Deep Learning-based Physical-Layer Secret Key Generation for FDD Systems

Xinwei Zhang, *Student Member, IEEE,* Guyue Li, *Member, IEEE,* Junqing Zhang, Aiqun Hu, *Senior Member, IEEE,* Zongyue Hou, Bin Xiao, *Senior Member, IEEE*

***Abstract*—Physical-layer key generation (PKG) establishes cryptographic keys from highly correlated measurements of wireless channels, which relies on reciprocal channel characteristics between uplink and downlink, is a promising wireless security technique for Internet of Things (IoT). However, it is challenging to extract common features in frequency division duplexing (FDD) systems as uplink and downlink transmissions operate at different frequency bands whose channel frequency responses are not reciprocal any more. Existing PKG methods for FDD systems have many limitations, i.e., high overhead and security problems. This paper proposes a novel PKG scheme that uses the feature mapping function between different frequency bands obtained by deep learning to make two users generate highly similar channel features in FDD systems. In particular, this is the first time to apply deep learning for PKG in FDD systems. We first prove the existence of the band feature mapping function for a given environment and a feedforward network with a single hidden layer can approximate the mapping function. Then a Key Generation neural Network (KGNet) is proposed for reciprocal channel feature construction, and a key generation scheme based on the KGNet is also proposed. Numerical results verify the excellent performance of the KGNet-based key generation scheme in terms of randomness, key generation ratio, and key error rate. Besides, the overhead analysis shows that the method proposed in this paper can be used for resource-contrained IoT devices in FDD systems.**

***Index Terms*—Physical-layer security, frequency division duplexing, deep learning, secret key generation.**

I. Introduction

With the rapid development of the fifth generation (5G) and beyond communication systems, the security of wireless communication has received increasing attention [1]. Due to the open nature of wireless channels, attackers can initiate various attacks such as eavesdropping, which pose a huge threat to wireless security. Traditionally, the cryptographic approaches including symmetric key cryptography and asymmetric key cryptography have been used to protect confidential information from eavesdroppers [2]. However, due to the three typical characteristics of 5G Internet of Things (IoT) networks, concerning mobility, massive IoT with resource constraints and heterogeneous hierarchical architecture, the traditional cryptographic mechanisms face problems such as difficulty in key distribution, excessive reliance on mathematical complexity, etc., and are not suitable or efficient [3]. To mitigate these issues, researchers have developed a new secure communication method from the physical layer in wireless communication, termed as *physical-layer key generation (PKG)* [4]–[6]. This technique uses the inherent randomness of fading channels between two legitimate users, namely Alice and Bob, to generate keys without the need for a third party.

The realization of PKG depends on three unique propagation characteristics of electromagnetic wave, namely channel reciprocity, temporal variation and spatial decorrelation. Among them, channel reciprocity indicates that the same channel characteristics can be observed at both ends of the same link, which is the basis for key generation [5], [6]. For time division duplexing (TDD) systems, both the uplink and downlink are in the same carrier frequency band, and the channel responses obtained by Alice and Bob are reciprocal. However, for frequency division duplexing (FDD) systems, the uplink and downlink transmit over different carrier frequencies, and the uplink and downlink experience different fading. Hence, most of the mutually accessible channel parameters used in TDD systems, such as received signal strength, channel gain, envelope, and phase, may be completely different between the uplink and downlink in FDD systems [7]. Therefore, it is challenging to find reciprocal channel features for key generation in FDD systems. On the other hand, FDD dominates existing cellular communications, such as LTE and narrowband IoT. Key generation for these FDD-based systems will provide information-theoretically secure keys for them, hence is strongly desirable.

There have been several key generation methods developed for FDD systems [8]–[14]. Those methods generate keys by extracting frequency-independent reciprocal channel parameters or constructing reciprocal channel gains, which have many limitations, i.e., high overhead and security problems [10]. Therefore, how to design a key generation scheme in a secure manner at a low communication overhead for FDD systems is still an open question.

This work was supported in part by the National Natural Science Foundation of China under Grant 6217011510, 61801115 and 61941115, in part by the Zhishan Youth Scholar Program of SEU (3209012002A3), in part by Jiangsu key R & D plan BE2019109. (Corresponding author: G. Li)

X. Zhang, G. Li, Z. Hou are with the School of Cyber Science and Engineering, Southeast University, Nanjing 210096, China (e-mail: zxw1998@seu.edu.cn; guyuelee@seu.edu.cn; zyhou@seu.edu.cn).

J. Zhang is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, U.K. (e-mail: junqing.zhang@liverpool.ac.uk).

A. Hu is with the School of Information Science and Engineering, and National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: aqhu@seu.edu.cn).

G. Li and A. Hu are also with the Purple Mountain Laboratories for Network and Communication Security, Nanjing 210096, China.

B. Xiao is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China (e-mail: csbxiao@comp.polyu.edu.hk).

To address this open problem, we consider how to construct the reciprocal features used to generate the key in FDD systems firstly. In [15], the channel-to-channel mapping function in frequency is proved to exist under the condition that the channel mapping of the candidate position to the antenna is bijective, and the condition is realized with high probability in several practical multiple-input multiple-output (MIMO) systems. Inspired by this, we reveal the existence of band feature mapping function for a given environment in a FDD orthogonal frequency-division multiplexing (OFDM) system, which means that the feature mapping can be used to construct reciprocal features in FDD systems. However, the feature mapping between different frequency bands is difficult to be described by mathematical formulas. To solve this problem, we construct the reciprocal channel features by using deep learning to learn the channel mapping function between different frequency bands. Then a new key generation method based on the band feature mapping is proposed for FDD systems. The features of one frequency band is estimated simultaneously by both Alice and Bob to generate the key. In particular, this is the first time to apply deep learning for secret key generation in FDD systems. The main contributions of this paper are as follows.

- We prove the existence of the channel feature mapping function between different frequency bands for a given environment under the condition that the mapping function from the candidate user positions to the channels is bijective. Then, we prove that the channel feature mapping between different frequency bands can be approximated by a feedforward network with a single hidden layer. The above conclusions prove the feasibility of using deep learning to construct reciprocity features, and provide a new solution for the PKG in FDD systems.
- We propose a Key Generation neural Network (KGNet) for band feature mapping to generate reciprocal channel features, and verify the performance in the simulation. Compared with three benchmark deep learning networks, the performance of KGNet can achieve good fitting and generalization performance under low signal-noise ratio (SNR). In addition, we artificially add noise by combining the noise-free and 0 dB dataset at a certain size to construct the training dataset, which improves the robustness of KGNet under low SNR.
- Based on the KGNet, we propose a novel secret key generation scheme for FDD systems, in which Alice and Bob both estimate the features of one frequency band without any loop-back. Numerical results verify the excellent performance of the KGNet-based key generation scheme in terms of randomness, key generation ratio, and key error rate. Besides, the overhead analysis shows that the method proposed in this paper can be used for resource-contrained IoT devices in FDD systems.

The remainder of this paper is structured as follows. Section II presents the related work. The system model for FDD systems is introduced in Section III. In Section IV, we prove the existence of the band feature mapping for a given environment under a condition that the mapping function from the candidate user positions to the channels is bijective, and a feedforward network with a single hidden layer can approximate the mapping. The channel feature construction algorithm based on KGNet is presented in Section V. Section VI designs the key generation scheme. The simulation results for evaluating the performance of the KGNet and the proposed key generation scheme are provided in Section VII, which is followed by conclusions in Section VIII.

II. Related Work

This section introduces related work, including the previous FDD key generation methods and the application of deep learning in the wireless physical layer.

A. Secret Key Generation for FDD Systems

In the past few years, several secret key generation methods have been developed for FDD systems. The main methods of FDD key generation are summarized as follows.

1) Extracting the frequency-independent reciprocal channel parameters to generate the key [8], [9]. In [8], the angle and delay are used to generate the key as they are supposed to hold the reciprocity in FDD systems. However, the accurate acquisition of the angle and delay requires a lot of resources, such as large bandwidth and multiple antennas [16]. In addition, a secret key generation method based on the reciprocity of channel covariance matrix eigenvalues is also proposed in FDD systems [9]. But this method requires special configuration of the antenna array.
2) Establishing the channel with reciprocal channel gain by means of the additional reverse channel training phase to generate the key, called the loopback-based methods [10]–[13]. In [10], Alice and Bob generate the key by estimating the channel impulse response (CIR) of the combinatorial channel which is the combination of uplink and downlink channels. In [11], instead of the CIR of the combinatory channel, only the uplink channel state information (CSI) is estimated by both sides of the communication using a special CSI probing method to generate the key. Furthermore, there are two secret key generation schemes, which generate secret key by exploiting shared physical channel information on non-reciprocal forward and reverse channels [12]. However, the loopback-based key generation methods require additional reverse channel training and multiple iterative interactions, which not only increase the complexity of channel detection, but also increase the risk of eavesdropping. Furthermore, it has been proved not security in [17].
3) Constructing reciprocal features based on the prior knowledge of the channel model by separating channel paths to generate keys [14]. However, in the complex multi-path environment, separating the channel paths is not simple.

Through the analysis of the previous FDD key generation methods, the existing key generation methods in the FDD

systems have problems such as large overhead and insecurity. The key generation method proposed in this paper uses the mapping function between different frequency bands to construct reciprocal channel features from a new perspective. Alice and Bob only need to probe the channel once, without additional reverse channel training. In addition, we use deep learning to learn the mapping function between frequency bands, which is data-driven, so there are not many restrictions on the model, and no complicated calculations are needed to extract reciprocal channel parameters.

B. Deep Learning for Wireless Physical Layer

Deep learning has been introduced to the wireless physical layer and achieved excellent performance in many areas such as channel estimation [18], CSI feedback [19], downlink CSI prediction [20], modulation classification [21], etc.

Gao et al. [22] proposed a direct-input deep neural network (DI-DNN) to estimate channels by using the received signals of all antennas. Wen et al. [19] used deep learning technology to develop CSINet, a novel CSI sensing and recovery mechanism that learns to effectively use channel structure from training samples. Yang et al. [23] used a spare complex-valued neural network (SCNet) for the downlink CSI prediction in FDD massive MIMO systems. Safari et al. [24] proposed a convolutional neural network (CNN) and a generative neural network (GAN) for predicting downlink CSI by observing the downlink CSI. To the best of the authors' knowledge, there is no study on deep learning-based secret key generation method for FDD systems.

Alrabeiah et al. [15] used a fully-connected neural network to learn and approximate the channel-to-channel mapping function. Inspired by this, we propose KGNet to generate reciprocal channel features and propose a KGNet-based key generation scheme for FDD systems. The KGNet is able to learn the mapping function by off-line training. After training, KGNet is used to directly predict the reciprocal features. Therefore, the method proposed in this paper has a small overhead in practical application and great potential for practical deployment. In particular, This paper applies deep learning to key generation for the first time.

III. System Model

A. Channel Model

Key generation involves two legitimate users, namely Alice (one base station) and Bob (one user) as well as an eavesdropper, Eve, located d m away from Bob. Alice and Bob will send signals to each other alternately and obtain channel estimation $\hat{h}_A$ and $\hat{h}_B$. In a TDD system, the channel estimation of Alice and Bob is highly correlated based on the channel reciprocity. Hence we can design a key generation protocol, $\mathcal{K}(\cdot)$, which will convert the analog channel estimation to a digital binary sequence. The process can be expressed as

$$K_A = \mathcal{K}(\hat{h}_A), \tag{1}$$

$$K_B = \mathcal{K}(\hat{h}_B). \tag{2}$$

The above protocol has worked well in TDD-based systems, e.g., WiFi [25], ZigBee [26], and LoRa [27], [28]. However, its adoption in FDD systems is extremely challenging, because the uplink and downlink are not reciprocal any more. For the first time, this paper will employ deep learning to construct reciprocal channel features at Alice and Bob and extend key generation for FDD-based systems.

Specifically, this paper considers Alice and Bob are equipped with a single antenna and operate at the FDD mode. Alice and Bob simultaneously transmit signals on different carrier frequencies, f_{AB} and f_{BA}, respectively. We denote the links from Bob to Alice and from Alice to Bob as Band1 and Band2, respectively. The CIR consists of N paths and can be defined as

$$h(f,\tau) = \sum_{n=0}^{N-1} \alpha_n e^{-j2\pi f\tau_n + j\phi_n}\delta(\tau-\tau_n), \tag{3}$$

where f is the carrier frequency, α_n, τ_n, and ϕ_n are the magnitude, delay, and phase shift of the n^{th} path, respectively. Note that α_n depends on (i) the distance d_n between Alice and Bob, (ii) the carrier frequency f, (iii) the scattering environment. The phase ϕ_n is determined by the scatterer(s) materials and wave incident/impinging angles at the scatterer(s). The delay $\tau_n = \frac{d_n}{c}$, where c is the speed of light.

In OFDM systems, the channel frequency response (CFR) of the l^{th} sub-carrier can be expressed as

$$H(f,l) = \sum_{n=0}^{N-1} \alpha_n e^{-j2\pi f\tau_n + j\phi_n} e^{-j2\pi\tau_n f_l}, \tag{4}$$

where f_l is the frequency of the l^{th} subcarrier relative to the center frequency f. Now, we define the $1\times L$ channel vector $\mathbf{H}(f) = \{H(f,0),...,H(f,L-1)\}$ as the CFR of frequency f, and $\mathbf{H}_1 = \mathbf{H}(f_{BA})$ as the CFR of Band1, $\mathbf{H}_2 = \mathbf{H}(f_{AB})$ as the CFR of Band2, where L is the total number of sub-carrier.

B. Attack Model

Based on the assumptions of most key generation schemes [29], [30], we also focus on passive adversary. The eavesdropper, Eve, is assumed to be located at least half of the larger wavelength in Band1 and Band2 away from the legitimate users, which can be mathematically given as

$$d > \max\{\frac{c}{2f_{AB}}, \frac{c}{2f_{BA}}\}. \tag{5}$$

Since wireless channel gains decorrelate over half a wavelength in a multipath environment, Eve's channel is assumed to be independent of the channel of legitimate users. Therefore, Eve cannot infer the channel between legitimate users to generate the key from the channels he listens on.

C. Key Generation Scheme for FDD Systems

According to the channel model, the carrier frequency will affect the phase and amplitude of the CFR, and the CFR difference is more obvious in the superposition of multiple paths at different frequencies [31]. This paper aims to construct reciprocal channel features in FDD systems for Alice and Bob based on the feature mapping function between different frequency bands. In Section IV, we first proved that there is

a feature mapping function between different frequency bands that can be obtained through deep learning. We designed a KGNet-based key generation scheme, as shown in Fig. 1. It consists of two stages, i.e., the KGNet training and KGNet-based key generation stages.

- We designed a KGNet model to learn the feature mapping function, which will be described in detail in Section V. Specifically, Alice trains KGNet to learn the feature mapping function between Band1 CFR $\mathbf{H}_1$ and Band2 CFR $\mathbf{H}_2$ stored in the database. The dataset in the database can be obtained by Alice collecting the CSI obtained by multiple channel detection measurements and the CSI feedback from Bob. The trained KGNet model will be used for key generation.
- In the KGNet-based key generation stage, Alice and Bob send a pilot signal at the same time and perform channel estimation to obtain $\mathbf{H}_1$ and $\mathbf{H}_2$, respectively. Alice and Bob then preprocess their channel vector and obtain channel features $\mathbf{x}_1$ and $\mathbf{x}_2$. Alice will use KGNet to map the Band1 features $\mathbf{x}_1$ to the estimated Band2 features $\widehat{\mathbf{x}}_2$, which enables Alice and Bob to obtain highly correlated channel characteristics $\widehat{\mathbf{x}}_2$ and $\mathbf{x}_2$, respectively. They will finally perform quantization, information reconciliation, and privacy amplification to generate the same key. The key generation scheme will be described in Section VI.

IV. Deep Learning for Band Feature Mapping

In this paper, our main theme is to construct reciprocal channel features for the FDD system to generate a key according to the feature mapping function between frequency bands. However, according to (4), we CANNOT determine whether the features between different frequency bands have a definite mapping function. And if there is a mapping function, how can we map the features on one frequency band to the features on another frequency band?

In this section, we define the band feature mapping function according to the approach in [15]. Then, we prove that leveraging deep learning can find the mapping function.

A. Existence of the Band Mapping

Consider the channel model in (4), the channel is completely defined by the parameters α_n, τ_n, ϕ_n, which are functions of the environment and the carrier frequency. Supposed that Alice is fixed, for a given static communication environment, there exists a deterministic mapping function from the position P_B of Bob to the channel $\mathbf{H}(f)$ at every antenna element m of Alice [32]. Therefore, when the number of antenna is 1, there is also a deterministic position-to-channel mapping function when Alice and Bob are both equipped with a single antenna.

Definition 1: The position-to-channel mapping $\mathbf{\Phi}_f$ can be written as follows,

$$\mathbf{\Phi}_f : \{P_B\} \rightarrow \{\mathbf{H}(f)\}, \tag{6}$$

where the sets $\{P_B\}$ represent the possible positions of Bob and the sets $\{\mathbf{H}(f)\}$ represent the CFR of the corresponding channels.

Then, we investigate the existence of the mapping from the channel vector $\mathbf{H}(f)$ to the position of Bob. For that, we adopt the following assumption.

Assumption 1: The position-to-channel mapping $\mathbf{\Phi}_f : \{P_B\} \rightarrow \{\mathbf{H}(f)\}$, is bijective.

This assumption means that every position of Bob has a unique channel vector $\mathbf{H}(f)$. The bijectiveness of this mapping depends on the number of subcarries, the set of candidate user locations and the surrounding environment. In massive MIMO systems, the probability that $\mathbf{\Phi}_f$ is bijective is actually very high in practical wireless communication scenarios, and approaches 1 as the number of antennas at the BS increased [15]. The same with a OFDM-FDD system, the probability that this mapping is bijective is also very high in practical wireless communication scenarios, and approaches 1 as the number of subcarries increases. Therefore, it is reasonable to adopt Assumption 1 in OFDM-FDD systems.

Under the Assumption 1, we define the channel-to-position mapping $\mathbf{\Phi}_f^{-1}$ as the inverse of the mapping $\mathbf{\Phi}_f$, which can be written as

$$\mathbf{\Phi}_f^{-1} : \{\mathbf{H}(f)\} \rightarrow \{P_B\}. \tag{7}$$

Proposition 1: If Assumption 1 is satisfied, there exists a channel-to-channel mapping function for a given communication environment, which can be written as follows,

$$\mathbf{\Psi}_{f_{BA} \rightarrow f_{AB}} = \mathbf{\Phi}_{f_{AB}} \circ \mathbf{\Phi}_{f_{BA}}^{-1} : \{\mathbf{H}(f_{BA})\} \rightarrow \{\mathbf{H}(f_{AB})\}, \tag{8}$$

where $\mathbf{\Phi}_{f_{AB}} \circ \mathbf{\Phi}_{f_{BA}}^{-1}$ represents the composite mapping related to $\mathbf{\Phi}_{f_{AB}}$ and $\mathbf{\Phi}_{f_{BA}}^{-1}$.

Proof 1: See Appendix A.

Proposition 1 illustrates that channels on different frequency bands have a certain mapping function. Therefore, we believe that it is possible to construct reciprocal channel features from different frequency bands to generate keys.

B. Deep Learning for Band Feature Mapping

Since the channel vectors are all complex numbers and differ in the order of magnitude of each subcarrier, they cannot be directly used in the deep learning algorithm and the quantization part of subsequent key generation. Therefore, we propose a feature extraction mapping function $\boldsymbol{\xi}$ to preprocess $\mathbf{h}$, which can be written as

$$\boldsymbol{\xi}_f : \mathbf{H}(f) \rightarrow \mathbf{x}(f). \tag{9}$$

Supposed $\boldsymbol{\xi}_f$ is linear, we can denote the inverse mapping of $\boldsymbol{\xi}$ as $\boldsymbol{\xi}^{-1}$ that can be given as

$$\boldsymbol{\xi}_f^{-1} : \mathbf{x}(f) \rightarrow \mathbf{H}(f). \tag{10}$$

Next, we investigate the existence of the Band feature mapping as shown below.

Proposition 2: With Proposition 1, there exists a Band feature mapping function for a given communication environment, which can be written as follows,

$$\mathbf{\Psi}'_{f_{BA} \rightarrow f_{AB}} = \boldsymbol{\xi}_{f_{AB}} \circ \mathbf{\Psi}_{f_{BA} \rightarrow f_{AB}} \circ \boldsymbol{\xi}_{f_{AB}}^{-1} : \mathbf{x}(f_{BA}) \rightarrow \mathbf{x}(f_{AB}). \tag{11}$$

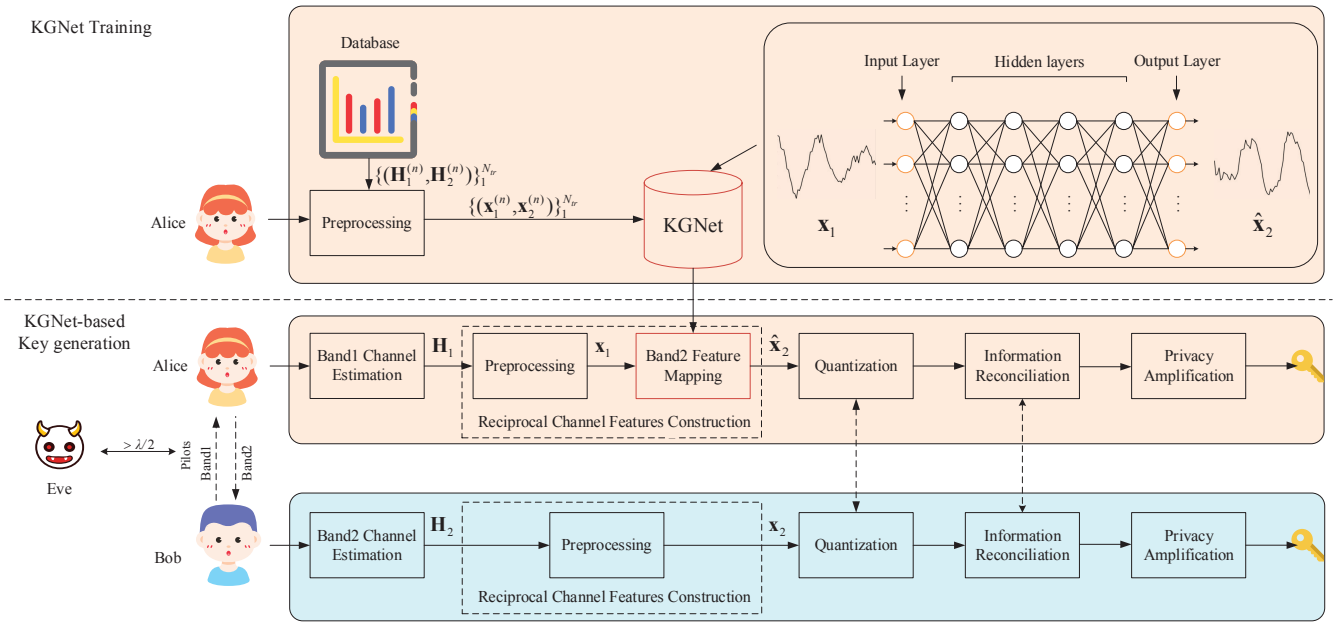


Fig. 1: The KGNet-based Key Generation scheme for FDD systems.

Proof 2: See Appendix B.

Although Proposition 2 proves that there is a feature mapping function between frequency bands, this function cannot be expressed as mathematical formulas. Therefore, based on the universe mapping [33], we introduce deep learning and obtain the following theorem.

Theorem 1: For any given error $\varepsilon > 0$, there exists a positive constant M large enough such that

$$\sup_{\boldsymbol{x}\in\mathbb{H}} \| \mathbf{NET}_M(\mathbf{x}(f_{BA}), \boldsymbol{\Omega}) - \boldsymbol{\Psi}'_{f_{BA}\to f_{AB}}(\mathbf{x}(f_{BA})) \| \leq \varepsilon, \quad \mathbb{H} = \{\mathbf{x}(f_{BA})\}, \tag{12}$$

where $\mathbf{NET}_M(\mathbf{x}(f_{BA}))$ is the output of a feedforward neural network with only one hidden layer. $\mathbf{x}(f_{BA})$, $\boldsymbol{\Omega}$ and M denote the input data, network parameters, and the number of hidden units, respectively.

Proof 3: See Appendix C.

Theorem 1 reveals that the Band feature mapping function can be approximated arbitrarily well by a feedforward network with a single hidden layer. Thus, we can use deep learning to obtain the feature mapping function between frequency bands, and generate reciprocal channel features to generate the same key.

V. KGNet-based Reciprocal Channel Features Construction

As a feedforward network with a single hidden layer can approximate the mapping of band features with different carrier frequencies, we propose a KGNet for channel feature mapping to construct reciprocal channel features. We will first introduce the dataset preprocessing. Then, we introduce the KGNet architecture and describe how to train and test the KGNet.

A. Dataset Generation

Since the wireless environment is complex and changeable, there are datasets $\{\mathbb{D}_O^e\}_{e=1}^E$ of E different environments, and $E \to \infty$. It is impossible to get datasets in all different environments. This paper currently only considers a scenario in a given environment. In the future, techniques such as transfer learning and meta learning can also be used to extend this method to new environments.

In a given environment, we denote the CSI of Band1 and Band2 as the original dataset, which is divided into the training dataset and testing dataset as $\mathbb{D}_{OTr}$ and $\mathbb{D}_{OTe}$, respectively. $\mathbb{D}_{OTr} = \{(\mathbf{H}_1^{(n)}, \mathbf{H}_2^{(n)})\}_{n=1}^{N_{tr}}$, including N_{tr} training label samples. $\mathbb{D}_{OTe} = \{(\mathbf{H}_1^{(n)}, \mathbf{H}_2^{(n)})\}_{n=1}^{N_{te}}$, including N_{te} testing label samples. It should be emphasized that the training dataset cannot include all possible channels between Alice and Bob. As long as the training dataset is sufficiently representative of the environment, the trained neural network can learn the band feature mapping of the environment. Therefore, Alice only needs to collect enough data that is sufficient to represent the environment.

B. Preprocessing of Dataset

In order to enable the deep learning model to perform efficiently, data samples usually go through a sequence of preprocessing operations, including realization and normalization. The first operation is realization. Since deep learning algorithms work in real domain, we introduce the mapping $\boldsymbol{\xi}^{(1)}$ to stack the real and imaginary parts of the complex channel vector, which can be written as

$$\boldsymbol{\xi}^{(1)} : \mathbf{H}' \to (\Re(\mathbf{H}), \Im(\mathbf{H})), \tag{13}$$

where $\Re(\cdot)$ and $\Im(\cdot)$ denote the real and imaginary parts of a matrix, vectors or scales, respectively. Through realization, the complex channel vector becomes $1 \times 2L$ real channel vector $\mathbf{H}_1'$ and $\mathbf{H}_2'$.

The second operation is normalization. Since each dimension of the original dataset usually has a different order of magnitude, directly using the original dataset to train the network will affect the efficiency of the network. Normalization is commonly used to normalize the dataset so that the range of the dataset is between 0 and 1. It is performed using the maximum and minimum value in the dataset. The value is given by:

$$\begin{cases} H_{1,max}^{'l} = \max\limits_{n=1,...,N_{tr}} \{H_1^{'l}\}^n \\ H_{1,min}^{'l} = \min\limits_{n=1,...,N_{tr}} \{H_1^{'l}\}^n \end{cases} \quad l = 0, ..., 2L-1, \quad (14)$$

where $H_1^{'l}$ is the lth element of $\mathbf{H}_1'$. We introduce the mapping $\boldsymbol{\xi}^{(2)}$ to normalize the dataset, which can be written as:

$$\boldsymbol{\xi}^{(2)} : x_1^l \rightarrow \frac{H_1^{'l} - H_{1,min}^{'l}}{H_{1,max}^{'l} - H_{1,min}^{'l}}, l = 0, ..., 2L-1, \quad (15)$$

where x_1^l is the lth element of $\mathbf{x}_1$. The normalization of $\mathbf{H}_2'$ follows the same procedure.

After the above processing, the training dataset and testing dataset as $\mathbb{D}_{Tr} = \{(\mathbf{x}_1^{(n)}, \mathbf{x}_2^{(n)})\}_{n=1}^{N_{tr}}$ and $\mathbf{D}_{Te} = \{(\mathbf{x}_1^{(n)}, \mathbf{x}_2^{(n)})\}_{n=1}^{N_{te}}$ that can be directly put into the network are obtained. We can define $\boldsymbol{\xi}_f = \boldsymbol{\xi}^{(2)} \circ \boldsymbol{\xi}^{(1)}$, which is a linear transformation that satisfies the supposition of (10).

Note that $H_{1,max}^{'l}$, $H_{1,min}^{'l}$, $H_{2,max}^{'l}$, $H_{2,min}^{'l}$ used to normalize the testing dataset are all derived from training dataset.

C. KGNet Architecture

Based on the feedforward network, we propose the KGNet, which consists of one input-layer, four hidden-layer and one output-layer. As shown in Fig. 1, the input of the network is $\mathbf{x}_1$ obtained after preprocessing $\mathbf{H}_1$. The output of the network is a cascade of nonlinear transformation of $\mathbf{x}_1$, i.e.,

$$\widehat{\mathbf{x}}_2 = \textbf{KGNet}(\mathbf{x}_1, \boldsymbol{\Omega}), \quad (16)$$

where Ω is all the trainable parameters in this network. Obviously, KGNet is used to deal with a vector regression problem. The activation functions for the hidden layers and the output layer are the rectified linear unit (ReLU) function and the sigmoid function, respectively.

D. Training and Testing

In the training stage, the dateset $\mathbb{D}_{Tr}$ is collected as the complete training dataset. In each time step, V training samples are randomly selected from $\mathbb{D}_{Tr}$ as $\mathbb{D}_{TrB}$. The KGNet is trained to minimize the difference between the output $\widehat{\mathbf{x}}_2$ and the label $\mathbf{x}_2$ by the adaptive moment estimation (ADAM) algorithm [34]. The loss function can be written as follow:

$$Loss_{\mathbb{D}}(\boldsymbol{\Omega}) = MSE(\widehat{\mathbf{x}}_2, \mathbf{x}_2) = \frac{1}{VN_{\mathbf{x}}} \sum_{v=0}^{V-1} \|\widehat{\mathbf{x}}_2^{(v)} - \mathbf{x}_2^{(v)}\|_2^2, \quad (17)$$

where V is the batch size, the superscript (v) denotes the index of the v-th training sample, $N_{\mathbf{x}}$ is the length of the vector $\mathbf{x}_2$, $\mathbb{D} = \{(\mathbf{x}_1, \mathbf{x}_2)\}_{v=0}^{V-1}$ is a batch-sized training dataset, and $\|\cdot\|_2$ denotes the ℓ_2 norm. Various loss functions can be used to train the neural network, e.g., mean square error (MSE), mean absolute error (MAE), and logcosh. Since the performance of using these loss functions is basically the same when dealing with this problem, we choose MSE as the loss function, which is also used in most works to deal with similar problems [23].

In the testing stage, the KGNet parameter $\boldsymbol{\Omega}$ is fixed. The testing dataset $\mathbb{D}_{Te}$ is generated and is used to test the performance of the KGNet. Furthermore, the dataset can also be used to evaluate the performance of the initial key after quantization.

VI. The KGNet-based Key Generation Scheme

The KGNet-based key generation scheme for FDD systems consists of five steps including channel estimation, reciprocal channel features construction, quantization, information reconciliation and privacy amplification, which are designed in this section.

A. Channel Estimation

Channel estimation is the first step for the key generation scheme, during which Alice and Bob send a pilot signal to each other simultaneously and estimate the CSI. Through this step, Alice and Bob can obtain the channel coefficient vector of Band1 and Band2, namely $\mathbf{H}_1$ and $\mathbf{H}_2$, respectively.

B. Reciprocal Channel Features Construction

Reciprocal channel features construction is a process in which Alice and Bob construct reciprocal channel features according to the CSIs estimated in Section VI-A, which is the most important step for the key generation scheme.

For Alice, this step consists of two parts: preprocessing channel coefficient vector $\mathbf{H}_1$ and Band2 feature mapping. The first part includes realization and normalization, as explained in Section V-B. In the second part, Alice performs feature mapping using the pre-trained KGNet designed in Section V-D. For Bob, this step only involves the preprocessing channel coefficient vector $\mathbf{H}_2$. Through the above steps, Alice and Bob can obtain the $1 \times 2L$ channel feature vector $\widehat{\mathbf{x}}_2$ and $\mathbf{x}_2$, respectively.

C. Quantization

Upon acquiring the channel features, Alice and Bob should apply the same quantization algorithm to convert channel features into a binary bit stream with low key error rate (KER). In this paper, based on the equal probability quantization method, we propose a Gaussian distribution-based quantization method with guard-band (GDQG) for OFDM systems.

Different from [35] that estimates the mean and variance of the amplitudes of CFR samples in a time sequence, we estimate the mean and variance of the real and imaginary parts of CFR across subcarriers, which can be expressed as

$$\mu = \frac{1}{2L} \sum_{l=0}^{2L-1} x^l, \quad (18)$$

$$\sigma^2 = \frac{1}{2L-1}\sum_{l=0}^{2L-1}(x^l - \mu)^2, \tag{19}$$

The distribution of channel features can be approximated by a Gaussian distribution. Therefore, we fit the probability of the channel features into a definite Gaussian distribution $\mathcal{N}_Q = \mathcal{N}(\mu, \sigma^2)$.

The k^{th} quantization interval is calculated as,

$$[F^{-1}(\frac{k-1}{K} + \varepsilon), F^{-1}(\frac{k}{K} - \varepsilon)], k = 2, ..., K-1, \tag{20}$$

and the 1^{st} quantization interval is $[0, F^{-1}(\frac{1}{K} - \varepsilon)]$, K^{th} quantization interval is $[F^{-1}(\frac{K-1}{K} + \varepsilon, 1]$, where F^{-1} is defined as the inverse of the cumulative distribution function (CDF) of $\mathcal{N}_Q$ and K is the quantization level. The $\varepsilon \in (0, 1/2K)$ is defined as the quantization factor, which is used to set the limit of the guard band. Then, we use the common binary encoding to convert the features into a bit stream, where all the features that are not in the quantization intervals are set to -1.

The steps of the quantization method are given in the Algorithm 1. Finally, Alice and Bob send each other indexes whose values are -1 and delete all these bits. They can get the initial secret key $\mathbf{Q}_A$ and $\mathbf{Q}_B$.

Algorithm 1 Gaussian distribution-based quantization method with guard-band

Input: The band feature vector $\mathbf{x}$; The quantization factor ε;
Output: The quantized binary sequence $\mathbf{Q}$;

```
Calculate the mean μ and variance σ^2 of the feature vector x;
Construct Gaussian distribution N_Q = N(μ, σ^2);
Calculate the inverse of the CDF F^{-1} of N_Q;
Initialize the value of ε;
Calculate the quantization intervals using (20);
for i = 0 : 2L − 1 do
    if x^i ∈ k^{th} quantization interval then
        Binary encoding
    else
        Coded as -1
    end if
end for
return Q.
```

D. Information Reconciliation and Privacy Amplification

To further reduce the KER, we can adopt information reconciliation to correct the mismatch bits. Through information reconciliation techniques, which can be implemented with protocols such as Cascade [36] or BCH code [37], etc, the performance of KER can be significantly improved. Privacy amplification can apply hash functions to distill a shorter but secret key. However, to ensure fairness of comparison, we only compare the performance of the initial secret key without information reconciliation and privacy amplification.

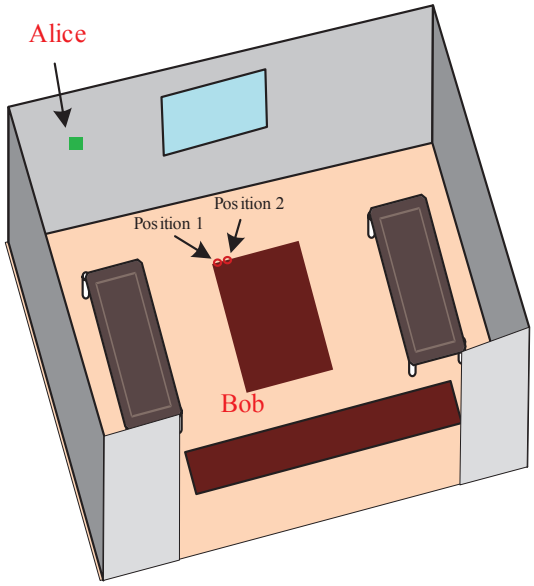


Fig. 2: An approximate depiction of the considered environment. The green little box on the ceiling represents the distributed antennas of the base station. The two maroon rectangles are two grids representing possible user locations.

VII. Simulation Results

In this section, we evaluate the performance of our proposed deep learning-based key generation scheme for FDD systems. We first describe the simulation setup and metrics, and then we discuss the simulation results and overhead.

A. Simulation Setup

We consider the indoor distributed massive MIMO scenario 'I1' that is offered by the DeepMIMO dataset [38] and is generated based on the accurate 3D ray-tracing simulator Wireless InSite [39]. As depicted in Fig. 2, the model involves one BS with single antenna and 100,000 users. We assume that the BS is Alice and multiple users are possible locations for Bob and Eve. This scenario is available at two operating frequencies 2.4 GHz and 2.5 GHz, which emulate the Band1 and Band2 carrier frequencies of the singer-input singer-output (SISO) setup in Section III. We set the number of OFDM subcarriers as 64, the number of paths as 5, and the bandwidth as 0.5 GHz to generate the dataset. This dataset constructs the channels between every candidate user location and a single antenna at the Band1 and Band2 frequencies. To form the training and testing dataset, we take the first 100,000 of the generated dataset and shuffle, and then split into a training dataset $\mathbb{D}_{OTr}$ with 80% of the total size and a testing dataset $\mathbb{D}_{OTe}$ with the rest 20%. These datasets are used to train the KGNet and evaluate the performance of the proposed key generation scheme.

The KGNet is implemented on a workstation with one Nvidia GeForce GTX 1660Ti GPU and one Inter(R) Core(TM)

TABLE I: Parameters for the KGNet.

Parameter	Value
Number of neurons in hidden layers	(512,1024,1024,512)
Optimization	ADAM [34]
Kernel initializer	glorot_uniform
Bias initializer	zeros
Exponential decay rates for ADAM: (ρ_1, ρ_2)	(0.9,0.999)
Disturbance factor for ADAM	1e-8
Learning rate	1e-3
Number of epochs	500
Batch size	128
Number of training samples	80,000
Number of testing samples	20,000

i7-9700 CPU. Tensorflow 2.1 is employed as the deep learning framework. The parameters of the KGNet are given in Table I.

B. Performance Metrics

We use the *Normalized Mean Square Error (NMSE)* to evaluate the predictive accuracy of the network, which is defined as

$$\mathrm{NMSE} = E\left[\frac{\| \widehat{\mathbf{x}}_2 - \mathbf{x}_2 \|_2^2}{\| \mathbf{x}_2 \|_2^2}\right], \tag{21}$$

where $E[\cdot]$ represents the expectation operation.

We evaluate the performance of the initial key using the following metrics.

- *Key Error Rate (KER)*: It is defined as the number of error bits divided by the number of total key bits.
- *Key Generation Ratio (KGR)*: It is defined as the number of initial key bits divided by the number of subcarriers. If all the real and imaginary features of the subcarriers are used to generate the key bits and the guard band is not used during quantization, then the KGR reaches a maximum of 2.
- *Randomness*: The randomness reveals the distribution of bit streams. The National Institute of Standards and Technology (NIST) statistical test [40] will be used for the randomness test for the key.

We evaluate the performance at Eve by using the *Normalized Vector Distance (NVD)* between two vectors $\mathbf{K}_1$ and $\mathbf{K}_2$, which is defined as

$$\mathrm{NVD}(\mathbf{K}_1, \mathbf{K}_2) = \frac{\| \mathbf{K}_1 - \mathbf{K}_2 \|_2^2}{\| \mathbf{K}_2 \|_2^2}. \tag{22}$$

C. Results

In this section, we evaluate the performance of KGNet and the initial key. Then, the security of the secret generation scheme proposed in this paper is analyzed.

1) The Performance of KGNet: The performance of the KGNet is critical to whether Alice and Bob can generate highly reciprocal channel characteristics. Besides KGNet, we also considered the following three benchmark models.

- FNN. A FNN is originally designed in [41] for uplink/downlink channel calibration for massive MIMO systems, which can be used for the band feature mapping for SISO-OFDM system. It consists of three hidden layers and we choose the numbers of neurons in the hidden layer are (512, 1024, 512) by adjustments. FNN in [41] differs from KGNet proposed in this paper not only in the number of layers of the network, but also in all layers of the FNN, the tanh function is used as the activation function.
- Advanced-FNN. It has the same architecture as FNN but its activation functions are the same as KGNet.
- 1D-CNN. Its structure is to add a 1D convolutional layer with 8 convolution kernels of size 8 and a max pooling layer with step size 2 and pooling kernel size 2 between the input layer and the first hidden layer of Advanced-FNN.

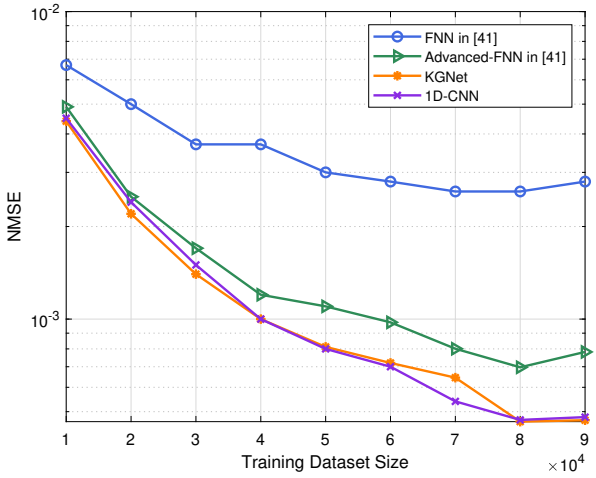


Fig. 3: The NMSE of the four networks versus the size of training dataset.

We will compare the KGNet with the three benchmark networks from three aspects, namely, the size of training data required, the fitting performance of the network, and the generalization performance under various SNR.

The size of training dataset is crucial in the KGNet training. Fig. 3 shows how the performance of the KGNet improves as the size of the training dataset grows. The network is trained from scratch for every training dataset size and is tested on the fixed 10,000 test set. We compare the size of training data required for each of the four networks to achieve optimal results. As expected, the performance of the four networks has improved as the size increased. However, all four networks achieve a stable NMSE after 80,000 training sets. Hence, we use 80,000 sets of data for training and 20,000 sets of data for testing in the rest of the paper.

In Fig. 4, we compare their fitting performance. From the perspective of fitting performance, KGNet and 1D-CNN structures can be selected for band feature mapping. In addition, the NMSE of the four networks is at a basic level when the number of epochs is 500, which is used in the rest of the paper.

Moreover, the generalization of the network is critical to the performance of key generation. The four networks were

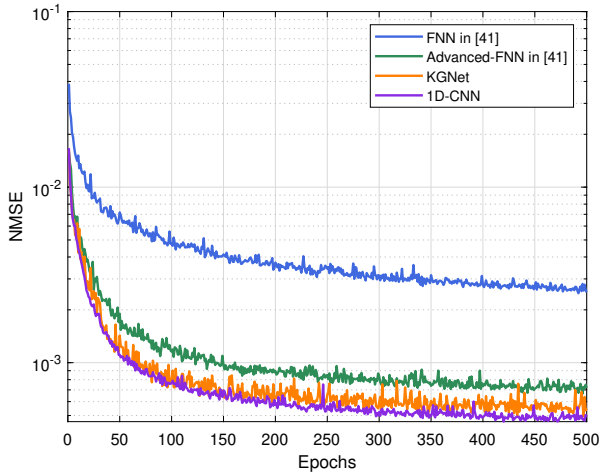


Fig. 4: The NMSE of the four networks versus the number of epochs.

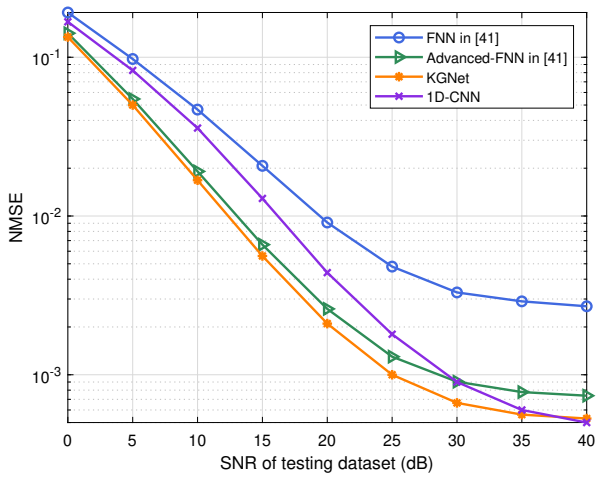


Fig. 5: The NMSE of the four networks versus SNR of test sets.

trained with the original dataset (without noise). We then added complex white Gaussian noise to the test dataset to generate new test datasets with different SNR levels in the range of 0-40 dB with a 5 dB step. As shown in Fig. 5, as the SNR increases, the NMSE of the network decreased. Compared with the other three networks, KGNet has better performance in the range of 0-40 dB.

Based on the analysis in the above three aspects, KGNet is the best choice for band feature mapping. Besides the above model, we tried other architectures by adding more convolutional layers and full connection layers based on the 1D-CNN and Advanced-FNN. Although the performance of network is slightly improved under high SNR, it also declines under low SNR. Hence they are not compared in this paper.

2) Adding Artificial Noise to Improve Robustness: In order to improve the generalization performance of the network under low SNR, we added artificial noise to the original training dataset. Firstly, we used the dataset under a single

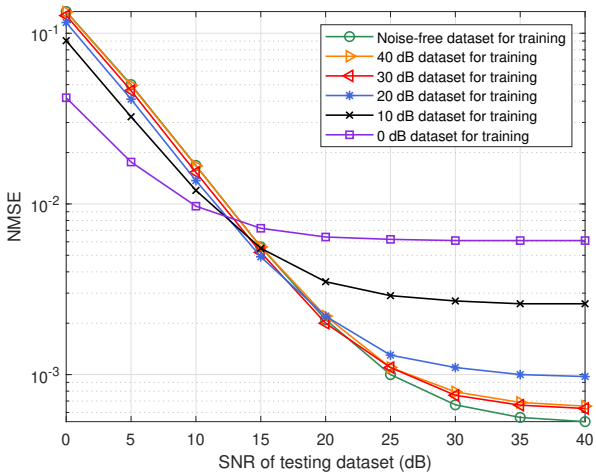


Fig. 6: The NMSE of the KGNet trained with different SNR dataset versus SNR of testing dataset.

SNR as the training dataset, and observed the NMSE of KGNet when tested under different SNRs. As shown in Fig. 6, when the SNR of the training dataset is lower, the NMSE of KGNet is better when the SNR is lower than 15 dB, and the NMSE is worse when the SNR exceeds 15 dB. When the training dataset is the original noise-free data, the network has the best performance under high SNR while the network performs well under low SNR when the training dataset is SNR of 0 dB. Therefore, we choose to cross the original noise-free dataset with the dataset with SNR of 0 dB, so that the KGNet can achieve excellent performance under 0 - 40 dB.

Then, we combined the noise-free dataset and the 0 dB dataset in different sizes to form the new training dataset. We constructed five cross datasets:

- Cross dataset 1: A dataset combining 80,000 sets of 0 dB dataset and 0 sets of noise-free dataset;
- Cross dataset 2: A dataset combining 60,000 sets of 0 dB dataset and 20,000 sets of noise-free dataset;
- Cross dataset 3: A dataset combining 40,000 sets of 0 dB dataset and 40,0000 sets of noise-free dataset;
- Cross dataset 4: A dataset combining 20,000 sets of 0 dB dataset and 60,000 sets of noise-free dataset;
- Cross dataset 5: A dataset combining 0 sets of 0 dB dataset and 80,000 sets of noise-free dataset.

As shown in Fig. 7, when the network is trained with the cross dataset 5, the performance of the network is the best when the SNR is above 25 dB, but its test performance under low SNR is too bad compared to the other four cross datasets. The test performance of the network trained with the cross datasets 1-4 is almost the same in the low SNR. Therefore, in order to ensure the excellent performance under high SNR and improve the performance under low SNR to a certain extent, we choose the cross dataset 4.

Finally, we used the cross dataset 4 to train the four networks, and the NMSE of the four networks tested under different SNRs is shown in Fig. 8. Compared with the network performance shown in Fig. 5, the use of cross dataset to train

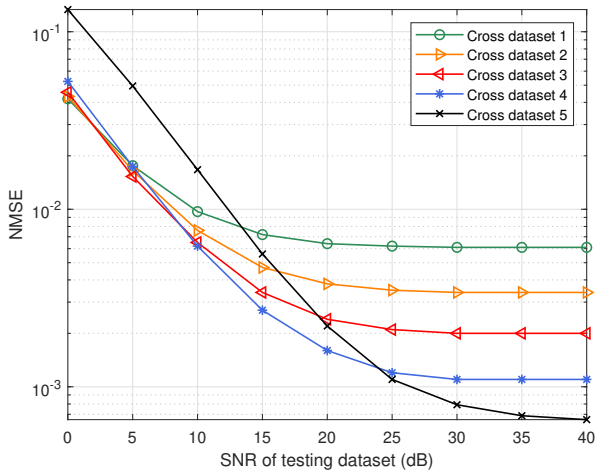


Fig. 7: The NMSE of the KGNet trained with cross datasets 1 - 5 versus SNR of testing dataset.

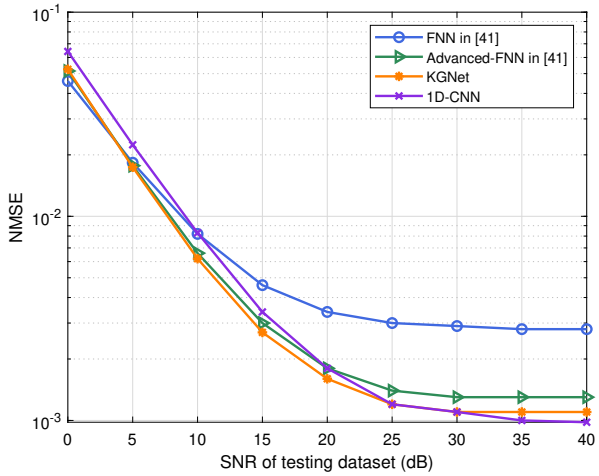


Fig. 8: The NMSE of the four networks versus SNR when all networks are trained with the cross dataset 4.

TABLE II: SNR thresholds to achieve KER = 10^{-1} when $\varepsilon = \frac{0.2}{2(K-1)}$.

Quantization level	$K = 2^1$	$K = 2^2$	$K = 2^3$
SNR threshold (dB)	5	15	–

the network reduces the performance of the four networks under high SNR, and the performance of the four networks under low SNR is improved. It is obvious that the generalization performance of the four networks has been improved, and the generalization performance of KGNet is still better than the other three networks.

3) The Performance of Initial Key: The performance of the initial key plays the most important role in secret key generation problems as it provides the capability Alice and Bob can achieve the same secret keys.

In practice, the commonly used information reconciliation methods can correct the initial key with KER less than 10^{-1}. Table II compares the lowest SNRs under different quantization levels to achieve this goal when $\varepsilon = \frac{0.2}{2(K-1)}$. $\varepsilon = \frac{0.2}{2(K-1)}$ means that regardless of the level of quantization, about 20% of the channel characteristics are discarded. When $K = 2^3$, the KER under 40 dB is 0.15, which still cannot meet the target. In order to better analyze the impact of other parameters on the performance of the initial key, the quantization level in the subsequent analysis is 2^1.

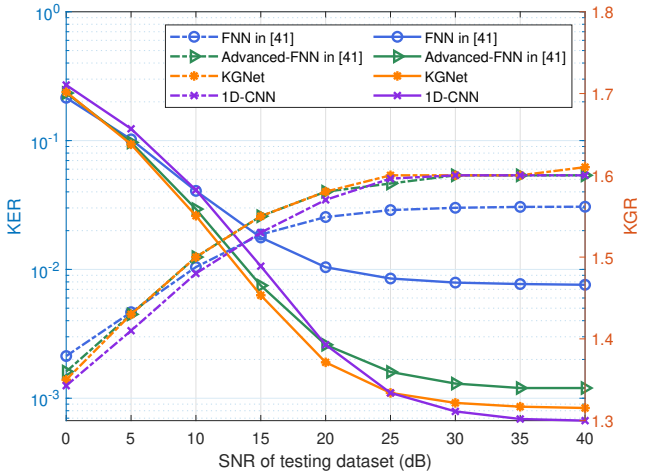


Fig. 9: The KER and KGR of the initial key based on the four networks versus SNR. All networks are trained with the cross dataset 4. The quantization factor is 0.1. The solid line represents KER and the dashed line represents KGR.

Fig. 9 compares the average KER performance of the four networks under 20,000 testing dataset. We chose the quantization factor ε of 0.1. As the SNR increases, the KER of the four networks decreases, and the KER of KGNet is always lower than the other networks. When the SNR is higher than 30 dB, the KER performance of KGNet is lower than 10^{-3}. Furthermore, Fig. 9 compares the average KGR performance of the three networks under 20,000 testing dataset. When the SNR is low and the network performance is not good enough, there will be more features in the isolation band, and its KGR performance is reduced. At the theoretical level, we choose the quantization factor to be 0.1, which means that about 20% of the channel features in the total $2L$ channel features are deleted during the quantization process, so the number of quantized key bits generated is $1.6L$ and the KGR is 1.6. As shown in Fig. 9, with the growth of SNR, the KGR performance of the four networks also continues to increase and is close to 1.6 when it is above 25 dB, which means the Gaussian distribution fits the distribution of channel features well.

Additionally, the choice of the quantization factor has a great influence on the performance of KER and KGR. Fig. 10 compares the performance of KER and KGR under the quantization factor is 0.005, 0.1, 0.2 and 0.3. The outcomes show that when ε is higher, the KER becomes smaller and the KGR becomes larger. In practice, to satisfy the high agreement, the KER need to be less than 10^{-3}. To reduce the KER as well as the information exposed during the information reconciliation, different quantization factors can

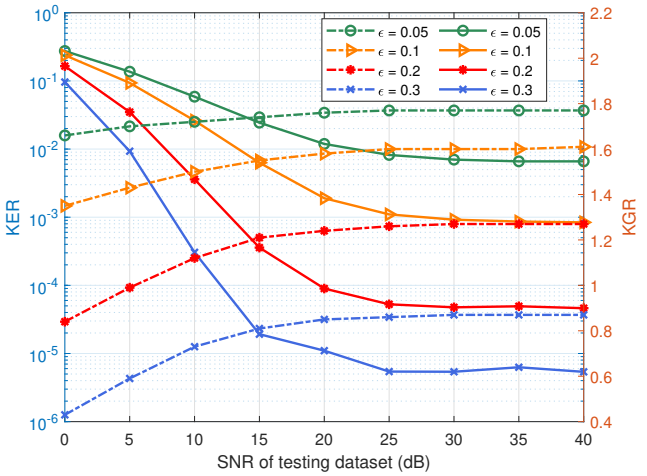


Fig. 10: The KER and KGR of the initial key based on the KGNet versus SNR. The KGNet is trained with the cross dataset 4. The solid line represents KER and the dashed line represents KGR.

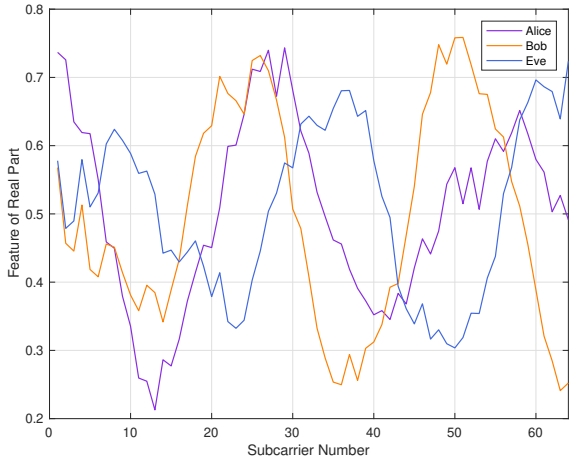
TABLE III: NIST statistical test pass ratio.

Test	Pass ratio
Approximate Entropy	0.8363
Block Frequency	0.9158
Cumulative Sums	0.9972
Discrete Fourier Transform	0.9995
Frequency	0.7103
Ranking	0.8511
Runs	0.8733
Serial	0.7461

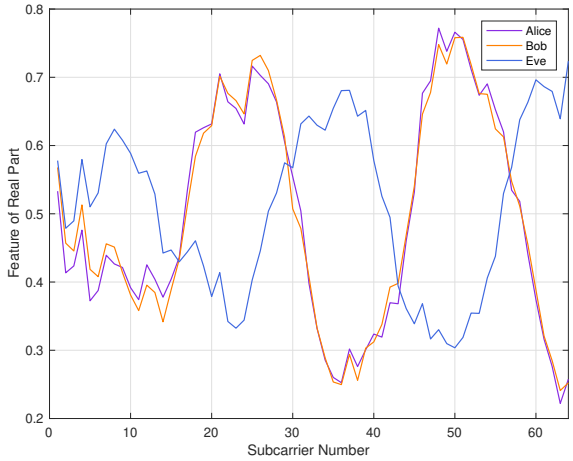
be set under different SNRs. For example, we can set ε is 0.1 when SNR exceeds 25 dB, ε is 0.2 when SNR is between 15 dB to 25 dB, ε is 0.3 when SNR is under 15 dB.

The NIST is a common tool to evaluate the randomness feature of binary sequences [29], which is also adopted in our work. The output of each test is the p-value. When the p-value is greater than a threshold, which is usually 0.01, the detected sequence passes the test. Since the quantization method we adopted was discarded in the quantization process, the length of each group of keys cannot reach the minimum requirement of 128 bits for NIST detection. We splice 20000 sets of keys together to form t sets of 128-bit keys, generally t is less than 20000. We perform 8 NIST statistical tests on $t = 15,981$ sets of keys generated under 25 dB. The serial test includes two types of tests; the serial test is deemed passed when both tests pass. Table III shows the pass rate of the test with different quantization intervals, which is the ratio of the number of sets passed to the number of all sets.

4) The Performance at Eve: The performance of Eve is related to whether the key can be eavesdropped. Since Eve is close to Bob and Bob receives the channel on f_{AB}, it is assumed that Eve eavesdrops on the channel over frequency f_{AB} without loss of generality. Alice, Bob and Eve can have channel feature vectors $\mathbf{x}_A$, $\mathbf{x}_B$ and $\mathbf{x}_E$ respectively. Fig. 11(a) shows the channel features obtained by Alice, Bob and Eve with Bob at position 1 and Eve at position 2. The



(a) Before Alice uses KGNet for channel feature mapping. SNR is 25 dB.



(b) After Alice uses KGNet for channel feature mapping. SNR is 25 dB.

Fig. 11: The performance of Alice, Bob and Eve.

distance between position 1 and position 2 is 0.15 m, which just satisfies (5) .The channel features obtained by Alice and Bob come from two channels with the same relative position and different frequencies, while those obtained by Bob and Eve come from two channels with different positions and the same frequencies. As Fig. 11(a) shown, changes in location and frequency both cause changes in channel characteristics. Fig. 11(b) shows the channel features obtained by Alice, Bob and Eve after Alice used KGNet for band feature mapping. The channel features obtained by Alice and Bob have a high degree of reciprocity, which are greatly different from those obtained by Eve.

Fig. 12 compares the NVD performance of channel features obtained by Alice and Bob before and after using KGNet for band feature mapping and between Eve and Bob under different SNRs, when Bob and Eve are fixed at positions 1 and 2, respectively. We can find that when Alice does not use KGNet for feature mapping, the NVD of channel features obtained by Alice and Bob and the NVD of channel features obtained by Eve and Bob are similar. However, after Alice

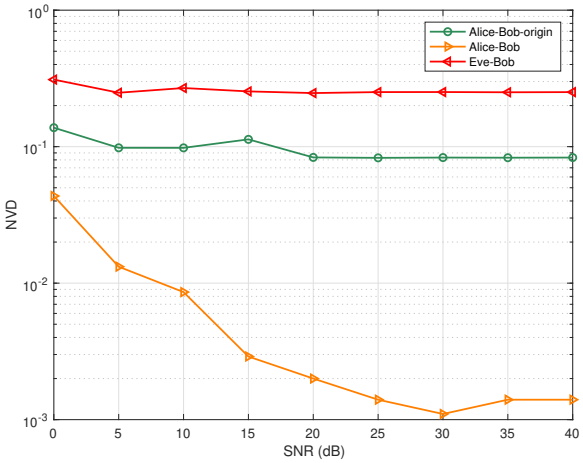


Fig. 12: The NVD of the channel features between Alice and Bob originally, between Alice and Bob after Alice uses KGNet, and between Eve and Bob versus SNR when Bob and Eve are fixed in position 1 and position 2, respectively.

uses KGNet, the NVD between the channel feature obtained by Alice and Bob are much smaller than those obtained by Eve and Bob. Therefore, we believe that even when Eve and Bob are close in the same room, Eve has limited access to the channel features that Alice and Bob can use to generate keys.

D. Overhead Analysis

To measure the complexity of the KGNet, multiple indicators are used in this paper. Table IV compares the complexity of the four networks in terms of five indicators, i.e., training time, CPU average load, GPU memory utilization, the number of float-point operations (FLOPs), and the number of trained parameters. We calculate the number of FLOPs with reference [42]. In terms of training time, it takes about 14min 42s to train the KGNet, which takes longer than other networks. In terms of the numbers of FLOPs and trained parameters, the numbers of FLOPs and training parameters required by the KGNet are nearly twice that of other networks. In terms of CPU and GPU cost, the training of the KGNet requires approximately 15.2% of the CPU load and 4.8 G of GPU memory (the total GPU memory is 9.9 G), which is similar to the cost of other types of networks. In general, the resource consumption required for the KGNet training is higher than that of other networks. However, this overhead is considered acceptable, and the performance improvement brought about by these excess consumption is more important. The reasons are summarized as follows.

- We choose to train and deploy the network at the base station, and the base station usually has a large amount of computing resources, which is sufficient to meet the resources required for network training and storage. The training time required to train the network on the base station will be greatly reduced than in the simulation. Besides, the training of the neural network can be done on the cloud without spending any resources of the base station.
- We are modeling based on location, and only need to train a network for a large area. All terminal devices in this area can establish a secure connection with the base station by using this network. As long as the environment in the area does not undergo large-scale changes, the trained network can be used forever. Even if the environment changes, we can fine-tune the network through a small number of datasets.
- The networks with better performance can generate initial keys with lower BER and higher BGR, which will reduce the overhead of subsequent information reconciliation. Therefore, from a long-term perspective, it is very cost-effective to exchange the additional cost required for training for a network with better performance.

Besides, it needs to be emphasized that compared with the key generation for TDD systems, the base station does not have any additional overhead except for the additional use of the KGNet for frequency band feature mapping in FDD systems. In particular, the terminal does not need to train and save the neural network, and there is no additional overhead other than regular key generation, which means it is suitable for resource-constrained IoT devices.

VIII. Conclusion

This paper is the first work to apply deep learning to the design of FDD key generation scheme. We first demonstrated the mapping of channel features in different frequency bands under a condition that the mapping function from the candidate user positions to the channels is bijective. Then, we proposed the KGNet for frequency band feature mapping to construct reciprocal channel feature between communication parties. Based on the KGNet, a novel secret key generation scheme for FDD systems was established. Simulation results have demonstrated that the KGNet performs better than other benchmark neural networks in terms of both fitting and generalization under low SNR. In addition, training the neural network with a cross dataset that combines noisy dataset and noise-free dataset can improve the robustness of the neural network. Moreover, the proposed KGNet-based FDD system key generation scheme is evaluated from the KER, KGR and randomness, and it is verified that it can be used for key generation for FDD systems. Our future work will extend this approach to design key generation for FDD-MIMO systems.

Appendix

A. Proof of Proposition 1

Proof 4: Under definition 1, we have the mapping $\mathbf{\Phi}_{f_{AB}}$: $\{P_B\} \rightarrow \{\mathbf{H}(f_{AB})\}$. Under assumption 1, we have the mapping $\mathbf{\Phi}^{-1}_{f_{BA}}$: $\{\mathbf{H}(f_{BA})\} \rightarrow \{P_B\}$. Since the domain of $\mathbf{\Phi}_{f_{AB}}$ is the same as co-domain of $\mathbf{\Phi}^{-1}_{f_{BA}}$, the composite mapping $\mathbf{\Psi}_{f_{BA}\rightarrow f_{AB}}$ exists.

TABLE IV: Complexity analysis and time cost for the four networks.

Network	Training time	CPU average load	GPU memory utilization	FLOPs	The number of trained parameters
FNN in [41]	12min 13s	15.2%	4.8 / 9.9 GB	2,357,120	1,181,824
Advanced-FNN in [41]	12min 25s	15.6%	4.8 / 9.9 GB	2,357,120	1,181,824
KGNet	**14min 42s**	15.2%	4.8 / 9.9 GB	**4,453,248**	**2,231,424**
1D-CNN	14min 2s	**18.0%**	**5.1 / 9.9 GB**	2,850,688	1,362,120

B. Proof of Proposition 2

Proof 5: Under proposition 1, we have the mapping $\boldsymbol{\Psi}_{f_{BA}\to f_{AB}}$. Under mapping function $\boldsymbol{\xi}_f$ and $\boldsymbol{\xi}^{-1}_{f_{AB}}$, we have the mapping $\boldsymbol{\xi}_{f_{AB}} : \mathbf{H}(f_{AB}) \to \mathbf{x}(f_{AB})$ and $\boldsymbol{\xi}^{-1}_{f_{BA}} : \mathbf{x}(f_{BA}) \to \mathbf{H}(f_{BA})$. Since the domain of $\boldsymbol{\xi}_{f_{AB}}$ is the same as co-domain of $\boldsymbol{\Psi}_{f_{BA}\to f_{AB}}$, the domain of $\boldsymbol{\Psi}_{f_{BA}\to f_{AB}}$ is the same as co-domain of $\boldsymbol{\xi}^{-1}_{f_{BA}}$, the mapping $\boldsymbol{\Psi}^{'}_{f_{BA}\to f_{AB}}$ exists.

C. Proof of Theorem 1

Proof 6: (i) Since $\mathbf{h}_{f_{BA}}$ is bounded and closed, $\mathbf{x}_{f_{BA}}$ obtained after linear change is still bounded and closed, $\mathbb{H}$ is a compact set; (ii) Since $\boldsymbol{\Phi}_{f_{AB}}$ and $\boldsymbol{\Phi}^{-1}_{f_{BA}}$ are continuous mappig and the composition of continuous mappings is still a continuous mappinng, we know $\boldsymbol{\Psi}_{f_{BA}\to f_{AB}}$ is a continuous function. (iii) Since $\boldsymbol{\Psi}_{f_{BA}\to f_{AB}}$and $\boldsymbol{\xi}$ are a continuous function, we know $\boldsymbol{\Psi}^{'}_{f_{BA}\to f_{AB}}(\mathbf{x}(f_{BA}))$ is also a continuous function for $\forall \mathbf{x}(f_{BA}) \in \mathbb{H}$. Based on (i), (ii), (iii) and universal approximation theorem [33, Theorem 1], Theorem 1 is proved.

REFERENCES

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

[2] W. Stallings, *Cryptography and Network Security: Principles and Practice*. Prenntice Hall, 2011.

[3] N. Wang, P. Wang, A. Alipour-Fanid, L. Jiao, and K. Zeng, "Physical-layer security of 5G wireless networks for IoT: Challenges and opportunities," *IEEE Internet of Things J.*, vol. 6, no. 5, pp. 8169–8181, Jul. 2019.

[4] J. Wan, A. Lopez, and M. A. A. Faruque, "Physical layer key generation: Securing wireless communication in automotive cyber-physical systems," *ACM Trans. Cyber-Phys. Syst.*, vol. 3, no. 2, Oct. 2018.

[5] J. Zhang, T. Q. Duong, A. Marshall, and R. Woods, "Key generation from wireless channels: A review," *IEEE Access*, vol. 4, pp. 614–626, Jan. 2016.

[6] J. Zhang, G. Li, A. Marshall, A. Hu, and L. Hanzo, "A new frontier for IoT security emerging from three decades of key generation relying on wireless channels," *IEEE Access*, vol. 8, pp. 138 406–138 446, Jul. 2020.

[7] G. Li, C. Sun, J. Zhang, E. Jorswieck, B. Xiao, and A. Hu, "Physical layer key generation in 5G and beyond wireless communications: Challenges and opportunities," *Entropy*, vol. 21, no. 5, p. 497, May 2019.

[8] W. Wang, H. Jiang, X. Xia, P. Mu, and Q. Yin, "A wireless secret key generation method based on Chinese remainder theorem in FDD systems," *Sci. China Inf. Sci.*, vol. 55, no. 7, pp. 1605–1616, Jul. 2012.

[9] B. Liu, A. Hu, and G. Li, "Secret key generation scheme based on the channel covariance matrix eigenvalues in FDD systems," *IEEE Commun. Lett.*, vol. 23, no. 9, pp. 1493–1496, Sep. 2019.

[10] S. J. Goldberg, Y. C. Shah, and A. Reznik, "Method and apparatus for performing JRNSO in FDD, TDD and MIMO communications," U.S. Patent 8 401 196 B2, Mar. 19, 2013.

[11] X. Wu, Y. Peng, C. Hu, H. Zhao, and L. Shu, "A secret key generation method based on CSI in OFDM-FDD system," in *Proc. IEEE Globecom Workshops. (GC Wkshps)*, Atlanta, GA, United states, Dec. 2013, pp. 1297–1302.

[12] D. Qin and Z. Ding, "Exploiting multi-antenna non-reciprocal channels for shared secret key generation," *IEEE Trans.Inf. Forensics Secur.*, vol. 11, no. 12, pp. 2693–2705, Dec. 2016.

[13] A. M. Allam, "Channel-based secret key establishment for FDD wireless communication systems," *Commun. Appl. Electron*, vol. 7, no. 9, pp. 27–31, Nov. 2017.

[14] G. Li, A. Hu, C. Sun, and J. Zhang, "Constructing reciprocal channel coefficients for secret key generation in FDD systems," *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2487–2490, Dec. 2018.

[15] M. Alrabeiah and A. Alkhateeb, "Deep learning for TDD and FDD massive MIMO: Mapping channels in space and frequency," in *Proc. 53rd Asilomar Conf. Rec. Asilomar Conf. Signals Syst. Comput. (ACSSC)*, Pacific Grove, CA, United states, Nov. 2019, pp. 1465–1470.

[16] D. Vasisht, S. Kumar, H. Rahul, and D. Katabi, "Eliminating Channel Feedback in Next-Generation Cellular Networks," in *Proc. ACM SIGCOMM*, Florianopolis, Brazil, Aug. 2016, p. 398–411.

[17] L. Peng, G. Li, J. Zhang, R. Woods, M. Liu, and A. Hu, "An investigation of using loop-back mechanism for channel reciprocity enhancement in secret key generation," *IEEE Trans Mob Comput*, vol. 18, no. 3, pp. 507–519, May 2018.

[18] Y. Yang, F. Gao, X. Ma, and S. Zhang, "Deep learning-based channel estimation for doubly selective fading channels," *IEEE Access*, vol. 7, pp. 36 579–36 589, Mar. 2019.

[19] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep Learning for Massive MIMO CSI Feedback," *IEEE Wireless Commun. Lett*, vol. 7, no. 5, pp. 748–751, Oct. 2018.

[20] J. Wang, Y. Ding, S. Bian, Y. Peng, M. Liu, and G. Gui, "UL-CSI data driven deep learning for predicting DL-CSI in cellular FDD systems," *IEEE Access*, vol. 7, pp. 96 105–96 112, Jul. 2019.

[21] Y. Lin, Y. Tu, and Z. Dou, "An Improved Neural Network Pruning Technology for Automatic Modulation Classification in Edge Devices," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5703–5706, Mar. 2020.

[22] S. Gao, P. Dong, Z. Pan, and G. Y. Li, "Deep Learning Based Channel Estimation for Massive MIMO With Mixed-Resolution ADCs," *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 1989–1993, Aug. 2019.

[23] Y. Yang, F. Gao, G. Y. Li, and M. Jian, "Deep Learning-Based Downlink Channel Prediction for FDD Massive MIMO System," *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 1994–1998, Nov. 2019.

[24] M. S. Safari, V. Pourahmadi, and S. Sodagari, "Deep UL2DL: Data-Driven Channel Knowledge Transfer From Uplink to Downlink," *IEEE Open J. Veh. Technol.,*, vol. 1, no. 5, pp. 29–44, Dec. 2020.

[25] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: extracting a secret key from an unauthenticated wireless channel," in *Proc. Annu Int Conf Mobile Comput Networking (Mobicom)*, San Francisco, CA, United states, Sep. 2008, pp. 128–139.

[26] T. Aono, K. Higuchi, T. Ohira, B. Komiyama, and H. Sasaoka, "Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels," *IEEE Trans. Antennas Propag.*, vol. 53, no. 11, pp. 3776–3784, Nov. 2005.

[27] W. Xu, S. Jha, and W. Hu, "LoRa-Key: Secure key generation system for LoRa-based network," *IEEE Internet of Things J.*, vol. 6, no. 4, pp. 6404–6416, Aug. 2019.

[28] H. Ruotsalainen, J. Zhang, and S. Grebeniuk, "Experimental Investigation on Wireless Key Generation for Low-Power Wide-Area Networks," *IEEE Internet of Things J.*, vol. 7, no. 3, pp. 1745–1755, Oct. 2019.

[29] G. Li, A. Hu, J. Zhang, L. Peng, C. Sun, and D. Cao, "High-Agreement Uncorrelated Secret Key Generation Based on Principal Component Analysis Preprocessing," *IEEE Trans Commun*, vol. 66, no. 7, pp. 3022–3034, Jul. 2018.

[30] G. Li, Z. Zhang, J. Zhang, and A. Hu, "Encrypting Wireless Communications on the Fly Using One-Time Pad and Key Generation," *IEEE Internet of Things J.*, vol. 8, no. 1, pp. 357–369, Jun. 2021.

[31] J. Zhang, A. Marshall, R. Woods, and T. Q. Duong, "Efficient key generation by exploiting randomness from channel responses of individual OFDM subcarriers," *IEEE Trans Commun*, vol. 64, no. 6, pp. 2578–2588, Apr. 2016.

[32] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, "Deep convolutional neural networks for massive MIMO fingerprint-based

positioning," in *Proc. IEEE Int Symp Person Indoor Mobile Radio Commun. (PIMRC)*, Montreal, QC, Canada, Oct. 2017, pp. 1–6.
[33] K. Hornik, M. Stinchcombe, H. White *et al.*, "Multilayer feedforward networks are universal approximators," *Neural Netw*, vol. 2, no. 5, pp. 359–366, 1989.
[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: http://arxiv.org/abs/1412.6980
[35] C. Zenger, J. Zimmer, and C. Paar, "Security Analysis of Quantization Schemes for Channel-Based Key Extraction," in *Proc. Workshop Wireless Commun. Security Phys. Layer*, Coimbra, Portugal, Jul. 2015, pp. 267–272.
[36] X. Zhu, F. Xu, E. Novak, C. C. Tan, Q. Li, and G. Chen, "Extracting secret key from wireless link dynamics in vehicular environments," in *Proc. IEEE INFOCOM*, Turin, Italy, Apr. 2013, pp. 2283–2291.
[37] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Proc. Int'l Conf. Theory and Applications of Cryptographic Techniques*, May 2004, pp. 523–540.
[38] A. Alkhateeb, "DeepMIMO: A Generic Deep Learning Dataset for Millimeter Wave and Massive MIMO Applications," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2019, pp. 1–8.
[39] "Remcom wireless insite." [Online]. Available: http://www.remcom.com/wireless-insite
[40] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Booz-allen and hamilton inc mclean va, Tech. Rep., 2001.
[41] C. Huang, G. C. Alexandropoulos, A. Zappone, C. Yuen, and M. Debbah, "Deep learning for UL/DL channel calibration in generic massive MIMO systems," in *Proc. IEEE Int Conf Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.
[42] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," 2017. [Online]. Available: http://arxiv.org/abs/1611.06440

Xinwei Zhang (Student Member, IEEE) received the B.S. degree in computer science and technology from Henan University of Technology, Zhengzhou, China, in 2018. He is currently pursuing the M.S. degree in computer technology with Southeast University. His research interests include physical-layer security, and secret key generation.

Guyue Li (Member, IEEE) received the B.S. degree in information science and technology and the Ph.D. degree in information security from Southeast University, Nanjing, China, in 2011 and 2017, respectively.

From June 2014 to August 2014, she was a Visiting Student with the Department of Electrical Engineering, Tampere University of Technology, Finland. She is currently an Associate Professor with the School of Cyber Science and Engineering, Southeast University. Her research interests include physical-layer security, secret key generation, radio frequency fingerprint, and link signature.

Junqing Zhang received the B.Eng and M.Eng degrees in Electrical Engineering from Tianjin University, China in 2009 and 2012, respectively, and the Ph.D degree in Electronics and Electrical Engineering from Queen's University Belfast, UK in 2016. From Feb. 2016 to Jan. 2018, he was a Postdoctoral Research Fellow with Queen's University Belfast. From Feb. 2018 to May 2020, he was a Tenure Track Fellow (Assistant Professor) with University of Liverpool, UK. Since June 2020, he is a Lecturer (Assistant Professor) with University of Liverpool. His research interests include Internet of Things, wireless security, physical layer security, key generation, and radio frequency fingerprint identification.

Aiqun Hu (Senior Member, IEEE) received the B.Sc.(Eng.), M.Eng.Sc., and Ph.D. degrees from Southeast University in 1987, 1990, and 1993, respectively.

He was invited as a Post-Doctoral Research Fellow with The University of Hong Kong from 1997 to 1998, and a TCT Fellow with Nanyang Technological University in 2006. He has published two books and more than 100 technical articles in wireless communications field. His research interests include data transmission and secure communication technology.

Zongyue Hou received the B.S. degree in communication engineering from the Jiangnan University, Wuxi, China, in 2020. He is currently pursuing the M.S. degree in electronic information with Southeast University. His research interests include physical-layer security, and secret key generation.

Bin Xiao (S'01-M'04-SM'11) received the B.Sc. and M.Sc. degrees in electronics engineering from Fudan University, China, and the Ph.D. degree in computer science from The University of Texas at Dallas, USA. He is currently a Professor with the Department of Computing, The Hong Kong Polytechnic University. Dr. Xiao has over ten years research experience in the cyber security, and currently focuses on the blockchain technology and AI security. He published more than 100 technical papers in international top journals and conferences. Currently, he is the associate editor of the Journal of Parallel and Distributed Computing (JPDC) and the vice chair of IEEE ComSoc CISTC committee. He has been the symposium co-chair of IEEE ICC2020, ICC 2018 and Globecom 2017, and the general chair of IEEE SECON 2018.