systems. From the PLS perspective, the DOA estimation techniques are also expected to play an important role in localizing the adversaries' positions, or at least the adversaries' direction.

3) Received Signal Strength (RSS): The RSS can be readily quantified through the measurement of the received power at a certain receiver. More specifically, if $P_r(d)$ is the *empirically* received power at a point d meters away from the transmitter, then $P_r(d)$ can be viewed as a quantitative value for RSS at distance d. In practice, RSS values are likely to be inferred indirectly from the *received signal strength indicator* (RSSI) values. It should be noted that RSSI values represent the readings obtained directly from a receiver. Moreover, the dynamic range of RSSI depends on the quality of the electronics manufacturers. Thus, if a simple receiver records the RSSI readings, this has to be translated from these values to the RSS values [62], [63]. For example, if TelosB motes[1] are used for measuring the RSSI values, then the corresponding RSS values can be calculated by using the following relationship [62]:

$$P_r(d) = \text{RSSI}(d) + \text{RSSI}_{\text{offset}} \tag{3}$$

where $P_r(d)$ is in dBm, $\text{RSSI}(d)$ is the measured RSSI value (in dBm) at the distance d, and $\text{RSSI}_{\text{offset}} = -75$ dB is an offset value.

4) Channel State/Response: Communication channels are random in nature and vary with the surrounding environment. Let us denote the channel between a transmitter and a receiver by $\mathbf{h}_{channel}$. The role of the channel in a simple received signal model can be formulated as

$$r_{Rx} = \mathbf{h}_{channel}^{\dagger} \mathbf{s}_{Tx} + n_{Rx}, \tag{4}$$

where $\mathbf{s}_{Tx}$ is the signal transmitted by multiple antennas, r_{Rx} is the signal received by a single antenna, n_{Rx} is the receiver noise, and † denotes the Hermitian operator. Normally, $\mathbf{h}_{channel}$ is assumed to obey a certain complex Gaussian distribution, but it depends on the type of channel fading. Common types of the channel fading are modelled by Rayleigh, Rician, and Nakagami-m fading, just to name a few [64], [65]. The unique random nature of $\mathbf{h}_{channel}$ is actually helpful in dealing with eavesdropping, because secure transmission can be performed at the physical layer [1], [66].

5) Fingerprints: Hardware imperfections during the manufacturing process are unavoidable. However, that makes each hardware device unique and thus, the uniqueness of hardware is what can also be exploited for device identification [46], [67]. The small differences in hardware are often termed as hardware *fingerprints*. From a security point of view, hardware fingerprints constitute unique characteristics that can be exploited as the unique features of the input data for ML algorithms. The typical fingerprints include local oscillator frequency [67], local oscillator offset, carrier frequency offset [68], and the mismatch between in-phase and quadrature-phase components [69].

6) Central tendency and Dispersion: A lot of characteristics in wireless communications are non-static, including but not limited to channel fading, transmitted signal, and noise. Hence, wireless studies often consider those characteristics as random variables and resort to probability and statistics for the sake of analysis. An example is channel fading that is theoretically assumed to obey a distribution with some central tendency and dispersion. The popular measures of central tendency include mean, mode and median, while central dispersion encompasses variance, deviation, and percentile. In general, the mean and variance of a distribution are mostly employed. When not having the actual mean and variance, the practical measurements will provide the sample mean and variance of a random variable for analysis.

[1]Each TelosB mote is integrated with an IEEE 802.15.4-compliant RF transceiver.

Let us denote the sample mean of the class $(+1)$ and that of the class (-1) by $\overline{\mathbf{x}}_{\mathcal{T}_1}$ and $\overline{\mathbf{x}}_{\mathcal{T}_2}$, respectively. Then, we have

$$\overline{\mathbf{x}}_{\mathcal{T}_i} = \frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} \mathbf{x}_{\text{input}}^{[t]}, \tag{5}$$

where the subscript i is in the set $\{1, 2\}$, $\mathbf{x}_{\text{input}}^{[t]}$ is an input vector of values measured at time slot t, $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 = \{1, \ldots, T\}$ is a set of T time slots. Also, we can calculate the sample covariance matrices as follows [70, Ch.5]:

$$\mathbf{\Sigma}_{sample,i} = \frac{\sum_{t \in \mathcal{T}_i} \left(\mathbf{x}_{\text{input}}^{[t]} - \overline{\mathbf{x}}_{\mathcal{T}_i}\right) \left(\mathbf{x}_{\text{input}}^{[t]} - \overline{\mathbf{x}}_{\mathcal{T}_i}\right)^{\top}}{(|\mathcal{T}_i| - 1)}, \tag{6}$$

7) Others: Apart from the aforementioned types of data, ML-aided wireless systems can also employ other types, including the direction of departure (DOD) and multipath delays [71], the distances among transceivers [72], the sample mean and variance of received signals [73], the Euclidean distance and Pearson correlation of channel samples [22], just to name a few. In general, on the basis of the information collected/measured by a wireless system, we can define a wide range of user-defined measures and use these measures as potential input data for ML. Indeed, there is no limit for combining the available information in the interest of creating helpful datasets. Having said that, the datasets built on the basis of channel measurements are commonly used as the input data, e.g., see [22], [24], [25].

To employ ML algorithms for PHY security research, it is not required to build separate datasets different from other wireless communication topics. Indeed, there is no standard or special requirement for building datasets for ML-aided PHY security. Moreover, the importance of different features affecting the PHY security performance has not been fully understood at the time of writing.

B. Input Data for ML

Once the wireless data to be evaluated has been collected, it is necessary to process the data to make them more useful. If wireless data is viewed as the raw data before preprocessing, then the processed data may be fed into ML models. The process of transforming the raw data into the processed data is termed as data preparation required for efficient ML-based processing.

Example 1. *Assume that we choose a certain channel vector $\boldsymbol{h}_{channel} = [2 + 1j, 3 - 2j]^T$ to be the raw data and want to enter it into an ML model, but this model only accepts real-valued vectors as its input. Then, we have to find a process that can create a new representation of the data, namely $\boldsymbol{x}_{input}$, from the raw data $\boldsymbol{h}_{channel}$. If we opt for a process that converts each element of $\boldsymbol{h}_{channel}$ into its absolute square, then we will obtain $\boldsymbol{x}_{input} = \left[|2 + 1j|^2, |3 - 2j|^2\right]^T$. As a result, the processed data now meets the requirement of the ML model for real-valued input.*

ML algorithms are expected to work well on the processed data and produce good performance. In general, data preparation is the first stage before being able to apply ML algorithms to practical tasks. In data preparation, the following steps should be taken into account:

- **Data cleaning**: Datasets from practical problems are not always readily available and properly formatted. Hence, there is a need to consider data cleaning so as to cope with missing data and transform categorized data into a numerical representation, which may be understandable by an ML model [74], [75].
- **Feature selection**: When a dataset contains irrelevant and redundant features, a process of selecting suitable features should be harnessed as part of a feature selection method in order to eliminate the irrelevant and redundant features [76], [77]. These feature selection methods are also often referred to as search methods. A range of popular search algorithms can be used for performing feature selection, such as the exhaustive search and the branch-and-bound algorithm [78].
- **Feature extraction**: An optional step to take is the process of extracting (or creating) new features from the original features. Depending on what type of data and problem we are handling, a suitable method of feature extraction can be employed for reducing the number of features. In general, the new features created by feature extraction may result in reduced complexity. For example, principal component analysis [79] can transform the data from a high-dimensional space into a lower-dimensional space, while retaining as much valuable information as possible [76], [80].

When it comes to the features of a dataset in a secure wireless system, it is still an open question, because there are no standards for defining the features. In general, the features can be extracted from information, such as the location, channel gain, or hardware [81]. It is also possible to employ the metrics mentioned in Subsection II-A in order to create relevant features.

To illustrate what has been discussed above, let us consider a wireless system whose transmission relies on time slots. A receiver of the system considered might rely on the DOA and RSS to create a dataset. Let us denote the realization of DOA and that of RSS at the t-th time slot by $\text{DOA}[t]$ and $\text{RSS}[t]$. By arranging $\text{DOA}[t]$ and $\text{RSS}[t]$ in a column of two elements, the receiver can form a data point as follows:

$$\mathbf{x}_{\text{input}}^{[t]} = \Big[\underbrace{\text{DOA}[t]}_{\text{feature 1}}, \ \underbrace{\text{RSS}[t]}_{\text{feature 2}}\Big]^T$$

in the two-dimensional space. In practice, $\text{DOA}[t]$ and $\text{RSS}[t]$ can take negative values, e.g., $\text{DOA}[t] = -45^\circ$ and $\text{RSS}^{[t]} = -20$ dB. If an ML model requires every element of $\mathbf{x}_{\text{input}}^{[t]}$ to be positive, then we have to define $\mathbf{x}_{\text{input}}^{[t]}$ in another way. For example, we can define it either as:

$$\mathbf{x}_{\text{input}}^{[t]} = \Big[\underbrace{|\text{DOA}[t]|}_{\text{feature 1}}, \ \underbrace{|\text{RSS}[t]|}_{\text{feature 2}}\Big]^T$$

or as:

$$\mathbf{x}_{\text{input}}^{[t]} = \Big[\underbrace{|\text{DOA}[t]|}_{\text{feature 1}}, \ \underbrace{10^{\text{RSS}[t]/10}}_{\text{feature 2}}\Big]^T.$$

In doing so, $\mathbf{x}_{\text{input}}^{[t]}$ can now be fed into that ML model. For the sake of generalization, let us define M as the number of features. Furthermore, let us denote by Φ_m (with $m \in \{1, 2, \ldots, M\}$) the m-th feature of the data, where $\phi_m^{[t]}$ represents the observed value of Φ_m at the t-th time slot. After T time slots, the receiver will have T data points that form the data input formulated as

$$\mathbf{X}_{\text{input}} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_{\text{input}}^{[1]} & \mathbf{x}_{\text{input}}^{[2]} & \ldots & \mathbf{x}_{\text{input}}^{[T]} \\ | & | & & | \end{bmatrix} = \begin{bmatrix} \phi_1^{[1]} & \phi_1^{[2]} & \ldots & \phi_1^{[T]} \\ \phi_2^{[1]} & \phi_2^{[2]} & \ldots & \phi_2^{[T]} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_M^{[1]} & \phi_M^{[2]} & \ldots & \phi_M^{[T]} \\ \text{slot 1} & \text{slot 2} & \ldots & \text{slot T} \end{bmatrix} \begin{matrix} \\ \text{feature } \Phi_1 \\ \text{feature } \Phi_2 \\ \vdots \\ \text{feature } \Phi_M \end{matrix} . \quad (7)$$

To generalize what will be discussed in the rest of the paper, we will not restrict the number of features. This means that the number of elements in the column vector $\mathbf{x}_{\text{input}}^{[t]}$ can be an arbitrary positive integer. Moreover, it is not necessary to employ the DOA and RSS to create features. Instead, the feature selection and extraction will depend on the particular problems considered.

A data point can be associated with a label or not. Hence, the input data can be classified into labelled data or unlabelled data or in fact a mixture of both. Moreover, based on the ratio of labelled data points and unlabelled ones, the input data can also be considered to be balanced or unbalanced. In the sequel, we will describe these types of input data in more details.

1) Labelled and Unlabelled Data: In authentication problems, it is important to recognize the presence and intrusion of adversaries that can be passive as eavesdroppers or active as attackers. Once an adversary has been detected, we can stick a label on the specific data input that is related to that adversary. If we stick the label $y_{\text{label}}^{[t]} = (-1)$ on the input data $\mathbf{x}_{\text{input}}^{[t]}$ in order to mark a wireless system pertubed by adversaries, then we may stick another label, e.g., $y_{\text{label}}^{[t]} = (+1)$ on a secure wireless system free any adversary. Based on the availability of $\mathbf{x}_{\text{input}}^{[t]}$ and $y_{\text{label}}^{[t]}$, three types of data can be formed:

- **Labelled data**: From the statistical knowledge of previous transmissions, a wireless system can learn about adversaries (such as jammers and spoofing attackers). To be more specific, if the system suspects the presence of adversaries at a certain time slot t (with $t \in \{1, \ldots, T\}$), then it will be able to form a labelled dataset: There are two possibilities:

Time slot	Labelled data	
	Input	Output
1	$\mathbf{x}_{\text{input}}^{[1]}$	$y_{\text{label}}^{[1]}$
$\vdots$	$\vdots$	$\vdots$
T	$\mathbf{x}_{\text{input}}^{[T]}$	$y_{\text{label}}^{[T]}$

  1) If $y_{\text{label}}^{[t]}$ is the same for $\forall t \in \{1, \ldots, T\}$, then the labelled dataset has only a single class (or one label).
  2) However, if we have $y_{\text{label}}^{[t]} \neq y_{\text{label}}^{[t']}$, with $t \neq t'$ and $t', t \in \{1, \ldots, T\}$, then the labelled dataset has two classes (or two labels).
- **Unlabelled data**: When there is no information about any adversary and we cannot ascertain whether the wireless system is secure, then the value of $y_{\text{label}}^{[t]}$ at the t-th time slot is simply unknown to the system. Explicitly, we must not put the *secure label* $(+1)$ on the output $y_{\text{label}}^{[t]}$, if we cannot be sure whether or not the corresponding input $\mathbf{x}_{\text{input}}^{[t]}$ is related to adversaries. As a result, unlabelled data will not produce any data output.

In practice, it is difficult to attain any knowledge about the covert adversaries. For example, a legitimate link between a legitimate user and a receiver may be perfectly known to the receiver. However,the illegitimate link between an eavesdropper and the receiver considered may be completely unknown since a so-called passive eavesdropper never transmits and hence, its channel cannot be estimated. Thus, it is reasonable to consider a practical wireless network to have no channel state information (CSI) for any of the eavesdroppers. This means that real-world datasets are likely to contain only a single class, i.e., $(+1)$. The lack of any knowledge about the eavesdroppers' CSI hampers the associated security analysis and design. Hence, typically, the assumption of having *imperfect* CSI for the eavesdroppers is used [73]. Moreover, further research is conducted to develop sophisticated methods of detecting passive eavesdroppers [82].

2) Balanced and Imbalanced Data: Let us now return to the labelled data and partition it into two groups: one of them includes all outputs $(+1)$, while the other includes all outputs (-1). Let us now denote by $T_{(+1)}$ and $T_{(-1)}$ the number of data points in the first group and in the second group, respectively. Naturally, $T_{(+1)} + T_{(-1)} = T$ is the total number of data points that have been collected and labelled so far. Depending on the ratio $r_{\text{freq}} = \frac{T_{(-1)}}{T}$, the data can be considered as balanced or imbalanced. If this ratio is nearly 0.5, the data is considered as balanced, since the number of data points in each class is nearly the same. By contrast, if the ratio r_{freq} tends to 0 (or 1), the data becomes imbalanced. The value of r_{freq} informs us of how frequently active attackers attempt to perturb the system [73].

Let us define

$$\mathcal{T}_1 = \left\{ t | y_{\text{label}}^{[t]} = (+1) \right\}, \quad (8)$$

$$\mathcal{T}_2 = \left\{ t | y_{\text{label}}^{[t]} = (-1) \right\} = \{1, \ldots, T\} \setminus \mathcal{T}_1. \quad (9)$$

As such, the number of elements in $\mathcal{T}_1$ is $|\mathcal{T}_1| = T_{(+1)}$, while that number in $\mathcal{T}_1$ is $|\mathcal{T}_2| = T_{(-1)}$. Fig. 3 presents an example in which adversaries are present in a wireless system $|\mathcal{T}_2|$ times during T transmissions. Let us assume for example that $|\mathcal{T}_2| = 10$ and $T = 1000$. Then the ratio of $T_{(-1)}$ to T is $r_{\text{freq}} = 10/1000$. In Fig. 3, the data point $\mathbf{x}_{\text{input}}^{[\tau]}$, which corresponds to the τ-th time slot, is classified as $(+1)$. Furthermore, $\mathbf{x}_{\text{input}}^{[\tau']}$ is classified as (-1), where $\tau' \neq \tau$.

Counting elements in each class	Labelled data		
	Input	Output	
1	$\vdots$	$(+1)$	No attack
$\vdots$	$\mathbf{x}_{\text{input}}^{[\tau]}$	$\vdots$	
$\lvert\mathcal{T}_1\rvert = 990$	$\vdots$	$(+1)$	
1	$\vdots$	(-1)	Attack
$\vdots$	$\mathbf{x}_{\text{input}}^{[\tau']}$	$\vdots$	
$\lvert\mathcal{T}_2\rvert = 10$	$\vdots$	(-1)	

Fig. 3. An example of imbalanced data table.

As such, if adversaries are present frequently in the case of passive eavesdropping (or they attack frequently in the case of active attacker) and we are aware of them, then we may build balanced data in which the cardinality of the two groups of data points is nearly equal. However, if the adversaries are not present frequently, we may have imbalanced data. It should be noted that if we do not know anything about adversaries, then the datasets only have a single label associated with normal data points. On the other hand, the datasets having more than one label imply that the existence of adversaries (i.e., passive eavesdroppers or active attackers) has been previously recorded by wireless systems.

Both balanced and unbalanced training datasets can be trained by supervised learning, as long as the datasets have more than a single label (i.e., a single class). The existence of multiple classes in the training dataset allows supervised learning to distinguish one class from the other during the training process. For example, we can use a wide range of ML algorithms to distinguish the two classes, which is a basic binary classification problem. Widely-used ML models include logistic regression, SVM, NNs, random forests, and decision trees, just to name a few.

In the case of unbalanced data, if there is only a single label $(+1)$, then unsupervised learning can be employed because it does not require the data to have multiple classes. On the other hand, if an unbalance dataset has a few (-1) data points, then several types of unsupervised ML algorithms can also be used to distinguish the group with fewer data points from the group with more data points. Among those algorithms are the isolation forest [83] and one-class support vector machine (OC-SVM) [84].

C. Insights and Further Discussions

Since there is no standard for selecting PHY data features, the input data formulation of an ML algorithm may depend on the availability of raw data. For instance, theory-oriented studies can generate channel realizations based on the assumption of certain channel distribution and then use the channels as the input data. In these studies, however, the fingerprinting data is unlikely to be available, because the unique characteristics of hardware imperfections (e.g., local oscillator offset and carrier frequency offset) should be obtained by experiments in measurement-oriented works. For this reason, there is no complete comparison of the influence of features on the attainable PHY security performance. Instead, the importance of different features should be appraised for specific systems and use cases.

It should also be noted that the availability of data comes with the data labelling. Labelled data is normally employed for training supervised ML algorithms. However, if some data points in the training datasets are incorrectly labelled, then the performance evaluation during the testing phase on new data may result in wrong label assignments. This means that supervised learning relies heavily on the labelling process that may be time-consuming and error-prone. On the other hand, unlabelled data can be used in unsupervised ML algorithms and does not require human intervention. However, the outcomes of unsupervised learning are likely to be less accurate than those of supervised learning.

The label assignment to the input data mainly depends on the prior knowledge of legitimate and illegitimate network entities. For example, if a wireless system has ever successfully detected jamming attacks based on the analysis of RSS, it holds a record of previous jamming attacks and assigns (-1) to the RSS values related to the previous jamming attacks. By contrast, $(+1)$ is assigned to other RSS values that are not related to jamming. However, if the system has never experienced a jamming attack, it may only have input data purely associated with all data points labelled as $(+1)$. Generally, the datasets having two classes are utilized by supervised learning, while the datasets having a single class can be used in anomaly detection algorithms (i.e., OC-SVM classifiers) that belong to the family of unsupervised learning (see [73] and [84]).

III. A Brief Summary of ML Algorithms

In this section, we review the basic principles of ML algorithms, including four typical supervised learning algorithms, four typical unsupervised learning algorithms, reinforcement learning, and NNs. Fig. 4 depicts the family-tree of ML algorithms.

A. Supervised Learning

In this subsection, four typical supervised learning algorithms are summarized, namely, the k-nearest neighbour, support vector machine, random forest, and linear discriminant analysis.

a) K-Nearest Neighbour (k-NN): The k-nearest neighbour (k-NN) clustering belongs to the group of *lazy learning* algorithms because it does not require any training process. Instead, it only uses its training data to classify new data points when those new data points are entered. Due to the lack of training, k-NN has to store all training data points for use, which makes it a computationally inefficient algorithm.

In principle, when classifying a testing data point $\mathbf{x}_{test}$, the k-NN algorithm considers the k nearest neighbours of $\mathbf{x}_{test}$ and then assigns $\mathbf{x}_{test}$ to the class based on majority voting [85]–[88]. As illustrated in Fig. 5, three out of four nearest neighbours of a testing instance belong to the class (+1), hence

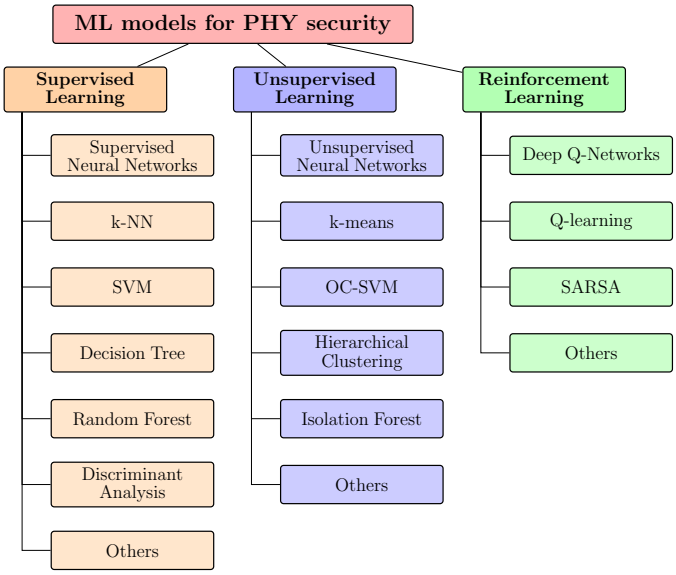


Fig. 4. The scope of this survey: Using ML models for detecting eavesdropping attacks and designing security strategies. A simplified anatomy of several ML models is also presented.

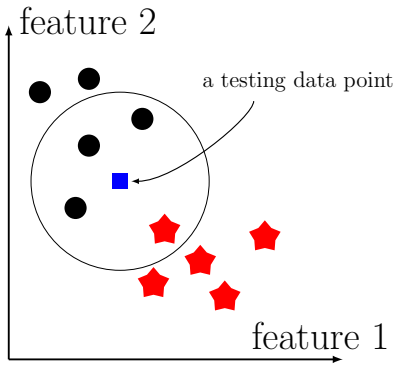


Fig. 5. An illustration of how k-NN classifies a new data point.

the testing instance will be classified as (+1). Since the k-NN algorithm processes all data points in the training data, it will become particularly inefficient when dealing with large datasets.

b) Support Vector Machine: The SVM is a powerful type of ML, which can efficiently perform both linear and non-linear classification tasks when the size of the data set is small or medium. The goal of SVM is to find the optimal boundary that separates 2 different classes in a dataset. Soft-margin SVM offers more versatility than hard-margin SVM, because the soft-margin SVM accepts some misclassified data points, thus avoiding over-fitting. Given that the equation of a hyperplane has the form $\hbar(\mathbf{x}) = \mathbf{x}^T\mathbf{a} + b = 0$, the goal of SVM is find the optimal values of $\mathbf{a}$ and b through solving the following optimization problem:

$$\min_{\mathbf{a},b,\xi_t} \quad \underbrace{\frac{1}{2}\|\mathbf{a}\|^2}_{\text{regularizer}} + C\underbrace{\sum_{t=1}^{T}\xi_t^{\kappa}}_{\text{error}} \tag{10a}$$

$$\text{s.t.} \quad y_{\text{label}}^{[t]}\left(\mathbf{a}^\top \mathbf{x}_{\text{input}}^{[t]} + b\right) \geq 1 - \xi_t \tag{10b}$$

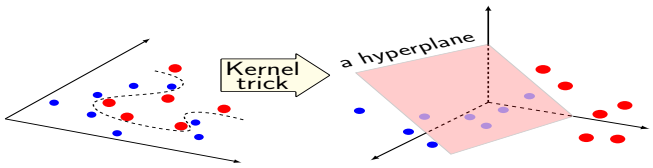


Fig. 6. The use of the kernel trick in SVM classifiers can bring the data to a higher-dimensional space, thereby making the data more separable.

where C is the margin parameter in (10a). $\kappa = 1$ implies the 1-norm (L1) soft-margin SVM, while $\kappa = 2$ implies the 2-norm (L2) soft-margin SVM.

When using SVM, a "magic" technique called *kernel trick* may be exploited to transform the data from the original input space into a higher-dimensional space. The reason behind the use of the kernel trick is that the data in a higher-dimensional space may be linearly separable, which was not separable in the original space [89]–[91]. Fig. 6 illustrates how the kernel trick works, where a dataset with two features becomes more separable, when being transformed into a new space with three features. Let $\phi(\cdot)$ be a nonlinear mapping that stretches the original input space $\mathcal{X}_{\text{input}}$ to some higher-dimensional space $\mathcal{H}$. For the t-th data point $\mathbf{x}_{\text{input}}^{[t]}$, we have the corresponding value of $\phi_t = \phi\left(\mathbf{x}_{\text{input}}^{[t]}\right)$. Then the inner product $\mathcal{K}(t,t') = \phi_t\phi_{t'}^\top$ is a kernel function. When using the kernel trick, we will encounter the calculation of $\mathcal{K}(t,t')$ instead of $\phi(\mathbf{x}_{\text{input}}^{[t]})$ and $\phi(\mathbf{x}_{\text{input}}^{[t']})$. This alleviates the computational cost because it is not necessary to calculate explicit expressions for $\phi(\mathbf{x}_{\text{input}}^{[t]})$ and $\phi(\mathbf{x}_{\text{input}}^{[t']})$. Normally, kernel functions are predefined and there are many types of kernels to choose from, such as linear, radial basis function, polynomial, and sigmoid kernels.

c) Random Forest: A "*random forest*" is basically a collection of decision trees. A decision tree is built top-down from a root node. The leaf nodes (i.e., the terminal nodes) of the tree represent decisions (i.e., labels). A decision node of the tree represents the test of a specific feature. Note that the topmost decision node is the root node. Each decision node has at least two branches each representing an outcome of the corresponding test. Fig. 7 depicts a decision tree having basic elements. To construct decision trees, we can use different algorithms. Among widely-used algorithms are ID3 (i.e., iterative dichotomiser 3), CART, CHAID, MARS and so on [92], [93]. Let us consider the ID3 algorithm that uses entropy and information gain to construct a decision tree based on measuring/scoring the features of data. Based on the features that have been scored, a decision tree will be built top-down as follows:

- Step 1: At a decision node, select the highest-score feature for a test.
- Step 2: From that decision node, create branches and split the training data into subsets so that each subset corresponds to a branch.
- Step 3: Find leaf nodes and repeat the steps recursively on each subset.

To build a random forest from decision trees, we first need to

This article has been accepted for publication in IEEE Communications Surveys & Tutorials. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/COMST.2024.3363639

t	Φ_1	Φ_2	Label
1	0.2	0.9	$+1$
2	0.3	0.4	-1
3	0.75	0.8	-1
4	0.8	0.5	$+1$

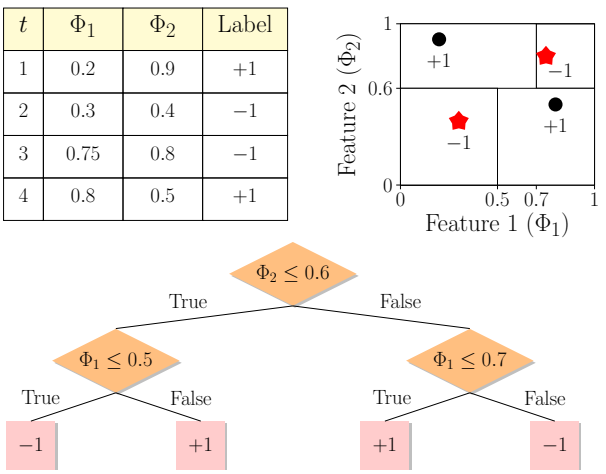


Fig. 7. An example of building a decision tree from a dataset.

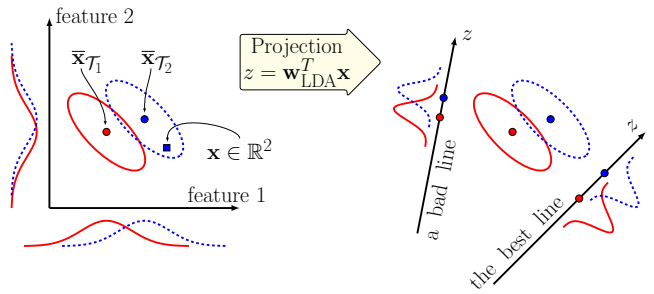


Fig. 8. An illustration of how linear discriminant analysis can help with separating a sample from the other. In this figure, we assume that the training data has two features. A new feature will be created to replace the old ones by using a projection onto a certain line. Using linear discriminant analysis allows us to find the best line $z = \mathbf{w}_{\text{LDA}}^T \mathbf{x}$. Each ellipse represents the confidence region of a distribution [96], [97].

create *bootstrapped datasets* from the training data. By using a statistical technique called *bootstrapping*, some data points in the training set will be randomly picked in order to create a bootstrapped dataset. Thus, a bootstrapped dataset is a subset of the training data. Note that the number of bootstrapped datasets is equal to the number of decision trees in a random forest. Moreover, a certain data point may belong to many bootstrapped datasets because the sampling allows a data point to be picked many times. This sampling method is known as *sampling with replacement*. Once the bootstrapped datasets have been prepared, we can build the decision trees each corresponding to a bootstrapped dataset. Then using *majority voting* [94], a new data point will be predicted.

d) Linear Discriminant Analysis (LDA): Linear discriminant analysis (LDA) falls into the family of discrimination techniques. The purpose of discrimination is to make the data as separable as possible [95, Ch. 11]. Let us assume that a pair of two samples belonging to labels $(+1)$ and (-1) obeys two different statistical distributions with covariance matrices $\mathbf{\Sigma}_{theory,1}$ and $\mathbf{\Sigma}_{theory,2}$. In order to use linear discriminant analysis, it is required that $\mathbf{\Sigma}_{theory,1}$ is equal to $\mathbf{\Sigma}_{theory,2}$. In practice, $\mathbf{\Sigma}_{theory,1}$ and $\mathbf{\Sigma}_{theory,2}$ are normally unknown, thus we replace them with the *sample* covariance matrices $\mathbf{\Sigma}_{sample,1}$ and $\mathbf{\Sigma}_{sample,2}$, which are formed by the available samples in our training data (see Subsection II-B2). The formulation of a sample covariance is presented in (6) in Subsection II-A6. Accordingly, $\mathbf{\Sigma}_{sample,1}$ and $\mathbf{\Sigma}_{sample,1}$ are expected to be equal in order to use linear discriminant analysis. At this point, a problem arises from the fact that $\mathbf{\Sigma}_{sample,1}$ and $\mathbf{\Sigma}_{sample,2}$ may not be equal, i.e., $\mathbf{\Sigma}_{sample,1} \neq \mathbf{\Sigma}_{sample,2}$. To overcome this problem, we use a weighted average of $\mathbf{\Sigma}_{sample,1}$ and $\mathbf{\Sigma}_{sample,2}$, which is called the pooled sample covariance matrix. This matrix can be expressed as follows [70, Ch.5], [95]:

$$\mathbf{\Sigma}_{pooled} = \frac{(|\mathcal{T}_1| - 1)\,\mathbf{\Sigma}_{sample,1} + (|\mathcal{T}_2| - 1)\,\mathbf{\Sigma}_{sample,2}}{|\mathcal{T}_1| + |\mathcal{T}_2| - 2}, \tag{11}$$

where $|\mathcal{T}_1|$ and $|\mathcal{T}_2|$, defined in eqs.(8)–(9), are the number of samples in the class $(+1)$ and that in the class (-1), respectively. Then, we use $\mathbf{\Sigma}_{pooled}$ as the single common covariance matrix for the two populations of interest.

In theory, LDA seeks some transformation vector $\mathbf{w}_{\text{LDA}}$, which is used to project a data point $\mathbf{x}$ in the input space onto a line $z = \mathbf{w}_{\text{LDA}}^T \mathbf{x}$, so that the function

$$\lambda_{\text{LDA}} = \frac{\mathbf{w}_{\text{LDA}}^T \mathbf{S}_{\text{between}} \mathbf{w}_{\text{LDA}}}{\mathbf{w}_{\text{LDA}}^T \mathbf{S}_{\text{within}} \mathbf{w}_{\text{LDA}}} \tag{12}$$

is maximized. In (12), $\mathbf{S}_{\text{within}}$ is the within-class (scatter) matrix, while $\mathbf{S}_{\text{between}}$ is the between-class (scatter) matrix.[2] These matrices can be expressed as [70], [98]

$$\mathbf{S}_{\text{within}} = (|\mathcal{T}_1| + |\mathcal{T}_2| - 2)\,\mathbf{\Sigma}_{pooled}, \tag{13}$$

$$\mathbf{S}_{\text{between}} = \sum_{j=1}^{2} |\mathcal{T}_j| \left(\overline{\mathbf{x}}_{\mathcal{T}_j} - \overline{\mathbf{x}}\right) \left(\overline{\mathbf{x}}_{\mathcal{T}_j} - \overline{\mathbf{x}}\right)^T, \tag{14}$$

where $\overline{\mathbf{x}}_{\mathcal{T}_1}$ and $\overline{\mathbf{x}}_{\mathcal{T}_2}$ are the sample mean of the class $(+1)$ and that of the class (-1), respectively. The expressions for $\overline{\mathbf{x}}_{\mathcal{T}_1}$ and $\overline{\mathbf{x}}_{\mathcal{T}_2}$ are are defined in (5), while $\overline{\mathbf{x}}$ is the sample mean of the whole training data and it is defined as $\overline{\mathbf{x}} = \frac{1}{T}\left(|\mathcal{T}_1|\overline{\mathbf{x}}_{\mathcal{T}_1} + |\mathcal{T}_2|\overline{\mathbf{x}}_{\mathcal{T}_2}\right)$.

Finding the maximal value of λ_{LDA} leads to $\mathbf{w}_{\text{LDA}} = \mathbf{S}_{\text{within}}^{-1}\left(\overline{\mathbf{x}}_{\mathcal{T}_1} - \overline{\mathbf{x}}_{\mathcal{T}_2}\right)$. Observe from (12) that λ_{LDA} is also an eigenvector of the matrix $\mathbf{S}_{\text{LDA}} \overset{def}{=} \mathbf{S}_{\text{within}}^{-1}\mathbf{S}_{\text{between}}$. Accordingly, LDA can be intuitively interpreted as a method in which the data points in the input data will be projected onto the *largest* eigenvector of $\mathbf{S}_{\text{LDA}}$. Fig. 8 illustrates how LDA projects the two samples in the original 2-dimensional space onto the largest eigenvector of $\mathbf{S}_{\text{LDA}}$.

By using LDA, a new data point $\mathbf{x}_{new}$ in the test data will be classified as the first class if the inequality

$$\begin{aligned}&\left(\overline{\mathbf{x}}_{\mathcal{T}_1} - \overline{\mathbf{x}}_{\mathcal{T}_2}\right)^T \mathbf{\Sigma}_{pooled}^{-1} \mathbf{x}_{new} \\ &\geq \frac{1}{2}\left(\overline{\mathbf{x}}_{\mathcal{T}_1} - \overline{\mathbf{x}}_{\mathcal{T}_2}\right)^T \mathbf{\Sigma}_{pooled}^{-1}\left(\overline{\mathbf{x}}_{\mathcal{T}_1} + \overline{\mathbf{x}}_{\mathcal{T}_2}\right) + \ln\left(\frac{c(1|2)p_2}{c(2|1)p_1}\right)\end{aligned} \tag{15}$$

holds. Herein, $c(1|2)$ and $c(2|1)$ represent the costs of misclassification, while p_1 and p_2 are the prior probabilities of the

[2]The between-class matrix is different from the between-class covariance matrix. To obtain the latter, we can divide the former by $|\mathcal{T}_1| + |\mathcal{T}_2| - 1$.

first class and the second class, respectively. The left hand side of (15) is a point on the line, which the largest eigenvector passes through. Meanwhile, the right hand side of (15) is a given threshold that is used for binary classification. Note that the principle of LDA can also be extended to multi-class classification [99], [98, Ch.6].

B. Unsupervised Classification Algorithms

In this subsection, four typical unsupervised learning algorithms are summarized, namely K-means clustering, one-class support vector machine, isolation forest, and hierarchical clustering.

a) K-Means Clustering (K-means): K-means clustering acts directly on the testing data without requiring a training process [85, Ch. 9]. The benefit of K-means clustering is that it can cluster the data into K clusters. Using K-means clustering for detecting eavesdropping attacks, we may only have to classify the data into two clusters by setting $K = 2$, where one cluster corresponds to eavesdropping attacks, and the other to no attack.

In general, K-means clustering yields K clusters each having a centroid. At the beginning, K-means may generate its centroids randomly and then updates the positions of the centroids iteratively. At each iteration, each data point will be assigned to the closest centroid. Once all data points have been assigned to one of the centroids, the newly-created clusters may or may not be different from the earlier clusters. If the new clusters are not the same as the old clusters, it is necessary to update the centroids. The update rule of a centroid is based on calculating the average value of all the data points in the corresponding cluster. After the centroids have been updated, the iterative algorithm moves on to the next iteration. The algorithm will continue its iterations until there is no change to the centroids or the maximum number of iterations is reached.

The accuracy of K-means clustering relies heavily on the initialization of centroids. Thus, different initializations result in different accuracy levels. Instead of randomly generating the initial centroids, advanced techniques suggest selecting the positions of initial centroids more carefully. For example, K-means++ exhibits its superiority over the standard K-means clustering in terms of both speed and accuracy [100]. In essence, the only difference between K-means++ and K-means clustering lies in the initialization. To be more precise, K-means++ starts with choosing a centroid randomly before calculating the distances of points in the dataset from the selected centroid. Then the next centroids are found based on considering a functional relationship of those distances.

b) One-class Support Vector Machine (OC-SVM): As a variant of SVM, the one-class support vector machine (OC-SVM) has been developed to deal with outliers or imbalanced data. According to [84], [101], [102], the goal of OC-SVM is to evaluate the decision function

$$\hbar(\mathbf{x}) = \operatorname{sign}\left(\mathbf{w}^T \phi(\mathbf{x}) - \rho\right) \tag{16}$$

where $\operatorname{sign}(\cdot)$ is the sign function, $\phi(\cdot)$ is a mapping that stretches the input space $\mathcal{X}_{\text{input}}$ to a higher-dimensional space,

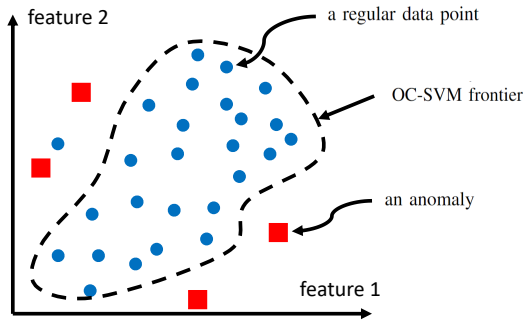


Fig. 9. An illustration of the OC-SVM algorithm.

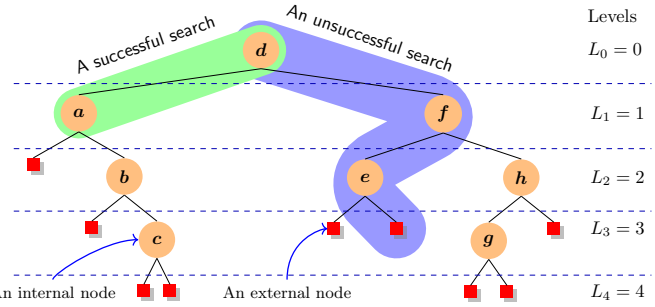


Fig. 10. An illustration of a binary search tree with $n_{int} = 8$ internal nodes ($a < b < c < d < e < f < g < h$). While a successful search terminates at an internal node (e.g., searching for b), an unsuccessful search terminates at an external node (e.g., searching for x given that $e < x < f$). $\texttt{IPL} = 1L_0 + 2L_1 + 3L_2 + 2L_3 + 0L_4 = 14$ and $\texttt{EPL} = \texttt{IPL} + 2n_{int} = 30$.

while $\mathbf{w}$ and ρ are found by solving the optimization problem:

$$\underset{\mathbf{w},\rho,\xi_i}{\text{minimize}} \quad \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{\text{regularizer}} + \underbrace{\frac{1}{\nu T_{tot}} \sum_{w=1}^{T_{tot}} \xi_w - \rho}_{\text{error}} \tag{17a}$$

$$\text{subject to} \quad \langle \mathbf{w}, \phi(\mathbf{x}_w) \rangle \geq \rho - \xi_w. \tag{17b}$$

Herein, $\xi_w \geq 0$ is a slack variable, and $\nu \in (0, 1]$ is a parameter balancing the maximal distance from the origin and the number of data points in the region created by the hyperplane [84].

In OC-SVM, any data points that cannot form a dense cluster will be treated as outliers/anomalies [102]. In other words, the outliers reside within the regions of low density. In the meantime, regular data points will be treated as the inliers because they form the regions of high density. Fig. 9 depicts the idea behind the use of OC-SVM.

c) Isolation Forest: An isolation forest (iForest) is an ensemble method, whose anomaly scores are averaged over multiple isolation trees. The structure of an isolation tree is very similar to that of a binary search tree (BST). To understand the concept of an isolation tree, we first recall the BST structure.

Remark 1. The BST structure: *A BST is a binary tree having the following properties:*

- *Each node in a BST has at most 2 child nodes. Each node in a BST has a unique value (or key); denote the value*

This article has been accepted for publication in IEEE Communications Surveys & Tutorials. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/COMST.2024.3363639

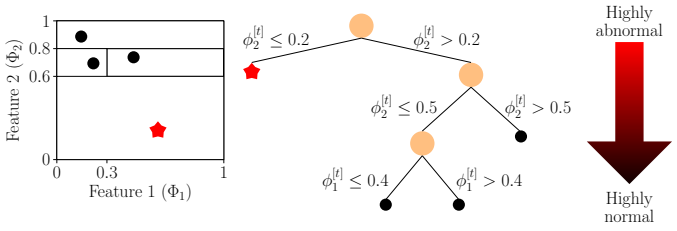


Fig. 11. Given four instances $\{\mathbf{x}_{\text{input}}^{[t]}\}_{t=1}^{4}$, the iForest algorithm can *isolate* them in four external nodes (or leaf nodes) of an isolation tree. Multiple isolation trees will then be averaged to become a single tree called iForest. The depth of iForest determines whether an instance is considered as abnormal or normal.

of node X by $\texttt{Val}_X$.

- *Let Y be a child node of X. If $\texttt{Val}_Y < \texttt{Val}_X$ (or $\texttt{Val}_Y > \texttt{Val}_X$), then Y is the left (or right) sub-tree of X.*

Let us denote the number of *internal* nodes in a BST by n_{int}, the *internal* path length by IPL, and the *external* path length by EPL. According to [103], we have $\texttt{EPL} = \texttt{IPL} + 2n_{int}$. By averaging over many BSTs, the average external path length of a binary search tree is calculated as $\overline{\texttt{EPL}} = \overline{\texttt{IPL}} + 2n_{int}$, where $\overline{\texttt{IPL}}$ is the average internal path length of a binary search tree. For n_{int} internal nodes, a BST will have $n_{ext} = n_{int} + 1$ external nodes. Thus, the average external path length of an external node can be calculated as

$$\overline{\texttt{EPL}}_0 = \frac{\overline{\texttt{EPL}}}{n_{ext}} = \frac{\overline{\texttt{IPL}}}{n_{ext}} + \frac{2n_{int}}{n_{ext}} = 2\sum_{k=1}^{n_{int}} \frac{1}{k} + \frac{2n_{int}}{n_{int}+1}. \quad (18)$$

Herein, $\overline{\texttt{EPL}}_0$ is a function of n_{int} (or equivalently, n_{ext}). Furthermore, $\overline{\texttt{EPL}}_0$ can be viewed as the depth per an external node and the depth is averaged over all binary search trees.

Using the above idea of BSTs, the authors of [83] suggest building isolation trees. Yet another aspect is that the iForest algorithm exploits the idea of the unsuccessful search events in a binary search tree to define the path length $h(\mathbf{x})$ of a data point $\mathbf{x}$. By definition, an unsuccessful search traverses a path from the root node to an external node (see Fig. 10 for illustration). Similarly, in the context of an isolation tree, the path length $h(\mathbf{x})$ is defined as *the number of edges* that $\mathbf{x}$ "traverses from the root node until the traversal is terminated". Using sub-sampling without replacement, the original dataset is divided into NoT sub-datasets, each being associated with an isolation tree. Thus there are NoT isolation trees in total. To find abnormal data points, [83] defines the so-called anomaly score as follows:

$$\text{score}(\mathbf{x}, \psi) = 2^{-\mathbb{E}\{h(\mathbf{x})\}/c(\psi)}, \quad (19)$$

where ψ is the sub-sampling size, $c(\psi) = \overline{\texttt{EPL}}_0\big|_{n_{int}=\psi}$ is a constant, $\mathbb{E}\{h(\mathbf{x})\}$ is the average path length of $h(\mathbf{x})$ and it is averaged over all NoT isolation trees. After calculating the score of each data point, we sort all the scores in descending order. Given the sorted array of scores, we can show that the first top scores correspond to anomalies.

d) Hierarchical Clustering: In contrast to K-means clustering, which has to predetermine the number of clusters/-groups, hierarchical clustering does not require that number

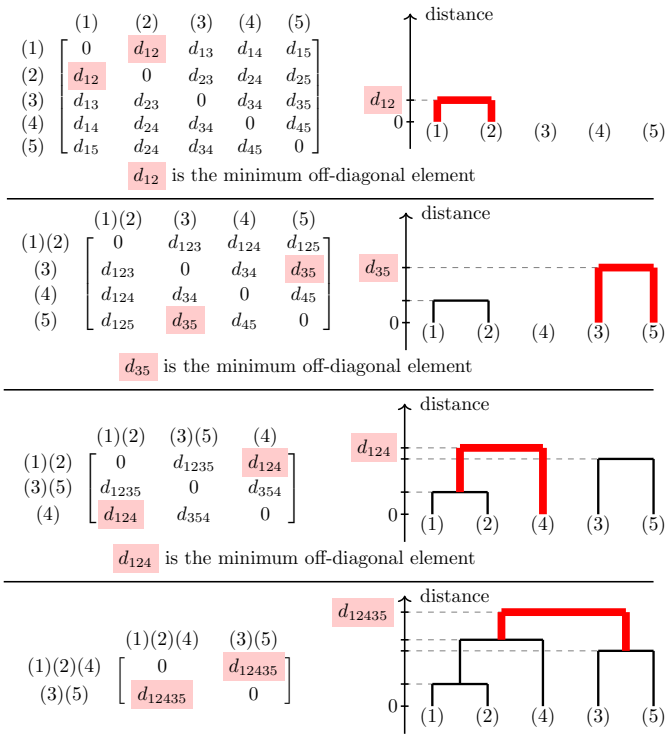


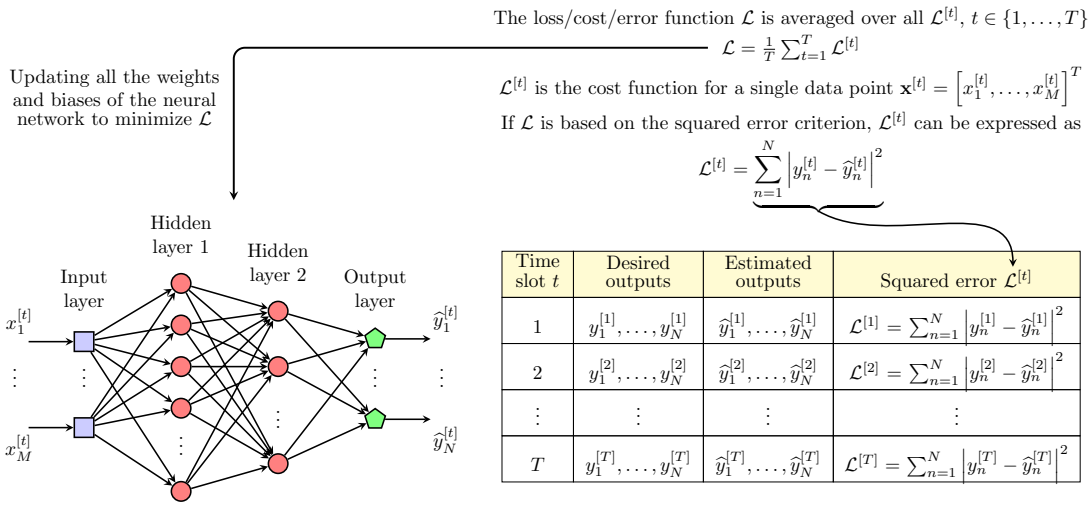
Fig. 12. A 4-step illustration of the agglomerative hierarchical clustering. (t) represents $\mathbf{x}_{\text{input}}^{[t]}$ for $t \in \{1, 2, 3, 4, 5\}$. In each step, we find the minimum off-diagonal element in the proximity matrix and group related examples.

in advance. Instead, the number of clusters can be determined after a hierarchical dendrogram has been formed [104], [105]. There are two common types of hierarchical clustering, namely agglomerative and divisive hierarchical clustering:

- In principle, the agglomerative approach is a bottom-up approach, which considers each data point as a separate cluster and then merges two clusters at each step until all clusters are unified as a single one.
- By contrast, the divisive approach is performed in a top-down manner, where the whole data is considered as a single cluster at the beginning, and then divides the data into smaller clusters until each data point becomes a cluster of its own.

Fig. 12 presents four steps in the process of constructing a simple dendrogram on the basis of the *agglomerative* hierachical clustering. At each step, we update a proximity matrix and use this new matrix to decide, which two clusters will be merged. A proximity matrix is also known as a distance matrix that is symmetric. Each off-diagonal element in a proximity matrix represents the *dissimilarity* between two distinct clusters. In short, the agglomerative approach uses distance measures to quantify to what degree a group/cluster/example is dissimilar to another. Note that distance measures are also applied to the divisive approach.

Given a proximity matrix at each step, it is necessary to have a criterion for deciding which two clusters will be merged. There are many different criteria, which are also known as *linkage* criteria. In so-called single linkage (or nearest neighbour) clustering, we aim to find two clusters that



Time slot t	Desired outputs	Estimated outputs	Squared error $\mathcal{L}^{[t]}$
1	$y_1^{[1]},\ldots,y_N^{[1]}$	$\widehat{y}_1^{[1]},\ldots,\widehat{y}_N^{[1]}$	$\mathcal{L}^{[1]} = \sum_{n=1}^{N}\left\|y_n^{[1]} - \widehat{y}_n^{[1]}\right\|^2$
2	$y_1^{[2]},\ldots,y_N^{[2]}$	$\widehat{y}_1^{[2]},\ldots,\widehat{y}_N^{[2]}$	$\mathcal{L}^{[2]} = \sum_{n=1}^{N}\left\|y_n^{[2]} - \widehat{y}_n^{[2]}\right\|^2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
T	$y_1^{[T]},\ldots,y_N^{[T]}$	$\widehat{y}_1^{[T]},\ldots,\widehat{y}_N^{[T]}$	$\mathcal{L}^{[T]} = \sum_{n=1}^{N}\left\|y_n^{[T]} - \widehat{y}_n^{[T]}\right\|^2$

Fig. 13. A neural network with two hidden layers.

are closest to each other and merge them into a new cluster. Fig. 12 is an example of single linkage in which we find the minimum off-diagonal element in a proximity matrix at each step, then the corresponding row and column of that element will identify a pair of two clusters needed to be merged. In contrast to single linkage, complete linkage considers the maximum distance, which is determined by the most distant nodes in two clusters. Furthermore, there are other linkage criteria, such as the average linkage [106] and Ward linkage [107].

C. Neural Networks

Fig. 13 describes the basic architecture of a multiple layer perceptron (MLP) having three layers (including an input layer, a hidden layer and an output layer). Each layer has at least one neuron, which can connect all the neurons of the previous layer to all the neurons of the next layer. A detailed investigation of the related mathematical expressions that describe the relationship between the input and output of a neuron and the relationships among neurons can be found for example in [108]. The relationship between the input and output of the whole NN can be described by a function of the form $\widehat{y}_n = f_n(x_1,\ldots,x_M|\mathcal{W},\mathcal{B})$, where $x_1,\ldots,x_M$ are variables that constitute the input of an NN, $\mathcal{W}$ is the set of link weights, and $\mathcal{B}$ is the set of biases at the neurons. The goal of the network is to find $\mathcal{W}$ and $\mathcal{B}$ ensuring that the functions $\{f_n(x_1,\ldots,x_M|\mathcal{W},\mathcal{B})\}_{n=1}^{N}$ can approximate and generalize the relationship between $\{x_1,\ldots,x_M\}$ and $\{y_1,\ldots,y_N\}$, as long as the difference between the network output (i.e., $\{\widehat{y}_1,\ldots,\widehat{y}_N\}$) and the *actual* data (i.e., $\{y_1,\ldots,y_N\}$) is within some tolerance. This tolerance difference is quantified by some loss function (known as error/cost/objective function).

Since the loss function quantifies how well an NN works, it will be calculated at the output of the network. There are various types of cost functions each resulting in a different performance [109]–[111]. Depending on the assessment criteria, different types of cost functions may be preferred over others. In general, the goal of an NN is to minimize the loss function by appropriately adjusting the network parameters. This often leads to solving a minimization problem (whose objective function is the loss function) with respect to the network parameters (note that these parameters are in the sets $\mathcal{W}$ and $\mathcal{B}$ as mentioned above). Since the network parameters stick with the operation of neurons, the choice of activation functions for each and every neuron deserves special consideration.

In general, an MLP relies on neurons each being activated by a certain activation function $\sigma(\cdot)$. In fact, the activation functions used in a neural-network-based classifier have direct effects on the classification performance [112], [113]. In an NN, it is theoretically possible to combine different types of activation functions, including but not limited to a binary step, linear, sigmoid, tanh and ReLU functions. Furthermore, there is also a huge range of other activation functions, such as the maxout [114], softmax [115], piecewise linear [116], and distance-based activation functions [117], just to name a few.

As for the input layer, the number of neurons should be equal to the number of features in the data input. For example, we may wish to create an MLP, which allows a column vector $\mathbf{x}_{\text{input}}^{[t]}$ to pass through. Then, the number of input neurons should be equal to the length of $\mathbf{x}_{\text{input}}^{[t]}$. Recall that $\mathbf{x}_{\text{input}}^{[t]} = [x_1^{[t]},\ldots,x_M^{[t]}]^T$ is the t-th data point, and each element in $\mathbf{x}_{\text{input}}^{[t]}$ represents a feature (see Section II for more details). Regarding the output layer, a single neuron may be capable performing a binary classification task. In the case of more complex multi-class classification, using a single neuron at the output layer may not be sufficient for a confident decision. Indeed, it is plausible that multiple output neurons are necessary for making multi-class decision.

D. Reinforcement Learning

Reinforcement learning enables a system to interact with the environment and learn from these interactions. Normally, reinforcement learning problems can be formalized by the framework of a Markov decision process [18], [118].[3] Intu-

[3]Markov decision processes are used for decision making.

itively, reinforcement learning considers a certain agent that learns a policy. The agent first observes its current state and then it takes an action. Based on the action to be taken, the environment gives the agent some feedback based on the new state and the reward. Using the feedback obtained, the agent will take its next action. This process is iteratively executed many times to find the optimal policy that maximizes the expected total reward in the long run.

Some of the popular reinforcement learning algorithms are Q-learning [119], SARSA [120] and deep Q-learning [121]. While Q-learning operates in an offline fashion that attains the optimal policy after the whole algorithm converges, SARSA is an online learning-based algorithm that can find the optimal action at each individual iteration [118]. Hence, Q-learning may be more suitable for a small space of states and actions. On the other hand, deep Q-learning, which relies on the combination of reinforcement learning and NNs, relies on a deep Q-network for solving high-dimensional problems [122].

E. Other Advanced ML Algorithms Learning

a) Variational Bayesian Learning: Let $\mathbf{x}$ represent a data sample and $\mathbf{z}$ a vector of parameters. Using Bayes' theorem, we have the following *aposteriori* probability:

$$\underbrace{p(\mathbf{z}|\mathbf{x})}_{\text{posterior}} = \underbrace{p(\mathbf{x}|\mathbf{z})}_{\text{likelihood}} \times \underbrace{p(\mathbf{z})}_{\text{prior}} / \underbrace{p(\mathbf{x})}_{\text{evidence}} . \tag{20}$$

If $\mathbf{z}$ is continuous, then the denominator $p(\mathbf{x})$ can be calculated as $p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) d\mathbf{z}$. According to Jensen's inequality, we have $\log\left(\mathbb{E}_{p(\mathbf{x})}\left\{f(\mathbf{x})\right\}\right) \geq \mathbb{E}_{p(\mathbf{x})}\left\{\log\left(f(\mathbf{x})\right)\right\}$. On the other hand, $\log\left(p(\mathbf{x})\right)$ can be written as the logarithm of an expectation, i.e.

$$\begin{aligned}\log\left(p(\mathbf{x})\right) &= \log\left(\int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) d\mathbf{z}\right) = \log\left(\int_{\mathbf{z}} Q(\mathbf{z}) \frac{p(\mathbf{x}, \mathbf{z})}{Q(\mathbf{z})} d\mathbf{z}\right) \\ &= \log\left(\mathbb{E}_{Q(\mathbf{z})}\left\{\frac{p(\mathbf{x}, \mathbf{z})}{Q(\mathbf{z})}\right\}\right). \end{aligned} \tag{21}$$

Thus, applying Jensen's inequality to $\log\left(p(\mathbf{x})\right)$, we arrive at:

$$\underbrace{\log\left(p(\mathbf{x})\right)}_{\text{evidence}} \geq \underbrace{\mathbb{E}_{Q(\mathbf{z})}\left\{\log\left(\frac{p(\mathbf{x}, \mathbf{z})}{Q(\mathbf{z})}\right)\right\}}_{\text{evidence lower bound (ELBO)}} \triangleq L(\mathbf{z}). \tag{22}$$

More importantly, the difference between the log marginal probability of $\mathbf{x}$ and the ELBO turns out to be the Kullback-Leibler (KL) divergence [123] of

$$\log\left(p(\mathbf{x})\right) - L(\mathbf{z}) = \mathrm{KL}\left[Q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\right]. \tag{23}$$

The KL divergence is widely used for quantifying how much a probability distribution differs from another one. Hence $\mathrm{KL}\left[Q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\right]$ quantifies the difference/dissimilarity, between the *variational* probability distribution $Q(\mathbf{z})$ and the *true aposteriori probability* $p(\mathbf{z}|\mathbf{x})$. In this context maximizing the ELBO is equivalent to minimizing the KL divergence. The core idea here is to find some tractable distribution $Q(\mathbf{z})$ that approximates the *aposteriori* distribution $p(\mathbf{z}|\mathbf{x})$. Based on this, sophisticated variational Bayes (VB) ML algorithms have been proposed, which can be categorized into mean-field VB and fixed-form VB [124].

Some of the most popular VB ML algorithms are constituted by the family of *variational auto-encoders* (VAEs) proposed in the pioneering contribution [125]. A typical VAE includes an encoder that yields an approximate *aposteriori* distribution $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, and a decoder that yields a likelihood distribution $p_{\boldsymbol{\phi}}(\mathbf{x}|\mathbf{z})$. Herein, $\boldsymbol{\theta}$ denotes all the weights of the encoder, while $\boldsymbol{\phi}$ represents all the weights of the decoder. By training the VAE, the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are optimized so that the approximate *aposteriori* distribution $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ becomes similar to the true *aposteriori* distribution.

b) Mixture of Experts: When the data volume becomes large, it is necessary for an AI-aided system to be able to scale up its training models. One of the popular methods of scaling up AI models is the so-called mixture of experts (MOEs) technique [126], [127]. Briefly, the MOE method is a neural-network-based ensemble learning technique. Similar to other ensemble methods, the MOE is based on the 'divide-and-conquer' principle. Herein, the underlying idea behind the MOE is that the global input space is divided into smaller sub-spaces each being assigned to a specific local model for training. Then all the outputs of the local models will be combined. Note that each local model is termed as an expert. Furthermore, the MOE consists of an NN, referred to as the gating network, that supervises the 'divide-and-conquer' process.

Fig. 14 illustrates the partitioning of a dataset into two smaller datasets that will then be trained by a pair of local experts in support of the classification task. This results in beneficial specialization, because each local model (i.e., each expert) will specialize in different small tasks by training and learning based on different small input spaces. Based on the outputs of the experts, a gating network will be trained based on the global input data. Herein, the role of each expert is to learn from a local input space, while the role of the gating network is to learn from the outputs of experts. Indeed, the MOE functions in a 'divide-and-conqure' manner so that the input space may be beneficially divided into subsets - each being assigned to a suitably specialized expert for training.

c) Meta-learning: When it comes to learning from prior experience in the context of ML, meta-learning emerges as a compelling candidate, since it represents the concept of "*learning to learn*" [128], [129]. Naturally, we do not always have to learn new skills from scratch once we have suceeded in learning some other skills. In other words, after learning or inferring a specific skill from a task, it becomes easier to learn another skill based on a new task, because we "*learn how to learn*" by relying on skills transferred across tasks [129].

To elaborate a little further by describing the concept of meta-learning mathematically, let us consider a set of previous tasks $\{t_1, t_2, \ldots, t_M\}$, a set of parameter vectors $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_N\}$, and a set of system performance evaluations $E_{\text{previous}} = \{e_{11}, e_{12}, \ldots, e_{mn}, \ldots, e_{MN}\}$ with $m \in \{1, \ldots, M\}$, $n \in \{1, \ldots, N\}$. Herein, a task t_m can be also be viewed as a learning algorithm, e.g., an NN based learning processed harnessed for optimizing some function. In this context each parameter vector $\boldsymbol{\theta}_n$ represents a specific configuration of an ML model, e.g., it represents all the weights of an NN. Finally, each value e_{mn} is a value of the

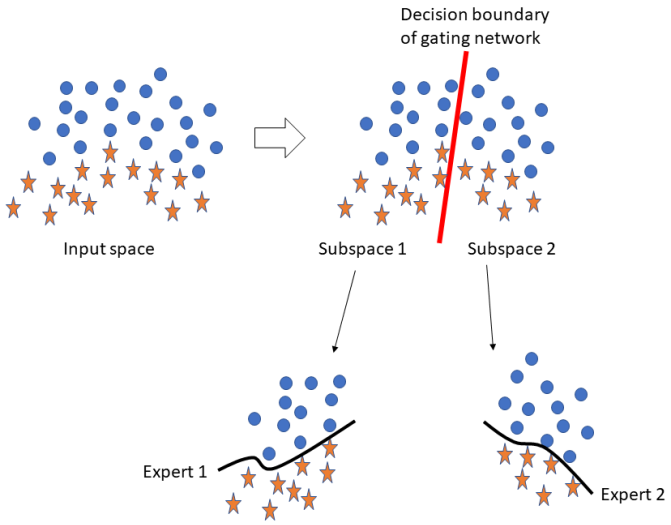


Fig. 14. An illustration of using MOE for classification.

function $e(t, \boldsymbol{\theta})$ evaluated at $t = t_m$ and $\boldsymbol{\theta} = \boldsymbol{\theta}_n$, e.g., e_{mn} can be the accuracy of an ML system. Let us assume now that we want to predict a new parameter vector $\boldsymbol{\theta}_{\text{next}}$ for the new task t_{next}, given that the new set of performance evaluations - namely $E_{\text{next}} = \{e_{\text{iter},1}, e_{\text{iter},2}, \ldots\}$ - can be obtained in an iterative manner during the training. Based on the concept of meta learning, we will train a new ML model based on the meta data $E_{\text{previous}} \cup E_{\text{next}}$. For further details, please refer to [128], [129], for example.

d) Self-supervised learning (SSL): In contrast to supervised learning that relies on labelled data, self-supervised learning (SSL) does not require the data to be labelled and thus it is normally referred to as a branch of unsupervised learning. However, while most of the popular unsupervised learning algorithms aim for detecting the pattern of data and clustering datasets, SSL is centered on recovering and self-generating the data [130]. It may also be observed that SSL algorithms tend to be constructed based on the architecture of NNs.

The architecture of SSL may be divided into two main categories: i) Generative SSLs and ii) Contrastive SSLs. As for generative SSLs, an encoder is used for encoding an input sample x iinto z, and then a decoder will process z to reconstruct x. A typical model using generative SSLs is constituted by the auto-encoder (AE) [131]. On the other hand, as for contrastive SSLs, after decoding x into z, the similarity/dissimilarity will be quantified for differentiating the input sample from others [132]. According to [130], there are also hybrids of generative and contrastive solutions, including the family of generative adversarial networks (GANs). Following their introduction in 2014 [133], GANs have received substantial attention as a benefit of their applicability in numerous fields. In the architecture of a GAN, a generator and a discriminator are simultaneously trained in a competing manner [134]. While the generator is trained for generating new samples, the discriminator is trained for differentiating the actual sample and the generated (fake) samples. Herein, the goal of a GAN is to confuse the discriminator in distinguishing the actual samples from fake ones and to allow the generator to create fake-but-plausible data that even humans find hard to realize.

e) Explainable AI: Despite the success of recent ML algorithms, there are still concerns about applying ML in a variety tasks, such as decision-making and recommender systems. Sometimes it is hard to interpret the results returned by an ML algorithm due to the lack of transparency and reasoning mechanisms. Indeed, ML models like NNs are commonly viewed as "black box" models, mainly because their results are not always explainable by humans [135]. Thus, explainable AI (XAI) is expected to be one of the next steps in the realms of AI development, where human users can understand more about the results returned by ML models.

There are diverse criteria for declaring an AI solution to be explainable, including transparency, causality and trust. Herein, transparency implies the capability of explaining the ML models and their results to both technical and non-technical users. Increasing the transparency of an ML model will help even non-technical users to interpret how the model works and what to expect upon using it. By contrast, causality implies that causal inferences may be drawn from the behaviour of ML models. Finally, trust quantifies to what degree human users can believe in ML models. In general, explainable AI aims for improving the explainability in order to provide users with an improved understanding and to lend the users confidence when applying ML models [135], [136].

IV. PHY Security Optimization by Neural Networks

This section emphasizes the potential benefits of NNs in optimizing PHY security. More specifically, we bridge the gap between

- *complex*-variable optimization problems to be faced in PHY security (see Sub-section IV-A)
- and *real*-variable optimization problems to be solved by *real*-valued NNs (see Sub-section IV-B).

Then we discuss the ML-oriented research issues of *complex*-variable based NNs, thereby showing the potential benefits of *complex*-valued NNs in handling PHY security optimization (see Sub-section V-C). Finally, multiple-objective optimization is also discussed as a promising future research issue (see Sub-section V-C).

A. Optimization in PHY Security Design

Let us consider a simple security system in which a base station (A) broadcasts its signals to a legitimate user (B), which is overheard by an adversary (E). Let us denote the A-B channel and the A-E channel by $\mathbf{h}_B$ and $\mathbf{h}_E$, respectively. Upon denoting the *instantaneous* capacity of the A-B channel and that of the A-E channel by C_B and C_E, respectively, the *instantaneous* secrecy rate can be expressed as

$$C_s = \max(0, C_{\text{B}} - C_{\text{E}}) = \begin{cases} 0, & \text{if } C_{\text{B}} \leq C_{\text{E}} \\ C_{\text{B}} - C_{\text{E}}, & \text{if } C_{\text{B}} > C_{\text{E}}. \end{cases} \quad (24)$$

Secure system designs typically deal with the quantity $\Delta = C_{\text{B}} - C_{\text{E}}$, which is the difference between the capacity of Bob's

TABLE III
TYPES OF OPTIMIZATION PROBLEMS IN PHY SECURITY

MAXIMIZATION		
(P1)	$\max_{\mathbf{s}}$	$\Delta(\mathbf{s}, \mathbf{h}_B, \mathbf{h}_E)$
	s.t.	$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}$
(P2)	$\max_{\mathbf{s}}$	$C_{\mathrm{B}}(\mathbf{s}, \mathbf{h}_B)$
	s.t.	$C_{\mathrm{E}}(\mathbf{s}, \mathbf{h}_E) \leq$ a threshold
		$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}$
MINIMIZATION		
(P3)	$\min_{\mathbf{s}}$	$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\}$
	s.t.	$\Delta(\mathbf{s}, \mathbf{h}_B, \mathbf{h}_E) \geq$ a threshold
(P4)	$\min_{\mathbf{s}}$	$C_{\mathrm{E}}(\mathbf{s}, \mathbf{h}_E)$
	s.t.	$C_{\mathrm{B}}(\mathbf{s}, \mathbf{h}_B) \geq$ a threshold
		$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}$

TABLE IV
TYPES OF STOCHASTIC OPTIMIZATION PROBLEMS IN PHY SECURITY

MAXIMIZATION		
(Q1)	$\max_{\mathbf{s}}$	$\Delta_{avr}(\mathbf{s})$
	s.t.	$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}$
(Q2)	$\max_{\mathbf{s}}$	$C_{B,avr}(\mathbf{s})$
	s.t.	$C_{E,avr}(\mathbf{s}) \leq$ a threshold
		$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}$
MINIMIZATION		
(Q3)	$\min_{\mathbf{s}}$	$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\}$
	s.t.	$\Delta_{avr}(\mathbf{s}) \geq$ a threshold
(Q4)	$\min_{\mathbf{s}}$	$C_{E,avr}(\mathbf{s})$
	s.t.	$C_{B,avr}(\mathbf{s}) \geq$ a threshold
		$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}$

channel and that of Eve's channel. Based on the definitions of Δ, C_{B} and C_{E}, we will formulate optimization problems with the associated power constraints in mind. Upon denoting the transmit vector of Alice by $\mathbf{s}$ and bearing in mind the transmit power limitation, each and every signal should satisfy the following constraint:

$$\mathbb{E}\left\{\|\mathbf{s}\|^2\right\} \leq P_{max}, \tag{25}$$

where P_{max} is the power budget of Alice.

It is plausible that C_B is a function of $\mathbf{s}$ and $\mathbf{h}_B$. At the same time, C_E is a function of $\mathbf{s}$ and $\mathbf{h}_E$. Consequently, Δ is a function of $\mathbf{s}$, $\mathbf{h}_B$ and $\mathbf{h}_E$. As such, we can either write C_B, C_E and Δ for short in the way we have presented them above, or write $C_B(\mathbf{s}, \mathbf{h}_B)$, $C_E(\mathbf{s}, \mathbf{h}_E)$ and $\Delta(\mathbf{s}, \mathbf{h}_B, \mathbf{h}_E)$ to indicate their dependence on both the transmit signal and the channels. Since we are unable to control the channels, the overall goal of optimization becomes that of appropriately designing the transmit signal $\mathbf{s}$. Table III illustrates four different optimization problems that are widely considered in the literature. The problems in Table III are not necessarily convex. It should also be noted that (P2) and (P4) may be viewed as *security* vs. *reliability* trade-off problems, because C_B and C_E constitute a pair of conflicting functions. Roughly speaking, C_B reflects the reliability of transmission, and C_E characterizes the ability of an adversary to decode the source signal. While increasing the transmit power improves the reliability by reducing the bit error rate, the adversary also gets a better chance of detection, which degrades the security level [137], [138].

Due to the random nature of channels, the *instantaneous* quantities $C_B(\mathbf{s}, \mathbf{h}_B)$, $C_E(\mathbf{s}, \mathbf{h}_E)$ and $\Delta(\mathbf{s}, \mathbf{h}_B, \mathbf{h}_E)$ will always fluctuate with $\mathbf{h}_B$ and $\mathbf{h}_E$. As such, it is practically more promising to deal with their expectations. Let $C_{B,avr}(\mathbf{s})$, $C_{E,avr}(\mathbf{s})$ and $\Delta_{avr}(\mathbf{s})$ represent the expected value functions of $C_B(\mathbf{s}, \mathbf{h}_B)$, $C_E(\mathbf{s}, \mathbf{h}_E)$ and $\Delta(\mathbf{s}, \mathbf{h}_B, \mathbf{h}_E)$ over $\mathbf{h}_B$ and $\mathbf{h}_E$, we have

$$C_{B,avr}(\mathbf{s}) = \mathbb{E}_{\mathbf{h}_B}\left\{C_B(\mathbf{s}, \mathbf{h}_B)\right\}, \tag{26}$$

$$C_{E,avr}(\mathbf{s}) = \mathbb{E}_{\mathbf{h}_E}\left\{C_E(\mathbf{s}, \mathbf{h}_E)\right\}, \tag{27}$$

$$\Delta_{avr}(\mathbf{s}) = \mathbb{E}_{\mathbf{h}_B, \mathbf{h}_E}\left\{\Delta(\mathbf{s}, \mathbf{h}_B, \mathbf{h}_E)\right\}, \tag{28}$$

where $\mathbb{E}_{\mathbf{z}}\left\{func(\mathbf{z})\right\}$ denotes the expectation operator that calculates the expected value of a certain function $func(\mathbf{z})$ over a certain random vector $\mathbf{z}$. Note that $C_{B,avr}(\mathbf{s})$, $C_{E,avr}(\mathbf{s})$ and $\Delta_{avr}(\mathbf{s})$ are still functions of $\mathbf{s}$.

If the distributions of $\mathbf{h}_B$ and $\mathbf{h}_E$ are known, then the expected values $C_{B,avr}(\mathbf{s})$, $C_{E,avr}(\mathbf{s})$ and $\Delta_{avr}(\mathbf{s})$ may be derived by integrating - i.e. averaging - the instantaneous quantities $C_B(\mathbf{s}, \mathbf{h}_B)$, $C_E(\mathbf{s}, \mathbf{h}_E)$ over the domains of $\mathbf{h}_B$ and $\mathbf{h}_E$. However, in practice, the three distributions of the channels $\mathbf{h}_B$ and $\mathbf{h}_E$ may not be known. Instead of using integration, the expected values may be estimated based on measurement. Let $\mathbf{h}_B^{[t]}$ and $\mathbf{h}_E^{[t]}$ be the measured value of $\mathbf{h}_B$ and that of $\mathbf{h}_E$ at the t-th time slot. Note that $\mathbf{h}_B^{[t]}$ and $\mathbf{h}_E^{[t]}$ are empirical observations, and hence they are given. After T time slots, $\Delta_{avr}(\mathbf{s})$, $C_{B,avr}(\mathbf{s})$ and $C_{E,avr}(\mathbf{s})$ can be approximated by the *sample means* (or *sample averages*) as follows:

$$C_{B,avr}(\mathbf{s}) = \frac{1}{T}\sum_{t=1}^{T} C_B(\mathbf{s}, \mathbf{h}_B^{[t]}), \tag{29}$$

$$C_{E,avr}(\mathbf{s}) = \frac{1}{T}\sum_{t=1}^{T} C_E(\mathbf{s}, \mathbf{h}_E^{[t]}), \tag{30}$$

$$\Delta_{avr}(\mathbf{s}) = \frac{1}{T}\sum_{t=1}^{T} \Delta(\mathbf{s}, \mathbf{h}_B^{[t]}, \mathbf{h}_E^{[t]}). \tag{31}$$

From a practical point of view, the objective functions, as well as constraints, in the optimization problems (P1)-(P4) of Table III should be modified. For example, they can be reformulated into new optimization problems (Q1)-(Q4), which are shown in Table IV.

In general, the problems (Q1)-(Q4) of Table IV are non-convex and computationally challenging to solve. The formulation of (Q1)-(Q4) can be generalized and expressed in the

TABLE V
IN-DEPTH RESEARCH ON USING NEURAL NETWORKS FOR SOLVING OPTIMIZATION PROBLEMS

Stochastic Optimization	Constrained Optimization	Gradient Descent (GD) Methods	Subject Emphasized	Papers
	✓	Projection	A review of PNNs and their applications	[139]
	✓	Projection	Analyzing the stability and convergence of PNNs	[140]
	✓	Projection	Solving convex programming by PNNs and examples	[141]
	✓	Projection	RNN based on Projection Method	[142]
	✓	Projection	Proposing PNNs with a single-layer structure for real-time solvers	[143]
	✓	Projection	Solving quadratic min-max optimization problems by PNNs	[144]
	✓	*Not to be mentioned*	Modified RNN based on Karush–Kuhn–Tucker Conditions	[145]
	✓	Projection	Complex-valued Projection neural network	[146]
✓		SGD	Showing the computational efficiency in large-scale ML problems	[147]
✓		ADAM	Proposing ADAM and showing its computational efficiency	[148]
✓		YOGI	Proposing YOGI and proving it superior to ADAM	[149]
✓		Online, Batch	Presenting online learning algorithms and analyzing their convergence	[150]
✓	✓	AdaGrad	Proposing a family of update methods that use the geometry of data	[151]
✓	✓	SGD + Projection	Proposing an algorithm for stochastic multiple objective optimization	[152]
✓	✓	SGD + Projection	The effect of stochastic errors on distributed subgradient algorithms	[153]
✓	✓	SGD + Projection	Analyzing the optimality of SGD and improving its convergence rate	[154]
✓	✓	SGD + Projection	Proposing a variant of mini-batch SGD to enhance the convergence rate	[155]

form of a minimization problem as follows:[4]

$$\min_{\mathbf{z}} \quad \mathcal{F}(\mathbf{z}) = \frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{z}) \tag{32a}$$

$$\text{s.t.} \quad \mathbf{z} \in \mathcal{S}, \tag{32b}$$

where $\mathbf{z}$ is a *complex-valued* random vector, $\mathcal{S}$ is some constraint domain in the *complex* field, $\mathcal{F}(\mathbf{z})$ is the overall objective function, $f_t(\mathbf{z})$ is a certain function of $\mathbf{z}$, and T is still the number of training examples. Note that the non-convex problem (32) is routinely encountered in many technological fields.

B. NN-aided Optimization Solutions

To handle non-convex problems, a wide range of methods has been proposed. If a non-convex problem can be innerly approximated by a convex problem, the solution of the latter is also a feasible point/sub-optimal solution of the former. However, if a non-convex problem is relaxed into a convex problem, the solution of the latter may not even be a feasible point for the former. In fact, there is no single universal method for efficiently solving all types of non-convex problems. This statement is true, when using NNs for solving any optimization problem. Explicitly, specific network designs may be used for specific optimization problems. Table V lists several papers, which use NNs for solving optimization problems.

[4]Since maximizing $f(\mathbf{x})$ is equivalent to minimizing $(-1)f(\mathbf{x})$, the maximization problems (Q1)-(Q2) can also be transformed into minimization problems.

In order to be able to harness NNs to solve our security problems, we may desire to transform (32) into the following equivalent (or approximate) optimization problem:

$$\min_{\mathbf{w}} \quad \mathcal{L}(\mathbf{w}) = \frac{1}{T}\sum_{t=1}^{T} \ell_t(\mathbf{w}) \tag{33a}$$

$$\text{s.t.} \quad \mathbf{w} \in \mathcal{W}, \tag{33b}$$

where $\mathbf{w}$ is a *real-valued* vector containing the *weights and biases* of an NN, $\mathcal{W}$ is a closed convex set in the *real* field, $\mathcal{L}(\mathbf{w})$ is the overall loss function, $\ell_t(\mathbf{w})$ is the loss function corresponding the t-th example, and T is the training data size, i.e., the number of examples in the training data. The desire for $\mathbf{w}$ to be a real-valued vector arises from the fact that most of the existing NNs have been developed for solving optimization problems associated with real variables. Hence, there is a paucity of studies using NNs for solving optimization problems having complex variables, except for the work of Zhang *et al.* [146].

As such, instead of directly dealing with an optimization problem w.r.t. the *complex-valued* vector $\mathbf{z}$, we might prefer using an NN to solve another optimization problem w.r.t. the *real-valued* parameter vector $\mathbf{w}$. Once the optimal (or near-optimal) solution of $\mathbf{w}$ has been found, $\mathbf{z}$ in the original problem will also be updated accordingly. However, there may or may not be a functional relationship between $\mathbf{w}$ and $\mathbf{z}$.

It is also a matter of *utmost importance* that network designers establish the relationship between the complex-valued vector $\mathbf{z}$ and the real-valued vector $\mathbf{w}$, because this relationship will also show the relationship between the constraint $\mathbf{z} \in \mathcal{S}$ in (32b) and the constraint $\mathbf{w} \in \mathcal{W}$ in (33b). Similarly, it is challenging, but important to establish the relationship

between the objective function $\mathcal{F}(\mathbf{z})$ in (32a) and the loss function $\mathcal{L}(\mathbf{w})$ in (33a). For example, $\mathbf{z}$ can be chosen to be a real-valued output vector that is dependent on $\mathbf{w}$ [30] or independent of $\mathbf{w}$ [156].

Last but not least, (33) represents a type of stochastic optimization that has been studied both in machine learning research as well as in applied mathematics for at least a decade [148], [155], [157]–[159]. As seen in Table V, stochastic optimization problems may be solved with the help of NNs and gradient descent methods (see [147]–[150], [152]–[155] for more details). Given the importance of gradient (descent) methods, the following discussions will unveil the the role of them in dealing with both: i) stochastic *unconstrained* optimization and ii) stochastic *constrained* optimization problems.

*1) **Stochastic Optimization without Constraints**:* For ease of exposition, in this sub-section, we temporarily remove the constraint $\mathbf{z} \in \mathcal{S}$ from the stochastic optimization problem (33). With this in mind, we will discuss how NNs can deal with stochastic optimization problems of the form

$$\min_{\mathbf{w}} \quad \mathcal{L}(\mathbf{w}) = \frac{1}{T}\sum_{t=1}^{T} \ell_t(\mathbf{w}). \tag{34a}$$

According to [150], [152], [160], we can find the optimal (or near-optimal) solution of the parameter vector $\mathbf{w}$ by relying on the classical gradient descent. In connection with each gradient descent method, the update of $\mathbf{w}$ at the $(i+1)$-st iteration, denoted by $\mathbf{w}_{i+1}$, follows a different rule, as shown below:

- Gradient descent (GD) [150], [160]:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \gamma_i \frac{1}{T}\sum_{t=1}^{T} \nabla_{\mathbf{w}} \ell_t(\mathbf{w}_i), \tag{35}$$

  where $\gamma_i > 0$ is the learning rate.
- Stochastic gradient descent (SGD) [149], [150], [152], [160]:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \gamma_i \nabla_{\mathbf{w}} \ell_i(\mathbf{w}_i). \tag{36}$$

- Mini-batch gradient descent [161]:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \gamma_i \frac{1}{\tau}\sum_{t=1}^{\tau} \nabla_{\mathbf{w}} \ell_t(\mathbf{w}_i), \tag{37}$$

  where $\tau \leq T$ is the number of training examples in a small subset of the full training dataset. In the case of the mini-batch GD, the training dataset of T samples is divided into many subsets, and each update corresponds to a subset. Setting $\tau = 1$ leads to the case of SGD, while setting $\tau = T$ leads to the gradient descent scenario.

Apart from the aforementioned gradient descent methods, there are also other gradient-based methods, such as the second-order gradient descent [147, eq. (3)], the second-order SGD [147, eq. (5)], ADAM [148], and YOGI [149], just to name a few.

*2) **Stochastic Optimization under Constraints**:* A specific type of feedback/recurrent neural networks (RRNs), termed as *projection neural networks*(PNNs), have been popularly used for solving constrained optimization over many years [139], [143], [162]. As for stochastic constrained optimization problems (e.g., (33)), PNNs can also be used for finding optimal (or near-optimal) solutions. In principle, the constraint $\mathbf{w} \in \mathcal{W}$ in (33) can be handled by the projection method of [152]–[155].

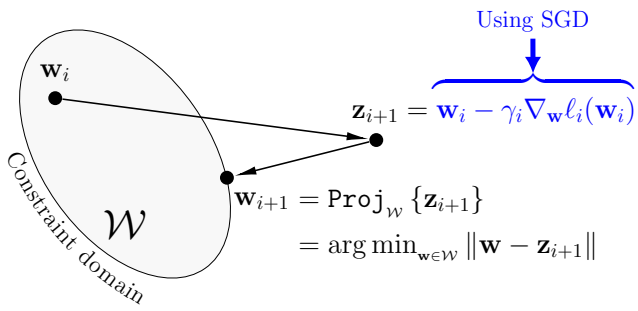


Fig. 15. An illustration of projection gradient descent. The projection-based update rule includes two steps as follows: Starting at a point $\mathbf{w}_i \in \mathcal{W}$, we use a gradient method (e.g., SGD) to find some intermediate point $\mathbf{z}_{i+1}$. Then, we project $\mathbf{z}_{i+1}$ onto the domain $\mathcal{W}$ to find the updated point $\mathbf{w}_{i+1}$ that satisfies the constraint $\mathbf{w}_{i+1} \in \mathcal{W}$.

Fig. 15 illustrates the update rule of a PNN. To be more particular, the update rule includes two steps:

- Let us assume that we commence from a point $\mathbf{w}_i$ at the i-th iteration. In the first step, a gradient descent method is used for finding an intermediate point, namely $\mathbf{z}_{i+1}$. Note that the point $\mathbf{z}_{i+1}$ does not necessarily fall within the domain $\mathcal{W}$ (i.e., it is possible to have $\mathbf{z}_{i+1} \notin \mathcal{W}$).
- In the second step, the projection method is used for projecting the intermediate point $\mathbf{z}_{i+1}$ onto the domain $\mathcal{W}$. Let $\mathbf{w}_{i+1}$ be the projected point. It is plausible that $\mathbf{w}_{i+1}$ is also the updated point at the $(i+1)$-st iteration, because the constraint $\mathbf{w}_{i+1} \in \mathcal{W}$ is satisfied.

This 2-step update process is also known as the projected gradient descent (PGD) technique [163], [164]. As an example, let us assume that in the first step, we use the SGD for updating the intermediate point, i.e,

$$\mathbf{z}_{i+1} = \mathbf{w}_i - \gamma_i \nabla_{\mathbf{w}} \ell_i(\mathbf{w}_i). \tag{38}$$

Then we use the projection method for updating the parameter vector as follows:

$$\begin{aligned} \mathbf{w}_{i+1} &= \mathtt{Proj}_{\mathcal{W}}\{\mathbf{z}_{i+1}\} \\ &= \arg\min_{\mathbf{w}\in\mathcal{W}} \|\mathbf{w} - \mathbf{z}_{i+1}\| \end{aligned} \tag{39}$$

where $\mathtt{Proj}_{\mathcal{W}}\{\cdot\}$ denotes the operator carrying out the projection onto $\mathcal{W}$.

V. The Application of ML in PHY Security

In this section, the application of ML in PHY security is presented. We divide the section into three sub-sections: i) ML-aided PHY authentication, ii) ML-aided secure PHY transmission, and iii) NN-aided security optimization. The first sub-section presents the employment of ML for creating classifiers to categorize, detect and authenticate devices. Meanwhile, the second sub-section presents the employment of ML for designing secure PHY transmission. By contrast, the last sub-section delves into how to deal with PHY security optimization problems using NNs.

TABLE VI
THE APPLICATION OF ML ALGORITHMS IN PHY AUTHENTICATION

Papers	Algorithms	At the heart of input data	Main contributions
[23]	Decision tree, k-NN, SVM	Channels	Extracting input features from channel measurements and comparing the authentication performance of decision tree, k-NN and SVM
[20]	SVM, LDA	TOA, RSS	Using TOA and RSS as input features of SVM and LDA classifiers and creating two authentication schemes based on SVM and LDA
[29]	k-means, k-NN	RSS	Respectively using k-means and k-NN for extracting the features of signals and for detecting spoofing attacks in wireless sensor networks
[73]	k-means, OC-SVM	Means & variance	Detecting PHY attacks by OC-SVM and k-means classifiers, where OC-SVM is trained on one-label data and k-means works directly on testing data
[25]	One-class nearest neighbour	Channels	Evaluating the detection performance of one-class nearest neighbour classifiers in realizing the presence of an active attacker
[27]	Logistic regression	RSSI	Using the RSSI of signals as the input data and building a logistic regression model for detecting spoofing attacks
[24]	GMM	Channels	A GMM-based authentication experiment in machine type communication
[26]	GMM	Channels	Extracting channel-gain-based features and detecting spoofing attacks by a GMM classifier, where GMM parameters are estimated by the expectation maximization algorithm
[28]	GMM	Channels	Using the Karhunen-Loeve transform for feature extraction and GMM for authentication
[22]	NN	Channels	Proposing an extreme-learning-based authentication model without requiring a detection threshold
[165]	NN	Channels	Exploiting data augmentation methods for accelerating the training speed and improving the accuracy of neural-network-based authentication models in IoT systems
[67]	NN	Fingerprints	Considering mobile hardware security and authenticating IoT nodes by using a neural-network-based fingerprinting method
[19]	RL	Channels	Proposing a hypothesis test for detecting spoofing attacks, where the detection threshold in the hypothesis test is determined by reinforcement learning
[166]	Kernel-based method	Channels	Proposing an adaptive authentication model based on the idea of the kernel machine, which is similar to classical SVM classifiers in terms of reducing the dimension of feature space

A. ML-aided PHY Authentication

1) Recent Applications: A range of ML-aided classification algorithms may also be applied for enhancing security at the physical layer. For example, Table VI lists the popular ML algorithms used and the related papers. In the following, we review these papers in more detail.

In [29], a beneficial combination of both k-NN and K-means techniques is suggested for detecting spoofing attacks in wireless sensor networks. K-means clustering is used for extracting features from RSS samples, while k-NN plays the role of a classifier [167]. The authors of [73] compare the performance of K-means and OC-SVM, and show that OC-SVM results in a more stable detection performance than K-means, when the power of a spoofing signal fluctuates. However, the performance of OC-SVM is worse than that of K-means, when the spoofing signal is transmitted at high power. The idea of using OC-SVM for training the data is also seen in [168], where Abdrabou *et al.* first build datasets based on the received signals and then evaluate the authentication performance of majority voting schemes. In [20], both SVM and LDA are harnessed for processing the RSS, TOA and another correlation-based feature.

As a further development, the authors of [23] suggest using estimated channel matrices for generating features and then compare four different ML algorithms, i.e., the k-NN, SVM, decision tree and bagged tree, in terms of their accuracy and prediction time. Similar to [73], the training data considered in [25] contains only a single class under the assumption that there is no knowledge concerning any of the potential adversary. Also assuming that the prior knowledge of the CSI of illegitimate devices is unavailable, Du *et al.* [169] utilize a CART decision tree for designing an authenticator that is trained on CSI-based datasets for differentiating legitimate devices from illegitimate ones. The authenticator in [169] was shown to reduce the dimension of the original datasets into lower-dimensional data for employment in industrial environments. In [25], single-class nearest neighbour classification is performed on the single-class data for finding both high- and low-density regions, thereby creating a predictive model for authentication. The authors of [27] propose a logistic regression model for authentication and estimate the coefficients of the model by using the popular Frank–Wolfe algorithm.

Furthermore, the Gaussian mixture model (GMM) is proposed in [24], [26], [28]. To elaborate, Weinand *et al.* [24] build a simple authentication framework for machine type communication by using the magnitude of channels as the input data. Qiu *et al.* [26] build the training data associated with two features that are based on the Euclidean distance and

Pearson correlation, respectively. Qiu *et al.* [28] exploit the Karhunen-Loeve transform for extracting the most significant features and for reducing the dimension of the data. The low-dimensional representation created by the Karhunen-Loeve transform in [28] allows ML algorithms to deal with high-dimensional datasets, while the lack of a dimensionality reduction technique in [24] requires a large amount of memory for data storage and incurs increased computation. The spoofing detection performance comparison between [28] and [24] once again confirms that bespoke feature selection is always crucial in the data preprocessing used for authentication.

The family of NNs is considered in [22], [67], [165]. Specifically, in [22], the data is formulated in a similar way to [26], apart from the difference that Wang *et al.* [22] use extreme machine learning based on NNs, while Qiu *et al.* [26] use a Gaussian mixture model. Liao *et al.* [165] focus their attention on data augmentation methods that harness an extra amount of data in addition to the existing data. The data augmentation is eminently suitable for improving the robustness of the data, especially when dealing with high-volume data. Chatterjee *et al.* [67] consider the identification of nodes in an Internet-of-Things system by developing a detection framework based on NNs and physical unclonable functions. On the other hand, Wang *et al.* [170] design a DNN relying on the softmax function at the output to distinguish between legitimate and illegitimate CSI samples. However, the detection mechanism of the authenticator in [170] is sensitive to the choice of a predetermined detection threshold. A special type of NN, namely VAE, is also employed for PHY authentication in [171], which does not require the CSI of attackers for training. However, the sophisticated authenticator of [171] is based on a hierarchical VAE architecture that consists of an AE module designed for feature extraction and a VAE module conceived for detecting spoofing attacks.

Regarding reinforcement learning, Xiao *et al.* [19] model the interaction between a legitimate user and an eavesdropper as a zero-sum authentication game in a dynamic environment, and employ reinforcement learning for finding the optimal threshold of hypothesis testing. Finally, Fang *et al.* [166] develop an adaptive learning model based on the kernel least mean square method in which the authentication problem is modelled as a linear system for reducing the dimension of the feature space and complexity. Zhang *et al.* [172] proposes a collaborative PHY authentication scheme, where edge devices cooperate to build an authenticator. In [172], RL is utilized for finding the most reliable devices that yield the highest authentication performance and for identifying less reliable devices that are potentially harmful.

2) Future Directions: In general, there is no consensus in the literature concerning the best criteria for creating the input data in previous works. For instance, the authors of [22] and [26] create the data based on the Euclidean distance and the Pearson correlation, respectively. By contrast, the data for the PHY authentication in [27] relies on the RSSI. The features of the data used for PHY authentication in [67] include the local oscillator frequency, Doppler shift, as well as the in-phase and quadrature components of transmitted signals. By contrast, the carrier frequency offset, channel impulse response, and RSSI are used as features for ML-aided PHY authentication in [166]. Given that the selection (or extraction) of features for the data directly affects on the detection performance of an ML-aided PHY authentication model as shown in [28] and [165], future works should pay more attention to the topic of feature selection.

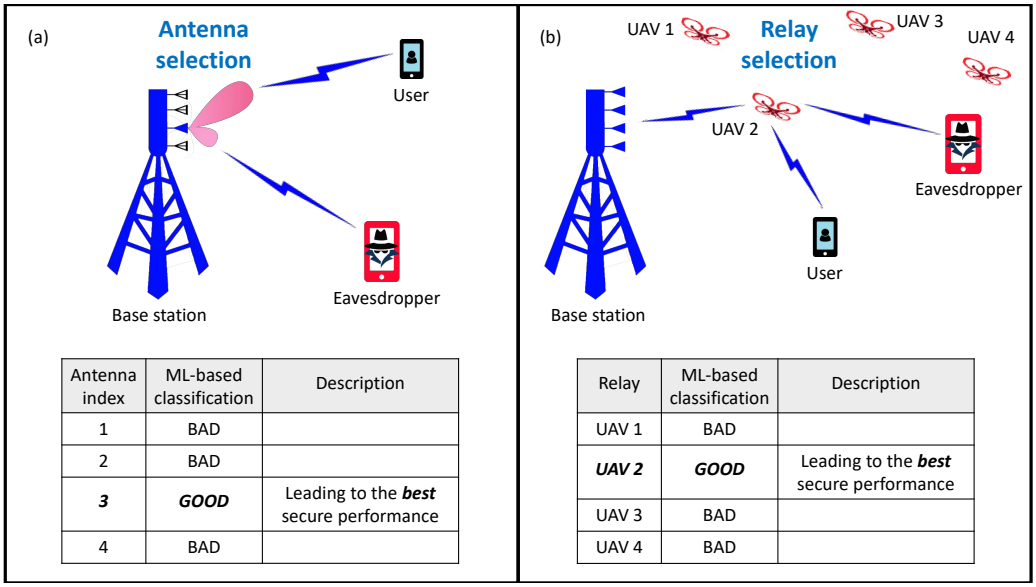
However, at the time of writing there are no authoritative comparison of the security performance of ML-aided classification algorithms. Thus future research has to carry out extensive studies to find suitable input data types for PHY authentication, and to find suitable ML classification algorithms for each type of data. On a similar note, anomaly detection methods require similar attention to that concerning PHY security. As part of the ML family, anomaly detection (also known as outlier detection) includes many methods that are developed for identifying rare events/observations and thus they are normally used for intrusion detection upper ISO layers [39]. However, anomaly detection methods remain underused in maintaining security at the physical layer. Additionally, given the ongoing development of NNs, it is shown that NNs can be readily exploited for discovering hidden-but-useful features from available data for authentication purposes [36], [43]. Finally, it is also worth considering the class of *cross-layer* authentication harnessing the data both at the physical and upper layers.

B. ML-aided Secure PHY Transmission

1) Recent Applications: When it comes to the integration of ML into ML-aided secure PHY transmission, a representative range of contributions are listed in Table VII, which are detailed below.

He *et al.* [30] exploit both beamforming and artificial noise to deal with an eavesdropper and formulate an optimization problem that maximizes the effective secrecy throughput by harnessing an NN. Xing *et al.* [31] also use NNs for secure transmission by relying on cooperative beamforming in a relay-aided system. Besser *et al.* [21] strike a trade-off between reliability and security by formulating a multiple-objective optimization problem. To resolve the associated trade-off, wiretap codes are designed based on NN-aided autoencoders. As another development, Li *et al.* [32] design a Q-learning-based power control strategy for secure transmission by considering a powerful attacker having a high number of antennas. Upon using reinforcement learning, Xiao *et al.* [33] propose an optimal beamforming scheme for visible light communication In [34], reinforcement learning is used by Miao and Wang to handle the frequency allocation problem without requiring any information exchange among base stations. In contrast to the above-mentioned papers, He *et al.* [35] do not use NNs or reinforcement learning in designing secure transmissions. Instead, an SVM and a naive Bayes algorithm are used for transmit antenna selection. By selecting the most suitable antenna for transmission, the security level is shown to be improved. By contrast, Wen *et al.* [173] assume that intelligent attackers can use supervised learning to decode even artificial-noise-contaminated signals.

More recently, Liu *et al.* [174] consider an RIS-based sensing and communication system and propose an optimal



Antenna index	ML-based classification	Description
1	BAD	
2	BAD	
3	***GOOD***	Leading to the ***best*** secure performance
4	BAD	

Relay	ML-based classification	Description
UAV 1	BAD	
UAV 2	***GOOD***	Leading to the ***best*** secure performance
UAV 3	BAD	
UAV 4	BAD	

Fig. 16. Two examples of using classification algorithms (i.e., ML classifiers) for selecting the best system components. In the sub-figure (a), the result of ML-based classification shows that the 3-rd antenna can lead to the best secure transmission. In the sub-figure (b), the result of ML-based classification shows that the 2-nd UAV can be the best relay for retransmitting signals to the intended user.

TABLE VII
THE APPLICATION OF ML ALGORITHMS IN SECURE PHY TRANSMISSION

Papers	Algorithms	Main contributions
[21]	NN	Employing a pair of neural networks for designing wiretap codes that can minimize both the block error rate and the amount of information leakage
[30]	NN	Considering an energy-harvesting system in the presence of an eavesdropper and using a neural network for learning system parameters that maximizes the security throughput
[31]	NN	Using a neural network for finding power allocation coefficients that enhance the secrecy rate of a relaying system, under the impractical assumption of having the eavesdropper's CSI
[32]	RL	Considering the Nash equilibrium in a zero-sum game between a transmitter and an attacker, and finding the Q-learning-based power control strategy for achieving the equilibrium
[33]	RL	Using reinforcement learning for finding the optimal beamforming policy that improves the secrecy rate of a visible light communication system
[34]	RL	Considering a two-layer ultra-dense network with the focus of securing the macro layer and proposing a reinforcement-learning-aided game-theoretic model for enhancing the security and delay performance
[35]	SVM, naive Bayes	Using SVM and naive Bayes for selecting the optimal transmit antenna that maximizes the secrecy performance
[173]	*not to be named*	Proposing learning-based attacks in favour of an attacker, where the self-defined supervised learning method utilizes the packet preamble and header as training data

secure PHY transmission scheme by jointly optimizing the transmit beamforming and RIS coefficients. To deal with the proposed security optimization problem, [174] employs an RL algorithm that is built on the basis of an actor NN and a critic NN. In [175], Lin *et al.* consider an energy-harvesting-assisted cognitive radio network in the presence of eavesdroppers and proposes a security solution based on optimizing the energy harvesting time and power allocation. The security optimization problems in [175] are addressed by an RL algorithm that contains a NN-based generator for producing the distribution of data and a NN-based discriminator for distinguishing real and fake output values.

2) Future Directions: In terms of secure PHY transmission, ML classification algorithms are capable of going beyond realizing eavesdroppers. More particularly, ML classifiers can be used for any classification tasks rather than being limited to eavesdropping detection. For example, based on the system component classification, we can decide to use the most suitable components for attaining an increased security level. To argument this further, let us consider the work of He *et al.* [35]. The idea of [35] is to use SVM and naive Bayes algorithms to classify the transmit antennas of a transmitter in order to find the best antenna for secure transmission. This idea may also be readily generalized to other system component selection, because there are many ways of enhancing the security performance through the selection of the most appropriate system components, such as antenna selection [176], transmitter selection [177], relay selection [137], [178], and so on [66]. Note that the authors of [176]–[178] do not consider the employment of ML algorithms for component

selection. To visualize the generalized idea of using ML for selecting system components, let us consider Fig. 16, where a pair of secure PHY transmission designs are considered: (a) using ML classifiers for selecting the best transmit antenna, and (b) using ML classifiers for selecting the best relay. In short, the use of ML classifiers can point out which system components are the best for secure and reliable transmission. However, at the time of writing, there is paucity of literature in this research direction.

C. NN-aided Security Optimization

1) Security Optimization by NNs: The variables encountered in optimization problems may be real-valued or complex-valued. Hence, we will classify the optimization problems considered into these two types:

- *Real*-variable optimization (real-OPT) problems, whose variables belong to the real field;
- *Complex*-variable optimization (complex-OPT) problems, whose variables belong to the complex field.

Indeed, the employment of NNs for solving optimization problems has been an active field for years. However, most of the optimization problems solved have been *real*-valued problems [179], [180]. Only a few authors consider the use of NNs for solving *complex*-valued optimization problems [146], [181].

Similarly to the optimization problems, we can also divide the family of NNs into a pair of categories:

- *Real*-valued neural networks (namely, real-NNs);
- *Complex*-valued neural networks (namely, complex-NNs).

Naturally, the first category relates to all NNs designed for solving real-OPT problems. By contrast, the second category encompasses the NNs designed for solving complex-OPT problems. As a matter of fact, in many fields of application, real-OPT problems are more likely to be encountered, but in the field of communications, typically the opposite is true. Hence, there is a pressing need for further research on solving the unsolved open optimization problems (including real-OPT and complex-OPT problems) of the communications comminity. To elaborate a little further, some authors have applied real-NNs for solving real-OPT problems to improve the performance of communication systems [30], [182], but there is a paucity of literature on the employment of complex-NNs for solving complex-OPT problems in communication systems.

- Using real-NNs for solving complex-OPT problems requires further research for transforming complex-OPT problems into real-valued ones, so that real-NN may be harnessed for solving them. The upper part of Fig. 17 illustrates this issue.
- However, it is more natural to use complex-NNs for directly solving complex-OPT problems [146]. Having said that, substantial future research is required on complex-NNs [146], [181] before they can be harnessed for solving complex-OPT problems routinely encountered by the communications community. The lower part of Fig. 17 illustrates this issue.

Since communication system designs include PHY security, it is anticipated that real-NNs, as well as complex-NNs will help enhance their security in the face of uncertainty, when the ability to learn from and adapt to the environment is of crucial importance.

2) Security Optimization for satisfying Multiple Objectives: When it comes to the formulation of optimization for a system, it may be desirable to simultaneously optimize multiple objectives even when they are conflicting. Multiple-objective optimization can be formulated as follows:

$$\min_{\mathbf{z}} \quad \left[f_1^{obj}(\mathbf{z}), f_2^{obj}(\mathbf{z}), \ldots, f_{M_{obj}}^{obj}(\mathbf{z})\right] \tag{40a}$$

$$\text{s.t.} \quad \mathbf{z} \in \mathcal{S}, \tag{40b}$$

where $f_m^{obj}(\cdot)$ (with $m \in \{1, 2, \ldots, M_{obj}\}$) is an objective function in the set of all objectives, while $\mathcal{S}$ is some constraint domain in the complex field. To elaborate, multiple-objective optimization (MOO) is different from the single-objective optimization problems presented in Tables III–IV of Section IV. Explicitly, the single-objective optimization problems of Section IV simplify the complex multi-objective real-life problems by simply using the conflicting objectives as constraints. By contrast, the goal of multiple-objective optimization is to *jointly* optimize several objectives at the same time, where each objective $f_m^{obj}(\cdot)$, $m \in \{1, 2, \ldots, M_{obj}\}$ represents a different performance metric. Naturally, the solution-space or search-space of this problem continues to grow upon including more metrics, which eventually renders the problem intractable.

Hence, multiple-objective optimization problems are challenging to deal with and the optimal solution may not even exist. Having said that, it is possible to circumvent the challenges of multiple-objective optimization using the Pareto optimality concept [183] relying on a set of solutions which constitute the *Pareto front* of all optimal solutions. For example, we can improve a performance metric (e.g., the capacity of a legitimate channel), but another performance metric will be degraded (e.g., the capacity of a wiretap channel will increase). Broadly speaking, the Pareto front is the collection of all optimal solutions, where none of the metrics may be improved without degrading at least one of the others.

To find the Pareto front, typically *bio-inspired metaheuristic* algorithms are employed [183]–[185]. Acording to [183], common bio-inspired metaheuristic algorithms include the following main categories: *evolutionary algorithms*, *swarm intelligence algorithms*, *NNs*, *reinforcement learning*, *fuzzy logic*, just to name a few. For example, the combination of NNs and the Pareto approach is studied in [186], while the combination of reinforcement learning and the Pareto approach is presented in [187]. Since machine learning itself may be viewed as a multiple-objective optimization tool, its pairing with the Pareto approach is a natural marriage [188].

Multiple-objective optimization is eminently suitable for network design [189], but it is rarely used in PHY security designs. Naturally, in PHY security, at least one of the objectives in the optimization problem (40) must be related to a security metric [190]. Indeed, wireless networks pose many challenges related to the maximization of the data rate, the minimization

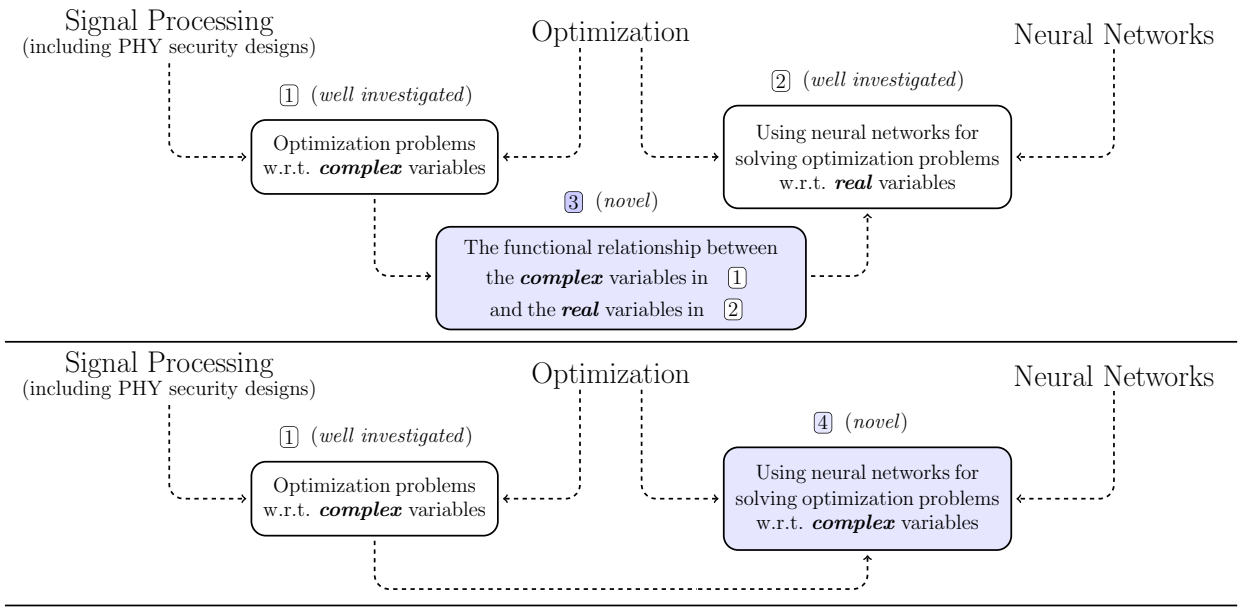


Fig. 17. In communications, we are most likely to deal with complex variables rather than real variables in optimization problems. Thus, future works are envisaged to happen with 2 possibilities: (i) find a way to transform the original optimization problems w.t.r complex variables into new ones that conventional NNs can solve, and (ii) use new types of NNs that can deal with complex numbers in order to solve the original optimization problems directly.

of consumed power, the maximization of security level, the minimization of latency and so on. These goals are hard to achieve at the same time, because they are often conflicting. For example, the bit error rate of a wireless system may be reduced by increasing the transmit power, but this increases the eavesdropping probability, which results in a security vs. information trade-off [137]. The trade-off between security and delay was considered in [191].

D. Insights and Further Discussions

PHY authentication problems can be handled by a wide range of ML models. For supervised ML classification models, the capability to distinguish between normal and abnormal data samples lays the foundation for building authenticators that can detect it, whether a new sample is associated with an attack or not. However, the PHY authentication based on supervised ML requires abnormal data samples to be available, which implies that there is prior knowledge of previous attacks. By contrast, in unsupervised learning, especially in anomaly detection models, the learning capability is often applied to normal data samples, thus making the models familiar with both normal occurrences and anomalies that statistically deviate from the norm. This ability to identify anomalies is the basic for building authenticators that detect attacks without requiring any information about attackers. Noticeably, to speed up the processing of PHY authenticators by reducing the computational burden, authors have resorted to some dimensionality reduction techniques.

In contrast to PHY authentication problems, secure PHY transmission does not focus on authenticating devices or detecting malicious attacks, it rather aims for the transmission among legitimate network entities securely. The security of PHY transmission can also be improved by applying ML models. The ML-aided secure PHY transmission solutions often rely on transmit beamforming designs, power allocation and other network parameters, depending on what specific network types are considered. NNs are eminently suitable for solving stochastic optimization problems (including security optimization). However, this area of research is still largely open, hence requiring future research.

VI. ML-aided PHY Security for Future Communication Systems

When new wireless networks are rolled out, their security level has to be improved. Given that ML techniques are gradually finding their way into the new generation of wireless networks, the investigation of ML-aided PHY security in those networks continues to be important part of future research. Below we critically appraise a number of emerging solutions in wireless networks:

- **Reconfigurable intelligent surfaces (RISs)**: It is expected that RISs will be deployed at locations near the base stations for mitigating the LoS blockages [192]. They are capable of creating additional reflected propagation paths and beneficially controlling the phase shifts of reflected signals, thus making robust connections between a transmitter and a receiver. In terms of security, it is still an open question how to orchestrate the coordination of RISs and ML techniques for improving the security level of wireless systems, although some insights have been provided in [193]–[195]. However, there is a pressing need to perform more extensive investigations on the security performance of ML-aided RISs.
- **Millimeter wave (mmWave)**: While most existing communication systems use carrier frequencies below 6 GHz, mmWave systems employ a wide range of frequency

spectrum spanning from 30 GHz to 300 GHz [196], [197]. Their PHY security has been characterized in [198], [199], which ML has been harnessed in [200], [201]. However, there is a paucity of literature on ML-aided PHY security in mmWave systems, apart from [202], [203]. Briefly, logistic regression is proposed in [202] to detect eavesdropping attacks in the uplink, while reinforcement learning is used in [203] to deal with jamming.

- **Visible light communication (VLC)**: VLC technology modulates visible light emanating from light-emitting diodes (LEDs) [204] used for illumination at low cost. Since the light from LEDs does not propagate through walls, VLC is robust to interference. Thus, VLC is a potential candidate for future indoor systems. However, due to the broadcast nature of VLC in the downlink, it is vulnerable to adversaries. Hence, PHY security has also been investigated in the context of VLC [205]–[207]. Having said that, a full investigation of ML-aided PHY security in VLC is still largely open.
- **Light fidelity (LiFi)**: While VLC only uses the visible light spectrum, LiFi additionally exploits the infrared and the ultraviolet bands [208], [209]. However, the ML-aided PHY security of LiFi is an open area.
- **Cell-free massive multiple-input multiple-output (MIMO)**: As a variant of distributed massive MIMO systems, cell-free massive MIMO has numerous benefits, especially in terms of throughput fairness [210], despite its low complexity. Its PHY security is investigated in [211]. As a further advance, *federated learning* is proposed for cell-free massive MIMO in [212], but the investigation of ML-aided PHY security is still in its infancy.
- **Body area networks**: A wireless body area network is typically comprised of medical sensors that are placed on the body to measure physiological signals. A practical wireless body area network is expected to collect data from a large pool of patients and use ML algorithms for analyzing the health and needs of patients. Indeed, personalized healthcare through mobile wearable devices is expected to revolutionize the future of healthcare. Furthermore, the authors of [213] rely on the RSSI as the feature of data and compare different ML algorithms (i.e., decision tree, SVM, k-NN and neural network) in the context of gait authentication. However, the large-scale deployment of wireless body area networks must meet stringent security policies and requirements. Once a wireless body area network is entitled to use confidential data for health surveillance relying on statutory medical services, PHY security becomes a pivotal issue [214], [215]. Thus, the design of body area networks has to ensure that the user database is not leaked to illegitimate users or organizations. Hence, ML-aided PHY security constitutes an exciting domain of research.
- **Space-air-ground communication**: This is a topical research area [216], [217], but there is limited literature on simultaneously considering all three network segments [218], [219].
- **Quantum communications**: Quantum-aided communication systems have rapidly evolved in recent years [220]–[223]. In terms of security, quantum cryptography relies on a range of secure protocols such as, quantum key distribution [221], quantum secret sharing, quantum secure direct communication [224], and controlled bidirectional quantum secure direct communication [222]. These quantum cryptography protocols have also found practical applications [224], [225]. In terms of ML, a number of quantum-based learning algorithms have been developed to deal with critical problems in learning from data [226], [227]. The family of popular quantum machine learning algorithms includes quantum k-NN [228], quantum SVMs [229], quantum NNs [230], quantum decision tree [231], just to name a few. Although quantum machine learning is still in the early stage of development, quantum machine learning algorithms are capable of speeding up computational processes, especially in learning from quantum-domain data [232]–[234].

Other future systems include machine-type communications [235], free space optical communication [236], and non-orthogonal multiple access systems [237], but ML-aided PHY security still remains a largely open domain of research. This is because the benefit of ML in PHY security has not been fully documented and the emergence of new communication systems will continue to widen the avenue for the investigation of ML-aided PHY security.

VII. Summary and Design Guidelines

A. Summary

In this work, we have summarized a variety of ML algorithms that can be employed in the context of PHY security. These range from typical supervised learning algorithms (i.e., k-NN, SVM, Random Forest, LDA) to typical unsupervised learning algorithms (i.e., k-means, OC-SVM, iForest, Hierarchical clustering). Neural networks, which can be classified as either supervised or unsupervised learning, has been also summarized. In parallel, we have discussed the state of the art in ML-aided PHY security by separately considering two aspects: ML-aided PHY authenticating and ML-aided PHY security design. Throughout the paper, we have shown that an ML classification algorithm can play two roles: i) it can classify the data for the authentication purpose, and ii) it can also take part in the process of selecting system components for secure transmission design. When it comes to security optimization, we have paid our special attention to the potential use of NNs for solving optimization problems that are likely to be faced in designing PHY security strategies. Accordingly, we have shown and bridged the gap between complex-variable optimization in PHY security design and real-valued/complex-valued NNs. Finally, we have presented the role of ML-aided PHY security in future communication systems.

B. Design Guidelines

Regardless, whether we embark on the design of an authentication solution or a secure transmission strategy, the input data should be considered as the first step. More explicitly,

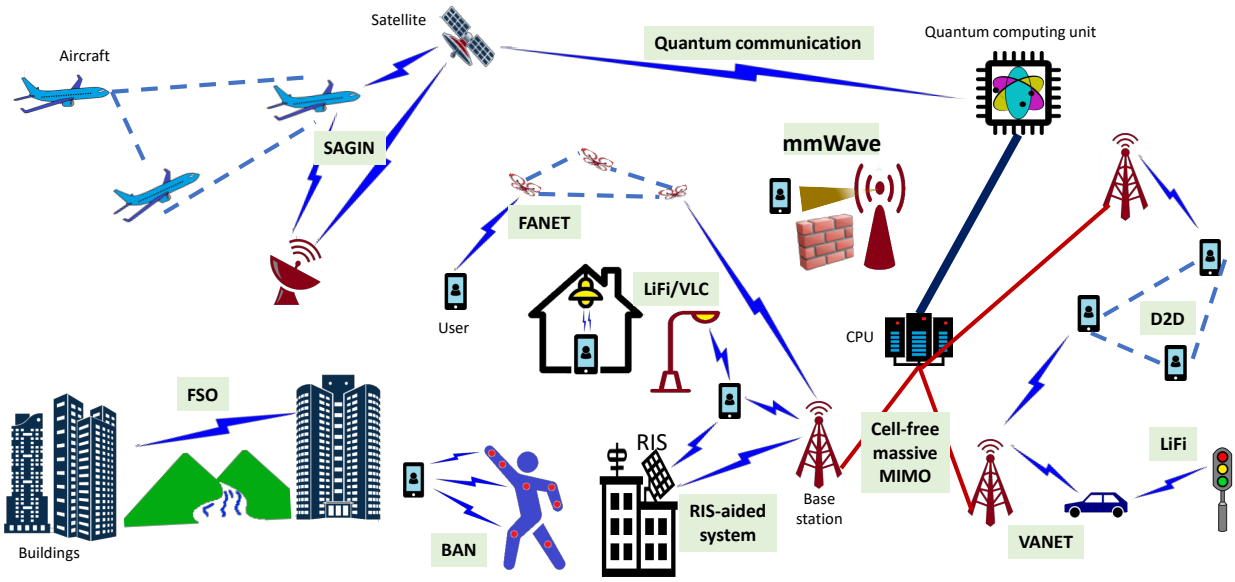


Fig. 18. A visualization of future networks.

the wireless data should be transformed into the input data for ML to interpret them. For example, an ML algorithm may be unable to directly use the modulated signals, because they are complex-valued; instead, it is necessary to decompose the complex-valued modulated signals into their amplitude and phase based representation in order for the ML algorithm to interpret and process them.

Given the input data, we will now briefly touch upon three representative authentication/security designs:

- **Authentication:** ML-based classification algorithms constitute prominent candidates for detecting security attacks, as well as for employment in authenticating devices. However, some ML algorithms work well on single-label data, while others only work on double-label data. Thus, it is necessary to determine the specific type of ML algorithm based on the nature of the available training datasets. A training dataset might contain attack-related samples if an attacker launched jamming/spoofing attacks in the past. Those attack-related samples may be labelled as "attack" if we have some information about the attacker, such as the CSI. On the other hand, without the CSI of the attacker, the attack-related samples in the dataset of interest might be treated as normal samples, which result in the wrong training dataset. Thus, based on the knowledge of the attacker, choosing carefully selected suitable data samples will allow us to choose a suitable ML algorithm for training. In this sense, given the attacker's CSI, binary/multi-label ML classification techniques may be used. On the other hand, without the attacker's CSI knowledge, both single-label ML classification techniques as well as anomaly detection techniques are more suitable candidates.
- **Component Selection:** The family of ML-aided classification algorithms is also eminently suitable for optimizing antenna selection, relay selection and RIS selection. Let us consider an RIS-aided system as our example. When RISs are deployed in the field, they create additional propagation paths spanning from the source to the desired user for mitigating the effects of lin-of-sight blocking. However, in the presence of an adversary, there will also be additional paths from the source to the adversary, thus making the system vulnerable to malicious eavesdropping. Increasing the transmit power improves the transmission integrity between the source and the desired user, but also increases the eavesdropping probability. It is desirable to design the reflection coefficients of the RIS for ensuring that the benefits obtained by the legitimate user are higher than those obtained by the adversary. By formulating an objective function based on the secrecy rate, bespoke ML classification algorithms can be trained to find the best set of RIS coefficients for maximizing the security rate.
- **Security Optimization:** While a real-valued neural network is routinely trained for minimizing its loss function $\mathcal{L}$, in wireless security design problems we often have to maximize the objective function, say Δ. By assigning $\mathcal{L} = -\Delta$, a real-NN can be employed for dealing with security optimization. However, it is of vital importance to define the relationship between the outputs of the real-NN and the variables of the proposed security optimization problem. For example, if $z = z_1 + j \times z_2$ is a complex-valued variable to be optimized, then z_1 and z_2 may be represented by two outputs of the NN, since a real-NN can only process real-valued variables. Following the training process, the loss function $\mathcal{L}$ is minimized (or the objective function Δ is maximized), and the outputs z_1 and z_2 of the real-NN can be arranged into $z = z_1 + j \times z_2$ to find a near-optimal solution of the security optimization problem considered.

REFERENCES

[1] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," *Proc. of the IEEE*, vol. 104, no. 9, pp. 1727–1765, 2016.

[2] W. Trappe, "The challenges facing physical layer security," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 16–20, 2015.

[3] A. Mukherjee, S. A. A. Fakoorian, J. Huang, and A. L. Swindlehurst, "Principles of physical layer security in multiuser wireless networks: A survey," *IEEE Commun. Surv. & Tut.*, vol. 16, no. 3, pp. 1550–1573, 2014.

[4] C. E. Shannon, "Communication theory of secrecy systems," *The Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949.

[5] A. D. Wyner, "The wire-tap channel," *Bell system technical journal*, vol. 54, no. 8, pp. 1355–1387, 1975.

[6] J. M. Hamamreh, H. M. Furqan, and H. Arslan, "Classifications and applications of physical layer security techniques for confidentiality: A comprehensive survey," *IEEE Commun. Surveys & Tut.*, vol. 21, no. 2, pp. 1773–1828, 2019.

[7] E.-K. Lee, M. Gerla, and S. Y. Oh, "Physical layer security in wireless smart grid," *IEEE Commun. Mag.*, vol. 50, no. 8, pp. 46–52, 2012.

[8] Z. Shu, Y. Qian, and S. Ci, "On physical layer security for cognitive radio networks," *IEEE Network*, vol. 27, no. 3, pp. 28–33, 2013.

[9] D. Kapetanovic, G. Zheng, and F. Rusek, "Physical layer security for massive MIMO: An overview on passive eavesdropping and active attacks," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 21–27, 2015.

[10] N. Xie, Z. Li, and H. Tan, "A survey of physical-layer authentication in wireless communications," *IEEE Commun. Surveys & Tut.*, vol. 23, no. 1, pp. 282–310, 2020.

[11] F. Zhu, F. Gao, H. Lin, S. Jin, J. Zhao, and G. Qian, "Robust beamforming for physical layer security in BDMA massive MIMO," *IEEE J. on Sel. Areas in Commun.*, vol. 36, no. 4, pp. 775–787, 2018.

[12] W. Wang, K. C. Teh, and K. H. Li, "Artificial noise aided physical layer security in multi-antenna small-cell networks," *IEEE Trans. on Info. Foren. & Sec.*, vol. 12, no. 6, pp. 1470–1482, 2017.

[13] Y. Zou, J. Zhu, X. Wang, and V. C. M. Leung, "Improving physical-layer security in wireless communications using diversity techniques," *IEEE Network*, vol. 29, no. 1, pp. 42–48, 2015.

[14] S. Asaad, A. Bereyhi, A. M. Rabiei, R. R. Müller, and R. F. Schaefer, "Optimal transmit antenna selection for massive MIMO wiretap channels," *IEEE J. on Sel. Areas in Commun.*, vol. 36, no. 4, pp. 817–828, 2018.

[15] R. Nakai and S. Sugiura, "Physical layer security in buffer-state-based max-ratio relay selection exploiting broadcasting with cooperative beamforming and jamming," *IEEE Trans. on Info. Foren. & Sec.*, vol. 14, no. 2, pp. 431–444, 2019.

[16] T. M. Hoang, T. Q. Duong, N. S. Vo, and C. Kundu, "Physical layer security in cooperative energy harvesting networks with a friendly jammer," *IEEE Wireless Commun. Letters*, vol. 6, no. 2, pp. 174–177, Apr. 2017.

[17] D. Wang, B. Bai, W. Zhao, and Z. Han, "A survey of optimization approaches for wireless physical layer security," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 2, pp. 1878–1911, 2019.

[18] J. Wang, C. Jiang, H. Zhang, Y. Ren, K.-C. Chen, and L. Hanzo, "Thirty years of machine learning: The road to Pareto-optimal wireless networks," *IEEE Commun. Surv. & Tut.*, pp. 1–1, Jan. 2020.

[19] L. Xiao, Y. Li, G. Liu, Q. Li, and W. Zhuang, "Spoofing detection with reinforcement learning in wireless networks," in *2015 IEEE Global Commun. Conf.*, 2015, pp. 1–5.

[20] C. Pei, N. Zhang, X. S. Shen, and J. W. Mark, "Channel-based physical layer authentication," in *2014 IEEE Global Commun. Conf.*, 2014, pp. 4114–4119.

[21] K. Besser, P. Lin, C. R. Janda, and E. A. Jorswieck, "Wiretap code design by neural network autoencoders," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3374–3386, 2020.

[22] 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, 2017.

[23] F. Pan, Z. Pang, H. Wen, M. Luvisotto, M. Xiao, R. Liao, and J. Chen, "Threshold-free physical layer authentication based on machine learning for industrial wireless CPS," *IEEE Trans. on Ind. Info.*, vol. 15, no. 12, pp. 6481–6491, 2019.

[24] A. Weinand, M. Karrenbauer, R. Sattiraju, and H. Schotten, "Application of machine learning for channel based message authentication in mission critical machine type communication," in *European Wireless 2017*, Dresden, Germany, 2017, pp. 1–5.

[25] L. Senigagliesi, L. Cintioni, M. Baldi, and E. Gambi, "Blind physical layer authentication over fading wireless channels through machine learning," in *2019 IEEE Int. Workshop on Info. Foren. and Security (WIFS)*, 2019, pp. 1–6.

[26] X. Qiu, T. Jiang, S. Wu, and M. Hayes, "Physical layer authentication enhancement using a Gaussian mixture model," *IEEE Access*, vol. 6, pp. 53 583–53 592, 2018.

[27] L. Xiao, X. Wan, and Z. Han, "PHY-layer authentication with multiple landmarks with reduced overhead," *IEEE Trans. on Wirel. Commun.*, vol. 17, no. 3, pp. 1676–1687, 2018.

[28] X. Qiu, T. Jiang, S. Wu, C. Jiang, H. Yao, M. H. Hayes, and A. Benslimane, "Wireless user authentication based on KLT and Gaussian mixture model," in *2019 IEEE Wirel. Commun. and Netw. Conf. (WCNC)*, 2019, pp. 1–5.

[29] E. M. d. L. Pinto, R. Lachowski, M. E. Pellenz, M. C. Penna, and R. D. Souza, "A machine learning approach for detecting spoofing attacks in wireless sensor networks," in *2018 IEEE 32nd Int. Conf. on Adv. Info. Netw. & App. (AINA)*, 2018, pp. 752–758.

[30] D. He, C. Liu, H. Wang, and T. Q. Quek, "Learning-based wireless powered secure transmission," *IEEE Wirel. Commun. Lett.*, vol. 8, no. 2, pp. 600–603, Apr. 2019.

[31] J. Xing, T. Lv, and X. Zhang, "Cooperative relay based on machine learning for enhancing physical layer security," in *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. (PIMRC)*. IEEE, Sep. 2019.

[32] C. Li, W. Zhou, K. Yu, L. Fan, and J. Xia, "Enhanced secure transmission against intelligent attacks," *IEEE Access*, vol. 7, pp. 53 596–53 602, 2019.

[33] L. Xiao, G. Sheng, S. Liu, H. Dai, M. Peng, and J. Song, "Deep reinforcement learning-enabled secure visible light communication against eavesdropping," *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 6994–7005, Oct. 2019.

[34] Z. Miao and Y. Wang, "Physical-layer-security-oriented frequency allocation in ultra-dense-networks based on location informations," *IEEE Access*, vol. 7, pp. 90 190–90 205, 2019.

[35] D. He, C. Liu, T. Q. Quek, and H. Wang, "Transmit antenna selection in MIMO wiretap channels: A machine learning approach," *IEEE Wirel. Commun. Lett.*, vol. 7, no. 4, pp. 634–637, Aug. 2018.

[36] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Commun. Surv. & Tut.*, vol. 20, no. 4, pp. 2595–2621, Oct. 2018.

[37] A. Diro and N. Chilamkurti, "Leveraging lstm networks for attack detection in fog-to-things communications," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 124–130, 2018.

[38] M. A. Alsheikh, S. Lin, D. Niyato, and H. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surv. & Tut.*, vol. 16, no. 4, pp. 1996–2018, 2014.

[39] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surv. & Tut.*, vol. 18, no. 2, pp. 1153–1176, Apr. 2016.

[40] E. Ruzomberka, D. J. Love, C. G. Brinton, A. Gupta, C.-C. Wang, and H. V. Poor, "Challenges and opportunities for beyond-5G wireless security," *arXiv*, 2023. [Online]. Available: https://arxiv.org/abs/2303.00727

[41] W. Khalid, M. A. U. Rehman, T. V. Chien, Z. Kaleem, H. Lee, and H. Yu, "Reconfigurable intelligent surface for physical layer security in 6G-IoT: Designs, issues, and advances," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[42] M. S. J. Solaija, H. Salman, and H. Arslan, "Towards a unified framework for physical layer security in 5G and beyond networks," *IEEE Open Journal of Vehicular Technology*, vol. 3, pp. 321–343, 2022.

[43] Y. Chen, Y. Zhang, S. Maharjan, M. Alam, and T. Wu, "Deep learning for secure mobile edge computing in cyber-physical transportation systems," *IEEE Netw.*, vol. 33, no. 4, pp. 36–41, July 2019.

[44] H. Fang, X. Wang, and S. Tomasin, "Machine learning for intelligent authentication in 5G and beyond wireless networks," *IEEE Wirel. Commun.*, vol. 26, no. 5, pp. 55–61, 2019.

[45] Q. Wang, H. Sun, R. Q. Hu, and A. Bhuyan, "When machine learning meets spectrum sharing security: Methodologies and challenges," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 176–208, 2022.

[46] G. Baldini and G. Steri, "A survey of techniques for the identification of mobile phones using the physical fingerprints of the built-in components," *IEEE Commun. Surv. & Tut.*, vol. 19, no. 3, pp. 1761–1789, July 2017.

[47] S. N. Islam, Z. Baig, and S. Zeadally, "Physical layer security for the smart grid: Vulnerabilities, threats, and countermeasures," *IEEE Trans. Ind. Info.*, vol. 15, no. 12, pp. 6522–6530, Dec. 2019.

[48] M. A. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for Internet of Things (IoT) security," *IEEE Commun. Surv. & Tut.*, pp. 1–1, 2020.

[49] V.-L. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, and Y.-D. Lin, "Security and privacy for 6G: A survey on prospective technologies and challenges," *IEEE Communications Surveys Tutorials*, vol. 23, no. 4, pp. 2384–2428, 2021.

[50] Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, "Machine learning for the detection and identification of internet of things devices: A survey," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 298–320, 2022.

[51] M. Dibaei, X. Zheng, Y. Xia, X. Xu, A. Jolfaei, A. K. Bashir, U. Tariq, D. Yu, and A. V. Vasilakos, "Investigating the prospect of leveraging blockchain and machine learning to secure vehicular networks: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 683–700, 2022.

[52] A. Talpur and M. Gurusamy, "Machine learning for security in vehicular networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 24, no. 1, pp. 346–379, 2022.

[53] K. Yu and Y. J. Guo, "Statistical NLOSidentification based on AOA, TOA, and signal strength," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 1, pp. 274–286, Jan 2009.

[54] R. M. Vaghefi, J. Schloemann, and R. M. Buehrer, "NLOS mitigation in TOA-based localization using semidefinite programming," in *the 10th Workshop on Posi., Navi. and Commun. (WPNC)*, Dresden, Germany, Mar. 2013, pp. 1–6.

[55] I. Guvenc, C. Chong, and F. Watanabe, "NLOS identification and mitigation for UWB localization systems," in *the IEEE Wireless Commun. and Net. Conf. (WCNC*, Kowloon, China, Mar. 2007, pp. 1571–1576.

[56] V. Dizdarevic and K. Witrisal, "On impact of topology and cost function on LSE position determination in wireless networks," in *Proc. Workshop on Positioning, Navigation, and Commun. (WPNC)*, Hannover, Germany, Mar. 2006, pp. 129—138.

[57] D. B. Jourdan and N. Roy, "Optimal sensor placement for agent localization," in *the IEEE/ION Position, Location, and Navi. Symp.*, Apr. 2006, pp. 128–139.

[58] B. Friedlander, "Chapter 1 - Wireless direction-finding fundamentals," in *Classical and Modern Direction-of-Arrival Estimation*, T. E. Tuncer and B. Friedlander, Eds. Boston: Academic Press, 2009, pp. 1–51.

[59] N. Dey and A. S. Ashour, *Direction of Arrival Estimation and Localization of Multi-Speech Sources*. Springer, 2018.

[60] P. Laxmikanth, S. Susruthababu, L. Surendra, S. S. Babu, and D. V. Ratnam, "Enhancing the performance of AOA estimation in wireless communication using the MUSIC algorithm," in *2015 Int. Conf. on Sig. Process. and Commun. Eng. Sys.*, 2015, pp. 448–452.

[61] T. M. Hoang, T. V. Chien, T. V. Luong, S. Chatzinotas, B. Ottersten, and L. Hanzo, "Detection of spoofing attacks in aeronautical ad-hoc networks using deep autoencoders," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1010–1023, 2022.

[62] P. Kumar, L. Reddy, and S. Varma, "Distance measurement and error estimation scheme for RSSI based localization in wireless sensor networks," in *the Fifth Int. Conf. on Wireless Commun. and Sensor Net. (WCSN)*, Allahabad, Dec. 2009, pp. 1–4.

[63] D. Konings, N. Faulkner, F. Alam, F. Noble, and E. Lai, "Do RSSI values reliably map to RSS in a localization system?" in *2017 2nd Workshop on Recent Trends in Tele. Research (RTTR)*, Feb 2017, pp. 1–5.

[64] Y. Qi, H. Kobayashi, and H. Suda, "Analysis of wireless geolocation in a non-line-of-sight environment," *IEEE Trans. on Wirel. Commun.*, vol. 5, no. 3, pp. 672–681, 2006.

[65] N. O'Donoughue and J. M. F. Moura, "On the product of independent complex Gaussians," *IEEE Trans. on Sig. Proc.*, vol. 60, no. 3, pp. 1050–1063, 2012.

[66] X. Chen, D. W. K. Ng, W. H. Gerstacker, and H. Chen, "A survey on multiple-antenna techniques for physical layer security," *IEEE Commun. Surv. & Tut.*, vol. 19, no. 2, pp. 1027–1053, 2017.

[67] B. Chatterjee, D. Das, S. Maity, and S. Sen, "RF-PUF: Enhancing iot security through authentication of wireless nodes using In-Situ machine learning," *IEEE Internet of Things J.*, vol. 6, no. 1, pp. 388–398, 2019.

[68] O. H. Salim, A. A. Nasir, H. Mehrpouyan, and W. Xiang, "Multi-relay communications in the presence of phase noise and carrier frequency offsets," *IEEE Trans. on Commun.*, vol. 65, no. 1, pp. 79–94, 2017.

[69] A. A. D'Amico, L. Marchetti, M. Morelli, and M. Moretti, "Frequency estimation in OFDM direct-conversion receivers using a repeated preamble," *IEEE Trans. on Commun.*, vol. 64, no. 3, pp. 1246–1258, 2016.

[70] B. Flury, *A first course in multivariate statistics*. Springer Science & Business Media, 2013.

[71] S. Navabi, C. Wang, O. Y. Bursalioglu, and H. Papadopoulos, "Predicting wireless channel features using neural networks," in *2018 IEEE Int. Conf. on Commun. (ICC)*, 2018, pp. 1–6.

[72] S. Aldossari and K.-C. Chen, "Predicting the path loss of wireless channel models using machine learning techniques in mmwave urban communications," in *2019 22nd Int. Symp. on Wirel. Pers. Multi. Commun. (WPMC)*, 2019, pp. 1–6.

[73] T. M. Hoang, N. M. Nguyen, and T. Q. Duong, "Detection of eavesdropping attack in UAV-aided wireless systems: Unsupervised learning with one-class SVM and K-means clustering," *IEEE Wirel. Commun. Lett.*, vol. 9, no. 2, pp. 139–142, Feb. 2020.

[74] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, "Data cleaning: Overview and emerging challenges," in *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, vol. 26-June-20. Association for Computing Machinery, June 2016, pp. 2201–2206.

[75] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas, "Data preprocessing for supervised leaning," *Int. J. Comput. Sci.*, vol. 1, pp. 111–117, 2006.

[76] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Science and Info. Conf.*, London, UK, 2014, pp. 372–378.

[77] Z. Zhao and H. Liu, "Searching for interacting features in subset selection," *Intelligent Data Analysis*, vol. 13, no. 2, pp. 207–228, 2009.

[78] A. Meyer-Baese and V. Schmid, "Feature selection and extraction," in *Pattern Recognition and Signal Analysis in Medical Imaging*. Academic Press, Jan. 2014, ch. 2, pp. 21–69.

[79] F. Song, Z. Guo, and D. Mei, "Feature selection using principal component analysis," in *2010 International Conference on System Science, Engineering Design and Manufacturing Informatization*, vol. 1, 2010, pp. 27–30.

[80] X.-T. Yuan and B.-G. Hu, "Robust feature extraction via information theoretic learning," in *Proc. of the 26th Annual Int. Conf. on Machine Learning*. Montreal, Canada: Association for Computing Machinery, 2009, pp. 1193–1200.

[81] N. Wang, W. Li, P. Wang, A. Alipour-Fanid, L. Jiao, and K. Zeng, "Physical layer authentication for 5G communications: Opportunities and road ahead," *IEEE Network*, vol. 34, no. 6, pp. 198–204, 2020.

[82] A. Mukherjee and A. L. Swindlehurst, "Detecting passive eavesdroppers in the MIMO wiretap channel," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 2809–2812.

[83] F. T. Liu, K. M. Ting, and Z. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, Dec 2008, pp. 413–422.

[84] Y. Wang, J. Wong, and A. Miner, "Anomaly intrusion detection using one class SVM," in *The Fifth Annual IEEE SMC Info. Assurance Workshop*, West Point, NY, June 2004, pp. 358–364.

[85] C. M. Bishop, *Pattern Recognition and Machine Learning*. Singapore: Springer, 2006.

[86] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN model-based approach in classification," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 2888, pp. 986–996, 2003.

[87] V. Garcia, E. Debreuve, and M. Barlaud, "Fast K nearest neighbor search using GPU," in *2008 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work. CVPR Work.*, 2008.

[88] Y. Liao and V. R. Vemuri, "Use of K-nearest neighbor classifier for intrusion detection," pp. 439–448, Oct. 2002.

[89] S. Abe, *Support vector machines for pattern classification*. Springer, 2005, vol. 2.

[90] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[91] F. Perez-Cruz and O. Bousquet, "Kernel methods and their potential use in signal processing," *IEEE Sig. Process. Mag.*, vol. 21, no. 3, pp. 57–65, 2004.

[92] S. Pathak, I. Mishra, and A. Swetapadma, "An assessment of decision tree based classification and regression algorithms," in *2018 3rd International Conference on Inventive Computation Technologies (ICICT)*, 2018, pp. 92–95.

[93] B. Gupta, A. Rawat, A. Jain, A. Arora, and N. Dhami, "Analysis of various decision tree algorithms for classification in data mining," *International Journal of Computer Applications*, vol. 163, no. 8, pp. 15–19, 2017.

[94] K. Fawagreh, M. M. Gaber, and E. Elyan, "Random forests: from early developments to recent advancements," *Systems Science & Control Engineering*, vol. 2, no. 1, pp. 602–609, 2014.

[95] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, 5th ed. USA: Prentice-Hall, 2018.

[96] B. Wang, W. Shi, and Z. Miao, "Confidence analysis of standard deviational ellipse and its extension into higher dimensional Euclidean space," *PLoS One*, vol. 10, no. 3, Mar. 2015.

[97] M. Friendly, G. Monette, and J. Fox, "Elliptical insights: Understanding statistical methods through elliptical geometry," *Stat. Sci.*, vol. 28, no. 1, pp. 1–39, Feb. 2013.

[98] S. Konishi, *Introduction to multivariate analysis: linear and nonlinear modeling*. CRC Press, 2014.

[99] A. Tharwat, T. Gaber, A. Ibrahim, and A. E. Hassanien, "Linear discriminant analysis: A detailed tutorial," *AI Commun.*, vol. 30, no. 2, pp. 169–190, Jan. 2017.

[100] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. of the Eighteenth Annual ACM-SIAM Symp. on Discrete Algo.*, New Orleans, Louisiana, Jan. 2007, pp. 1027–1035.

[101] L. M. Manevitz and M. Yousef, "One-class SVMs for document classification," *Journal of Machine Learning Research*, vol. 2, pp. 139–154, Dec. 2001.

[102] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, July 2001.

[103] B. R. Preiss, *Data Structures and Algorithms with Object-Oriented Design Patterns in Java*. Wiley, 1999.

[104] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.

[105] C. K. Reddy and B. Vinzamuri, "A survey of partitional and hierarchical clustering algorithms," in *Data clustering*. Chapman and Hall/CRC, 2018, pp. 87–110.

[106] M. Charikar, V. Chatziafratis, and R. Niazadeh, "Hierarchical clustering better than average-linkage," in *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2019, pp. 2291–2304.

[107] S. Miyamoto, R. Abe, Y. Endo, and J.-I. Takeshita, "Ward method of hierarchical clustering for non-Euclidean similarity measures," in *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*. IEEE, 2015, pp. 60–63.

[108] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[109] K. Janocha and W. M. Czarnecki, "On loss functions for deep neural networks in classification," Feb. 2017. [Online]. Available: http://arxiv.org/abs/1702.05659

[110] P. Golik, P. Doetsch, and H. Ney, "Cross-entropy vs. squared error training: a theoretical and experimental comparison," in *INTERSPEECH*, 2013, pp. 1756–1760.

[111] D. M. Kline and V. L. Berardi, "Revisiting squared-error and cross-entropy functions for training neural network classifiers," *Neural Comput. Appl.*, vol. 14, no. 4, pp. 310–318, Dec. 2005.

[112] E. A. Shenouda, "A quantitative comparison of different MLP activation functions in classification," in *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3971 LNCS. Springer Verlag, 2006, pp. 849–857.

[113] D. Pedamonti, "Comparison of non-linear activation functions for deep neural networks on MNIST classification task," Apr. 2018. [Online]. Available: http://arxiv.org/abs/1804.02763

[114] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," 2013. [Online]. Available: https://arxiv.org/abs/1302.4389

[115] X. Liang, X. Wang, Z. Lei, S. Liao, and S. Z. Li, "Soft-margin softmax for deep classification," in *Neural Inf. Process.* Cham: Springer Int. Publishing, 2017, pp. 413–421.

[116] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," Tech. Rep.

[117] W. Duch and N. Jankowski, "Survey of neural transfer functions," *Comput. Surv.*, vol. 2, pp. 163–213, 1999.

[118] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 4, pp. 3133–3174, 2019.

[119] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[120] T. Alfakih, M. M. Hassan, A. Gumaei, C. Savaglio, and G. Fortino, "Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA," *IEEE Access*, vol. 8, pp. 54 074–54 084, 2020.

[121] J. Fan, Z. Wang, Y. Xie, and Z. Yang, "A theoretical analysis of deep Q-learning," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 486–489.

[122] V. Mnih and E. al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[123] C. W. Fox and S. J. Roberts, "A tutorial on variational Bayesian inference," *Artificial intelligence review*, vol. 38, no. 2, pp. 85–95, 2012.

[124] M.-N. Tran, T.-N. Nguyen, and V.-H. Dao, "A practical tutorial on variational Bayes," *arXiv preprint arXiv:2103.01327*, 2021.

[125] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[126] S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 8, pp. 1177–1193, 2012.

[127] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: a literature survey," *The Artificial Intelligence Review*, vol. 42, no. 2, p. 275, 2014.

[128] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5149–5169, 2022.

[129] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial intelligence review*, vol. 18, pp. 77–95, 2002.

[130] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 1, pp. 857–876, 2023.

[131] M. Tschannen, O. Bachem, and M. Lucic, "Recent advances in autoencoder-based representation learning," *arXiv*, 2018. [Online]. Available: https://arxiv.org/abs/1812.05069

[132] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[133] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[134] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[135] H. Hagras, "Toward human-understandable, explainable AI," *Computer*, vol. 51, no. 9, pp. 28–36, 2018.

[136] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable AI for trees," *Nature machine intelligence*, vol. 2, no. 1, pp. 56–67, 2020.

[137] Y. Zou, J. Zhu, X. Li, and L. Hanzo, "Relay selection for wireless communications against eavesdropping: a security-reliability trade-off perspective," *IEEE Network*, vol. 30, no. 5, pp. 74–79, 2016.

[138] Y. Zou, B. Champagne, W. Zhu, and L. Hanzo, "Relay-selection improves the security-reliability trade-off in cognitive radio systems," *IEEE Trans. on Commun.*, vol. 63, no. 1, pp. 215–228, 2015.

[139] L. Jin, S. Li, B. Hu, and M. Liu, "A survey on projection neural networks and their applications," pp. 533–544, Mar. 2019.

[140] Y. Yang and X. Xu, "The projection neural network for solving convex nonlinear programming," in *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4682 LNAI. Springer, Berlin, Heidelberg, 2007, pp. 174–181.

[141] S. Effati, A. Ghomashi, and A. R. Nazemi, "Application of projection neural network in solving convex programming problems," *Appl. Math. Comput.*, vol. 188, no. 2, pp. 1103–1114, May 2007.

[142] Y. Yang and J. Cao, "A feedback neural network for solving convex constraint optimization problems," *Appl. Math. Comput.*, vol. 201, no. 1-2, pp. 340–350, July 2008.

[143] Y. Xia, H. Leung, and J. Wang, "A projection neural network and its application to constrained optimization problems," *IEEE Trans. Circuits Syst. Fundam. Theory Appl.*, vol. 49, no. 4, p. 458, Apr. 2002.

[144] Q. Liu and J. Wang, "A projection neural network for constrained quadratic minimax optimization," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 11, pp. 2891–2900, Nov. 2015.

[145] A. Malek and N. Hosseinipour-Mahani, "Solving a class of non-convex quadratic problems based on generalized KKT conditions and neurodynamic optimization technique," *Kybernetika*, vol. 51, no. 5, pp. 890–908, 2015.

[146] S. Zhang, Y. Xia, and J. Wang, "A complex-valued projection neural network for constrained optimization of real functions in complex variables," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 12, pp. 3227–3238, Dec. 2015.

[147] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT'2010*. Physica-Verlag HD, 2010, pp. 177–186.

[148] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–15.

[149] M. Zaheer, S. J. Reddi, D. Sachan, S. Kale, and S. Kumar, "Adaptive methods for nonconvex optimization," in *Conf. Neural Info. Process. Syst. (NeurIPS 2018)*, Montreal, Canada, 2018, pp. 9793–9803.

[150] L. Bottou, "Online learning and stochastic approximations," *Online Learn. neural networks*, vol. 17, no. 9, 1998.

[151] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, 2011.

[152] M. Mahdavi, T. Yang, and R. Jin, "Online stochastic optimization with multiple objectives," *Adv. Neural Inf. Process. Syst.*, Nov. 2012. [Online]. Available: http://arxiv.org/abs/1211.6013

[153] S. S. Ram, A. Nedic, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, no. 3, pp. 516–545, July 2010.

[154] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," *arXiv Prepr. arXiv1109.5647*, 2012.

[155] M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.* Association for Computing Machinery, 2014, pp. 661–670.

[156] J. M. Kang, C. J. Chun, and I. M. Kim, "Deep-learning-based channel estimation for wireless energy transfer," *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2310–2313, Nov. 2018.

[157] Y. Ermoliev and R. J.-B. Wets, *Numerical techniques for stochastic optimization*. Springer-Verlag, 1988.

[158] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," in *Adv. Neural Inf. Process. Syst.*, 2011, pp. 873–881.

[159] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," *arXiv Prepr. arXiv1509.01240*, Feb. 2016.

[160] L. Bottou, "Stochastic gradient descent tricks," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 421–436.

[161] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," Apr. 2018. [Online]. Available: http://arxiv.org/abs/1804.07612

[162] X. Hu and J. Wang, "Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network," *IEEE Trans. Neural Networks*, vol. 17, no. 6, pp. 1487–1499, Nov. 2006.

[163] M. Soltanolkotabi, "Learning ReLUs via gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[164] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," *Advances in neural information processing systems*, vol. 29, 2016.

[165] R. Liao, H. Wen, S. Chen, F. Xie, F. Pan, J. Tang, and H. Song, "Multiuser physical layer authentication in internet of things with data augmentation," *IEEE Internet of Things J.*, vol. 7, no. 3, pp. 2077–2088, 2020.

[166] H. Fang, X. Wang, and L. Hanzo, "Learning-aided physical layer authentication as an intelligent process," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2260–2273, Mar. 2019.

[167] H. A. Abu Alfeilat, A. B. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, H. S. Eyal Salman, and V. S. Prasath, "Effects of distance measure choice on K-nearest neighbor classifier performance: A review," *Big data*, vol. 7, no. 4, pp. 221–248, 2019.

[168] M. Abdrabou and T. A. Gulliver, "Adaptive physical layer authentication for IoT in MIMO communication systems using support vector machine," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[169] R. Du, L. Zhen, and Y. Liu, "Physical layer authentication based on integrated semi-supervised learning in wireless networks for dynamic industrial scenarios," *IEEE Trans. on Veh. Tech.*, vol. 72, no. 5, pp. 6154–6164, 2023.

[170] S. Wang, K. Huang, X. Xu, Z. Zhong, and Y. Zhou, "CSI-based physical layer authentication via deep learning," *IEEE Wirel. Commun. Lett.*, vol. 11, no. 8, pp. 1748–1752, 2022.

[171] R. Meng, X. Xu, B. Wang, H. Sun, S. Xia, S. Han, and P. Zhang, "Physical-layer authentication based on hierarchical variational autoencoder for industrial internet of things," *IEEE Internet of Things Journal*, vol. 10, no. 3, pp. 2528–2544, 2023.

[172] T. Zhang, Y. Huo, Q. Gao, L. Ma, Y. Wu, and R. Li, "Cooperative physical layer authentication with reputation-inspired collaborator selection," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[173] Y. Wen, M. Yoshida, J. Zhang, Z. Chu, P. Xiao, and R. Tafazolli, "Machine learning based attack against artificial noise-aided secure communication," in *IEEE Int. Conf. Commun.*, vol. 2019-May. IEEE, May 2019.

[174] Q. Liu, Y. Zhu, M. Li, R. Liu, Y. Liu, and Z. Lu, "DRL-based secrecy rate optimization for RIS-assisted secure isac systems," *IEEE Transactions on Vehicular Technology*, pp. 1–5, 2023.

[175] R. Lin, H. Qiu, J. Wang, Z. Zhang, L. Wu, and F. Shu, "Physical layer security enhancement in energy harvesting-based cognitive internet of things: A GAN-powered deep reinforcement learning approach," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[176] H. Alves, R. D. Souza, M. Debbah, and M. Bennis, "Performance of transmit antenna selection physical layer security schemes," *IEEE Sig. Proc. Lett.*, vol. 19, no. 6, pp. 372–375, 2012.

[177] J. Wang, H. Ge, M. Lin, J. Wang, J. Dai, and M. Alouini, "On the secrecy rate of spatial modulation-based indoor visible light communications," *IEEE J. on Sel. Areas in Commun.*, vol. 37, no. 9, pp. 2087–2101, 2019.

[178] H. Hui, A. L. Swindlehurst, G. Li, and J. Liang, "Secure relay and jammer selection for physical layer security," *IEEE Sig. Proc. Lett.*, vol. 22, no. 8, pp. 1147–1151, 2015.

[179] H. Lee, S. H. Lee, and T. Q. S. Quek, "Deep learning for distributed optimization: Applications to wireless resource management," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2251–2266, 2019.

[180] M. Eisen, C. Zhang, L. F. O. Chamon, D. D. Lee, and A. Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2775–2790, 2019.

[181] R. Savitha, S. Suresh, and N. Sundararajan, "Projection-based fast learning fully complex-valued relaxation neural network," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 24, no. 4, pp. 529–541, 2013.

[182] J.-M. Kang, C.-J. Chun, I.-M. Kim, and D. I. Kim, "Channel tracking for wireless energy transfer: A deep recurrent neural network approach," Dec. 2018. [Online]. Available: http://arxiv.org/abs/1812.02986

[183] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Commun. Surv. & Tut.*, vol. 19, no. 1, pp. 550–586, 2017.

[184] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Adv. Eng. Softw.*, vol. 114, pp. 163–191, Dec. 2017.

[185] A. K. Kar, "Bio inspired computing - a review of algorithms and scope of applications," *Expert Syst. Appl.*, vol. 59, pp. 20–32, Oct. 2016.

[186] H. A. Abbass, "Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization," in *2003 Congr. Evol. Comput. CEC 2003 - Proc.*, vol. 3. IEEE, 2003, pp. 2074–2080.

[187] B. Tozer, T. Mazzuchi, and S. Sarkani, "Many-objective stochastic path finding using reinforcement learning," *Expert Syst. Appl.*, vol. 72, pp. 371–382, Apr. 2017.

[188] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 38, no. 3, pp. 397–415, May 2008.

[189] Y. Chang, X. Yuan, B. Li, D. Niyato, and N. Al-Dhahir, "Machine-learning-based parallel genetic algorithms for multi-objective optimization in ultra-reliable low-latency WSNs," *IEEE Access*, vol. 7, pp. 4913–4926, 2019.

[190] W. Wu, S. Wu, and B. Wang, "Robust multi-objective beamforming design for power efficient and secure communication in MU-MISO networks," *IEEE Access*, vol. 5, pp. 13 277–13 285, July 2017.

[191] Y. Zhong, X. Ge, T. Han, Q. Li, and J. Zhang, "Tradeoff between delay and physical layer security in wireless networks," *IEEE J. on Sel. Areas in Commun.*, vol. 36, no. 7, pp. 1635–1647, 2018.

[192] Y. Liu, X. Liu, X. Mu, T. Hou, J. Xu, M. Di Renzo, and N. Al-Dhahir, "Reconfigurable intelligent surfaces: Principles and opportunities," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 3, pp. 1546–1577, 2021.

[193] X. Guo, Y. Chen, and Y. Wang, "Learning-based robust and secure transmission for reconfigurable intelligent surface aided millimeter wave UAV communications," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1795–1799, 2021.

[194] T. Jiang, L. Niu, X. Tang, X. Tang, and D. Zhai, "Aerial reconfigurable intelligent surface-assisted secrecy: A learning approach," *IEEE Communications Letters*, vol. 26, no. 1, pp. 18–22, 2022.

[195] T. M. Hoang, T. Van Luong, D. Liu, and L. Hanzo, "RIS-aided AANETs: Security maximization relying on unsupervised projection-based neural networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 2214–2219, 2022.

[196] R. W. Heath, N. González-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE J. of Sel. Topics in Sig. Proc.*, vol. 10, no. 3, pp. 436–453, 2016.

[197] T. S. Rappaport, G. R. MacCartney, M. K. Samimi, and S. Sun, "Wideband millimeter-wave propagation measurements and channel models for future wireless communication system design," *IEEE Trans. Commun.*, vol. 63, no. 9, pp. 3029–3056, Sep. 2015.

[198] C. Wang and H. M. Wang, "Physical layer security in millimeter wave cellular networks," *IEEE Trans. Wirel. Commun.*, vol. 15, no. 8, pp. 5569–5585, Aug. 2016.

[199] S. Vuppala, Y. J. Tolossa, G. Kaddoum, and G. Abreu, "On the physical layer security analysis of hybrid millimeter wave networks," *IEEE Trans. Commun.*, vol. 66, no. 3, pp. 1139–1152, Mar. 2018.

[200] H. Huang, Y. Song, J. Yang, G. Gui, and F. Adachi, "Deep-learning-based millimeter-wave massive MIMO for hybrid precoding," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3027–3032, Mar. 2019.

[201] H. Khan, A. Elgabli, S. Samarakoon, M. Bennis, and C. S. Hong, "Reinforcement learning-based vehicle-cell association algorithm for highly mobile millimeter wave communication," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1073–1085, Dec. 2019.

[202] N. Wang, L. Jiao, A. Alipour-Fanid, M. Dabaghchian, and K. Zeng, "Pilot contamination attack detection for NOMA in 5G mm-wave massive MIMO networks," *IEEE Trans. Inf. Foren. Secur.*, vol. 15, pp. 1363–1378, 2020.

[203] Z. Xiao, B. Gao, S. Liu, and L. Xiao, "Learning based power control for mmwave massive MIMO against jamming," in *IEEE Global Commun. Conf.* IEEE, 2018.

[204] D. Karunatilaka, F. Zafar, V. Kalavally, and R. Parthiban, "LED based indoor visible light communications: State of the art," *IEEE Commun. Surv. & Tut.*, vol. 17, no. 3, pp. 1649–1678, July 2015.

[205] A. Mostafa and L. Lampe, "Physical-layer security for MISO visible light communication channels," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 9, pp. 1806–1818, Sep. 2015.

[206] F. Wang, C. Liu, Q. Wang, J. Zhang, R. Zhang, L. L. Yang, and L. Hanzo, "Secrecy analysis of generalized space-shift keying aided visible light communication," *IEEE Access*, vol. 6, pp. 18 310–18 324, Jan. 2018.

[207] ——, "Optical jamming enhances the secrecy performance of the generalized space-shift-keying-aided visible-light downlink," *IEEE Trans. Commun.*, vol. 66, no. 9, pp. 4087–4102, Sep. 2018.

[208] M. D. Soltani, X. Wu, M. Safari, and H. Haas, "Bidirectional user throughput maximization based on feedback reduction in LiFi networks," *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 3172–3186, July 2018.

[209] M. Z. Chowdhury, M. T. Hossan, A. Islam, and Y. M. Jang, "A comparative survey of optical wireless technologies: Architectures and applications," pp. 9819–9840, Jan. 2018.

[210] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free massive MIMO versus small cells," *IEEE Trans. on Wirel. Commun.*, vol. 16, no. 3, pp. 1834–1850, 2017.

[211] T. M. Hoang, H. Q. Ngo, T. Q. Duong, H. D. Tuan, and A. Marshall, "Cell-free massive MIMO networks: Optimal power control against active eavesdropping," *IEEE Trans. Commun.*, vol. 66, no. 10, pp. 4724–4737, Oct. 2018.

[212] T. T. Vu, D. T. Ngo, N. H. Tran, H. Q. Ngo, M. N. Dao, and R. H. Middleton, "Cell-free massive MIMO for wireless federated learning," 2019. [Online]. Available: https://arxiv.org/abs/1909.12567

[213] M. Mohamed and M. Cheffena, "Received signal strength based gait authentication," *IEEE Sens. J.*, vol. 18, no. 16, pp. 6727–6734, Aug. 2018.

[214] H. Moosavi and F. M. Bui, "Delay-aware optimization of physical layer security in multi-hop wireless body area networks," *IEEE Trans. Inf. Foren. Secur.*, vol. 11, no. 9, pp. 1928–1939, Sep. 2016.

[215] R. Dautov and G. R. Tsouri, "Securing while sampling in wireless body area networks with application to electrocardiography," *IEEE J. Biomed. Heal. Informatics*, vol. 20, no. 1, pp. 135–142, Jan. 2016.

[216] P. V. R. Ferreira, R. Paffenroth, A. M. Wyglinski, T. M. Hackett, S. G. Bilen, R. C. Reinhart, and D. J. Mortensen, "Reinforcement learning for satellite communications: From LEO to deep space operations," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 70–75, May 2019.

[217] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.

[218] X. Huang, J. A. Zhang, R. P. Liu, Y. J. Guo, and L. Hanzo, "Airplane-aided integrated networking for 6G wireless: Will it work?" *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 84–91, Sep. 2019.

[219] J. Liu, Y. Shi, Z. M. Fadlullah, and N. Kato, "Space-air-ground integrated network: A survey," *IEEE Commun. Surv. & Tut.*, vol. 20, no. 4, pp. 2714–2741, 2018.

[220] N. Hosseinidehaj, Z. Babar, R. Malaney, S. X. Ng, and L. Hanzo, "Satellite-based continuous-variable quantum communications: State-of-the-art and a predictive outlook," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 1, pp. 881–919, Jan. 2019.

[221] D. E. Bruschi, T. Ralph, I. Fuentes, T. Jennewein, and M. Razavi, "Spacetime effects on satellite-based quantum communications," *Phys. Rev. D - Part. Fields, Gravit. Cosmol.*, vol. 90, no. 4, Sep. 2013. [Online]. Available: http://arxiv.org/abs/1309.3088

[222] F. Zarmehi and M. Houshmand, "Controlled bidirectional quantum secure direct communication network using classical XOR operation and quantum entanglement," *IEEE Commun. Lett.*, vol. 20, no. 10, pp. 2071–2074, Oct. 2016.

[223] S. Nauerth, F. Moll, M. Rau, C. Fuchs, J. Horwath, S. Frick, and H. Weinfurter, "Air-to-ground quantum communication," *Nat. Photonics*, vol. 7, no. 5, pp. 382–386, May 2013.

[224] J. Y. Hu, B. Yu, M. Y. Jing, L. T. Xiao, S. T. Jia, G. Q. Qin, and G. L. Long, "Experimental quantum secure direct communication with single photons," *Light Sci. Appl.*, vol. 5, no. 9, pp. e16 144–e16 144, Sep. 2016.

[225] D. Gottesman and H. K. Lo, "Proof of security of quantum key distribution with two-way classical communications," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 457–475, Feb. 2003.

[226] M. Schuld and N. Killoran, "Quantum machine learning in feature Hilbert spaces," *Phys. Rev. Lett.*, vol. 122, no. 4, p. 040504, Feb. 2019.

[227] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," *Contemp. Phys.*, vol. 56, no. 2, pp. 172–185, Apr. 2015.

[228] Y. Mezquita, R. S. Alonso, R. Casado-Vara, J. Prieto, and J. M. Corchado, "A review of K-NN algorithm based on classical and quantum machine learning," in *International Symposium on Distributed Computing and Artificial Intelligence*. Springer, 2020, pp. 189–198.

[229] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Physical review letters*, vol. 113, no. 13, p. 130503, 2014.

[230] M. Schuld, I. Sinayskiy, and F. Petruccione, "The quest for a quantum neural network," *Quantum Information Processing*, vol. 13, no. 11, pp. 2567–2586, 2014.

[231] S. Lu and S. L. Braunstein, "Quantum decision tree classifier," *Quantum information processing*, vol. 13, no. 3, pp. 757–770, 2014.

[232] C. Blank, D. K. Park, J.-K. K. Rhee, and F. Petruccione, "Quantum classifier with tailored quantum kernel," Sep. 2019. [Online]. Available: http://arxiv.org/abs/1909.02611

[233] Afham, A. Basheer, and S. K. Goyal, "Quantum K-nearest neighbor machine learning algorithm," Mar. 2020. [Online]. Available: http://arxiv.org/abs/2003.09187

[234] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, Sep. 2017.

[235] C. Zhao, L. Huang, Y. Zhao, and X. Du, "Secure machine-type communications toward LTE heterogeneous networks," *IEEE Wirel. Commun.*, vol. 24, no. 1, pp. 82–87, 2017.

[236] F. J. Lopez-Martinez, G. Gomez, and J. M. Garrido-Balsells, "Physical-layer security in free-space optical communications," *IEEE Photonics Journal*, vol. 7, no. 2, pp. 1–14, 2015.

[237] X. Zhao, H. Chen, and J. Sun, "On physical-layer security in multiuser visible light communication systems with non-orthogonal multiple access," *IEEE Access*, vol. 6, pp. 34 004–34 017, 2018.