2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC) | 978-1-6654-3161-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CCNC49033.2022.9700617

NN-key: A Neural Network-Based Secret Key for Demapping OFDM Symbols

Nasim Soltani, Yanyu Li, Deniz Erdogmus, Yanzhi Wang, and Kaushik Chowdhury

Department of Electrical and Computer Engineering, Northeastern University
Boston, Massachusetts, USA
Email: {soltani.n, li.yanyu}@northeastern.edu, erdogmus@ece.neu.edu, yanz.wang@northeastern.edu, krc@ece.neu.edu

***Abstract*—Generating custom modulation patterns as well as dynamically varying the mapping of the constellation points to their corresponding bit representations are some existing methods for mitigating eavesdropping attacks. In such cases, the custom symbol to bit mapping needs to be conveyed to the receiver through a secure and reliable channel. Instead of sending the representations of the modified symbols in regular information fields, we propose a machine learning-based approach, in which the modified symbols are encoded in the parameters of a light-weight neural network (NN). This NN is trained at the transmitter-side, sent as a secret key to the receiver, where it serves as a demapping block to recover the received symbols correctly. In addition, this paper explores the role of data augmentation during the training stage to increase the robustness of the NN with respect to the noise in the channel, as well as architecture compression to reduce transmission overhead. We validate the robustness of the proposed NN-based custom-modulation demapping approach by comparing it with demapping of a standard scheme (e.g., 16QAM), which reveals no appreciable loss in performance. We further quantitatively analyze the impact of channel and noise impairments on the demapping performance.**

***Index Terms*—secure transmission, eavesdropper, demapper neural network, neural network compression, quantization**

I. Introduction

Resource-constrained Internet of Things (IoT) devices benefit from physical layer security, as this lowers the software processing demands at the upper layers of the protocol stack. Recently, physical layer security methods involving emitter identification (i.e., RF fingerprinting) [1] have proven successful in detecting the presence of unauthorized devices through characteristic features in the transmitted signal [2]. However, they do not ensure confidentiality of information and protection against eavesdroppers [3]. The latter consideration is a critical need in the IoT paradigm, as devices are required to frequently exchange data and control signals with other sensors and access points.

• **Protection Against Eavesdroppers.** One possible approach to achieve confidentiality of transmitted information is through making the *demapping* operation difficult for eavesdroppers, while ensuring that the legitimate pair of devices retain their effective communication [4]. Generally, this is accomplished by generating a *secret key* [5]–[7]. Such a secret key, for example, can be generated based on the channel state information (CSI). The secret key is, then, communicated through a reliable and secure back channel from the receiver to the transmitter, which in turn, is at the risk of CSI-leakage. To this date, a number of secret key generation methods have been proposed and incorporated into different wireless technologies and protocols, such as orthogonal frequency division multiplexing (OFDM) [8], multiple input multiple output (MIMO) [5], and Bluetooth [9]. However, these existing methods are constrained by the modulation schemes within the standard, and may not be able to properly adapt to the channel quality within the limited configuration options provided by the standard. For example, in environments with varying noise levels, a higher order modulation scheme such as 64QAM will likely result in a high bit error rate (BER), while 16QAM may not provide sufficient data rate. Ideally, using a modulation scheme between them, such as 32QAM, may be desirable to suitably tradeoff the BER and data rate. However, such intermediate constellation sizes may not be part of the standard (e.g., 32QAM is not part of the `NonHT` format used in WiFi, from 802.11a to 802.11ac), which hinders the ability of the

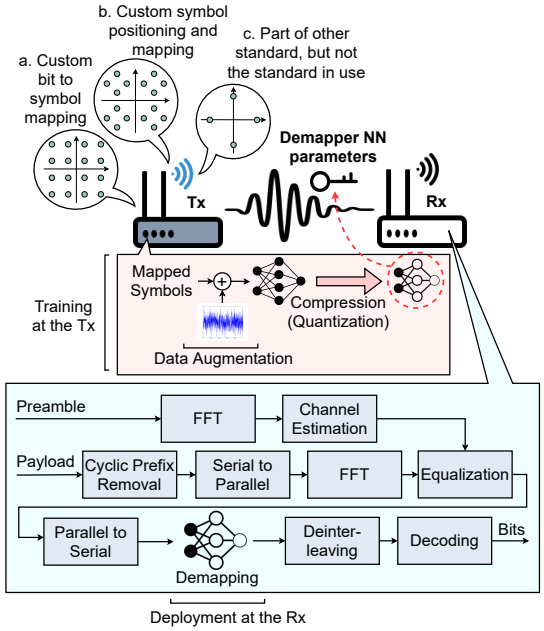

Fig. 1: Overview of NN-key system. The NN-based key corresponding to the selected modulation scheme is trained and compressed at the transmitter. The compressed key is sent to the receiver to be used as the demapping processing block.

communicating devices to properly adapt to bad links.

• **Proposed NN-key System.** To address the above limitations, we propose a secure transmission scheme that is (i) generalizable to all OFDM wireless systems, (ii) does not include complex signal processing in decryption stage, and (iii) does not require a secure back channel to convey CSI from the receiver to the transmitter. Our system is able to incorporate adaptive modulation algorithms, since it can provide custom modulation schemes with optional orders, irrespective of whether or not they are defined as part of the wireless standard. In our proposed approach, the transmitter varies the standard default bit to symbol mapping and/or the standard symbol positioning, and for that modified modulation scheme, trains a light-weight neural network (NN) as the demapper. Along the training phase, data augmentation is used to make the NN robust for demapping noisy samples, and quantization is used to compress NN parameters (weights and biases). The augmented and compressed trained NN is sent to the receiver as a demapping key, which replaces the demapper block in the receiver processing chain to demap up-coming data packets. To maintain security and data rate adaptability, the transmitter hops among custom modulation schemes with different orders and mappings, and trains separate keys for new schemes it chooses. The new trained key is sent to the receiver to update the demapper NN parameters, so that the receiver can adapt to the new modulation scheme selected at the transmitter-side. The overview of our proposed system is shown in Fig. 1.

Although NN-key encodes the secret key in the parameters of an NN, the modulation selection process at the transmitter-side happens through conventional, non-machine-learning adaptive methods. We further note that in NN-key, even though we change the standard symbol positioning in the constellation space to design new custom modulation schemes for secure transmission, our approach is different from steganography [10]. In steganography, the excessive channel capacity is used to implement a covert channel to stream secret data along with the overt data. However, NN-key *generates* a secret key that secures the whole overt data.

• **Contributions:** Our contributions are as follows:

- We propose an NN-based demapping key system for securing the communications between the transmitter and legitimate receivers. The NN architecture is saved on the legitimate receiver before the start of transmission, and during transmission, only the NN parameters are dynamically transmitted, which makes our scheme resilient to key-leakage. Moreover, the packet structure remains the same as the standards-compliant packet, which foils the eavesdropper's attempt to decode them through the standard chain.
- We propose a light-weight NN architecture for demapping OFDM symbols in wireless receivers. Avoiding complicated and compute-intensive encryption and decryption methods, our demapper NN requires simple mathematical operations (only addition and multiplication) and small computational overhead, since it has a small number of parameters.
- We propose a data augmentation step within the training pipeline that makes it possible for an NN trained on clean transmitter-side symbols, to demap noisy symbols at the receiver-side.
- We propose quantization to limit the parameters (weights and biases) of the NN to a finite and small set of values, reduce the parameter size, and hence, the length of the key. We show that quantization reduces the overhead of the proposed secure transmission system by 81%.
- We study the key corruption as a result of imperfect communication. We assess the demapping performance with a demapper NN with parameters corrupted with different rates. We show that 1 bit error in key communication, causes negligible 3% increase in data demapping BER.

The rest of the paper is organized as follows. Section II describes the conventional demapping task, its drawbacks for secure transmission, and proposes an NN for demapping custom or standard OFDM modulation schemes. Section III describes the secure transmission system with an NN-based demapper secret key. Section IV presents the numerical evaluations, and Section V concludes the paper.

II. Demapping Operation in OFDM Receivers

In this section, we study the process of conventional demapping and its limitations for securing data transfer with custom modulation schemes. Then we propose an NN-based demapping method that replaces the conventional demapping block in the receiver processing chain. We limit our examples and implementations to WiFi OFDM systems, given OFDM is an integral part of most modern standards.

A. Conventional Demapping

In an OFDM receiver, the demapper block is located after the equalizer as shown in Fig. 1 (bottom). The demapper takes the equalized symbols, Xs, with modulation order M, and for each symbol X, it generates a sequence of $k = \log_2 M$ soft-bits. The soft-bits (a.k.a. log likelyhood ratios (LLRs)), are next deinterleaved and handed over to the error correction block that decodes the data, and finally generates the received bits. The conventional deterministic demapper, although functional for modulation schemes within the standard, lacks flexibility for demapping custom modulation schemes with custom symbol positioning and/or custom bit to symbol mapping.

B. NN-Based Demapping

The task of demapping at the receiver can be done more flexibly using an NN, that learns the position of the equalized symbols in the constellation space, and associates them with specific bit sequences. In this process, each equalized symbol is independently demapped to a bit sequence. Hence, our network of choice has dense layers in which, unlike convolutional layers, each input is processed separately. The input layer has dimension 2 with ReLU activation, where I (in-phase) and Q (quadrature) of an equalized symbol are given in parallel as two channels. The output layer has dimension k, which is the number of bits per symbol. Since for each input

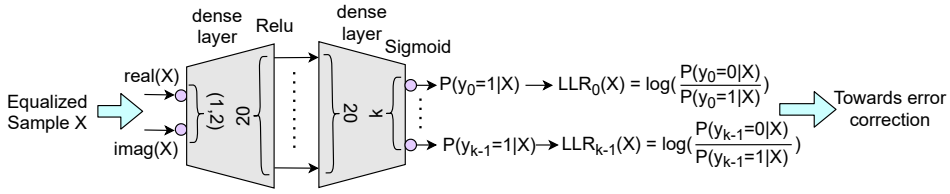

Fig. 2: The proposed demapper neural network architecture and the process of generating LLRs from equalized samples.

symbol X, multiple bits among the k outputs could be '1', we consider demapping as a multi-label classification problem. Consequently, at the output layer, we use Sigmoid activation and binary cross entropy loss function that optimizes each output neuron to take a value of either '0' or '1'. In this case, the output of the network at test time is k probabilities of the corresponding bits, y_is, being '1's. k LLR values, $\text{LLR}_i(X)$, for each input X are calculated as in (1).

$$\text{LLR}_i(X) = \log \frac{P(y_i = 0|X)}{P(y_i = 1|X)}, \quad i = 0, \ldots, k-1 \qquad (1)$$

In (1), $P(y_i = 0|X)$ is the probability of bit i, denoted as y_i, being '0'.

To find the smallest demapper NN, we search the design space thoroughly in terms of number of hidden layers and number of neurons in each layer. We finalize the architecture as a small dense network with two layers of sizes 20 and k, as shown in Fig. 2. The total number of parameters in the demapper network varies with the modulation order M, and is calculated as $60 + 20k + k$, where $k = \log_2 M$.

III. Secure Transmission Using the NN-Based Demapping Key

A. System Overview

We propose a secure and adaptable transmission system using our NN-based demapper. In this system, the transmitter chooses a custom digital modulation scheme with desired order, and specific bit to symbol mapping, and trains an NN to demap symbols with that specific scheme. Data augmentation is applied to enable the NN to demap noisy symbols at the receiver-side, while it is trained on the noiseless symbols at the transmitter-side. The NN is also compressed through quantization during training. The quantized demapper is sent to the receiver as a demapping key, in an initial packet prior to data transmission. The receiver, receives this key and uses it in the processing chain after the channel equalizer, to translate the received symbols to soft-bits. This overview is shown in Fig. 1. While the eavesdropper receiver demodulates and demaps the data packets incorrectly using the conventional demapper, the legitimate receiver can demap the symbols correctly, using the NN architecture saved on it before run-time, and the NN-based demapping key.

B. Packet Type and Structure

In our secure transmission system, there are two types of packets sent to the receiver: 1) The packets containing the secret key (NN-key packets), 2) The data packets. Both these packets have a structure compliant to standard formats.

Without losing generality, and for a concrete implementation case, we use the packet structure compliant to `Non-HT`, `HT`, and `VHT` formats used in WiFi 802.11a through 802.11ac. In the preamble of these formats, the standard reserves a few unused bits for future use. For example, bit 4 of the `L-SIG` field, which is part of the preamble in `Non-HT`, `HT`, and `VHT` formats, is unused. We use this bit as an *NN-flag* to determine the packet type as described below:

1) **The NN-key packet**: If the NN-flag is '1', the received packet is considered to contain NN parameters, and is called the NN-key packet. The NN-key packet is always modulated with a standard modulation scheme that is demapped through the standard conventional receiver chain. In the payload of the NN-key packet, the first 4 bits show the binary representation of k-1, where k is the number of mapped bits per symbol. k determines the output layer size of the NN trained for the custom modulation scheme. The rest of the bits in the NN-key packet represent NN parameters.
2) **The data packet**: If the NN-flag is '0', the received packet is considered to contain data. The data packets are modulated with a custom scheme chosen by the transmitter, and can only be correctly demapped using the NN-key.

C. Training the NN-Key

As explained in Section III-A, the transmitter needs to train a new secret key, for each modulation scheme that it selects. The transmitter symbols, used for key training, are clean and noiseless, as seen in Fig. 3a, since they have not yet passed through the wireless channel. If we train the NN directly with the clean transmitter constellations, the NN cannot well demap the distorted and noisy equalized samples at the receiver. To boost the demapping performance on the noisy symbols, we use data augmentation [11]. This technique make NNs robust to variations in the test set by applying similar variations to the training set. As shown in Fig. 1 (bottom), the major distortion remaining at the input of the demapper is the *additive noise*, since the channel effects are mostly negated by the equalizer. In order to make the NN robust to the additive noise, we use the noise model proposed in [11] to dynamically add noise to the clean constellation, during training. When a certain modulation scheme with order M is chosen by the transmitter, to construct the training set, we need only M different symbols and their corresponding bits, acquired at the output and input of the transmitter *mapper* block, respectively. For creating the augmented data, In each epoch, the M complex noiseless

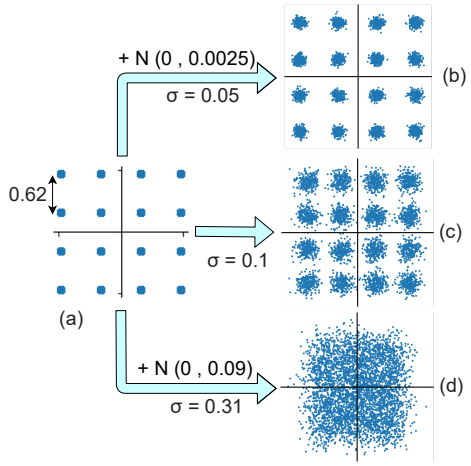

Fig. 3: The effect of adding noise drawn from Gaussian distribution $\left(\mathrm{N}(\mu{=}0\ ,\ \sigma^2)\right)$ to the noiseless transmitter symbols in data augmentation step.

symbols are copied P times, to build a training batch of size $(M \times P,\ 2)$, where I and Q come in the last dimension. Next, a batch of values with the same dimensions of $(M \times P,\ 2)$ is drawn from complex Gaussian distribution with mean $\mu = 0$ and a certain standard deviation (σ) (i.e., N(μ,σ)), which is summed with the noiseless symbol batch to yield the augmented batch [11].

An example of the transmitter constellation being augmented with noise with different σs is shown in Fig. 3. The optimal σ for data augmentation depends on the symbol positioning in the clean constellation, and is chosen in a way to make the augmented constellation clusters touch, but not overlap. This σ is achieved by dividing the minimum distance between the transmitter constellation points by two. An NN trained with constellation augmented with optimum σ yields minimum BER, when inserted in the receiver chain. For a 16QAM-like scheme shown as scheme *a* in Fig. 1, the optimum σ value for augmentation equals 0.31, that is shown in Fig. 3d.

D. Compressing the NN-Key

The NN-key packets (containing the secret key) create overhead for the proposed secure transmission system. Besides using the smallest NN possible, compression techniques can further reduce the overhead of the system. We use quantization as a compression method to reduce the bit-width of each NN parameter, and therefore, reduce the length of the payload in the NN-key packet. Quantization happens during training, where the weights and biases are limited to a finite set of values in the range of [0,1] with equal spacing, along with their negatives. Quantizing the parameters to L bits creates a set, J, of $2^L - 1$ values as shown in (2).

$$J = \bigcup_{j=-(2^{L-1}-1)}^{2^{L-1}-1} \frac{j}{2^{L-1}-1} \tag{2}$$

For example, quantizing the parameters to bit-width of $L = 4$ limits the weights and biases to be chosen from the $(2^4{-}1 =)$ 15-member set of $J = \{-1, -\frac{6}{7}, -\frac{5}{7}, \ldots, 0, \ldots, \frac{5}{7}, \frac{6}{7}, 1\}$. After training, these 15 values are scaled up to be bit-represented as 0000 to 1110 for the 4-bit quantization example. The bit-width for quantization is chosen as the smallest width that does not hurt the demapper NN performance.

E. Reliable Transmission of the NN-Key

The reliable transmission of the NN-key packet is critical for the correct demapping of data packets. Therefore, we use BPSK to modulate the NN parameters in the payload of the NN-key packet, as this low order modulation scheme is very resilient to noise.

Apart from BPSK, the NN-key packet can be also modulated with higher order standard modulation schemes, such as QPSK, 16QAM, etc. Using higher order modulation schemes shortens the NN-key packet payload, and reduces the system overhead. However, the higher order schemes could ensure correct data delivery only in the least noisy environments. Any bit errors in the NN-key packet could compromise the performance of the demapper key. In Section IV, we analyze and experimentally quantify the error rate that the NN-key packet can tolerate to retain good demapping performance.

As a numerical example, based on the explanations in Section II-B, the total number of parameters in the demapper NN that demaps a custom modulation of order 32, equals 165. If we select the width of 4 bits to quantize the demapper, and add 4 bits to the beginning of NN-key payload that represent the output layer size, the length of the information sequence will be 664 bits. Assuming a coding rate of 1/2, the coded payload will have 1328 bits. This payload length could be 1328, 664, or 332 symbols (without cyclic prefix and OFDM pilots), if BPSK, QPSK, or 16QAM is used to modulate the NN-key packet.

F. Demapping at the Receiver Using the NN-Key

In the proposed system when the receiver receives a packet, after synchronization and channel estimation, it examines the NN-flag to determine the packet type, as explained in Section III-B.

If the received packet is an NN-key packet, first, the receiver passes the payload all through the conventional standard processing blocks, including the conventional demapper, and generates the received bits. The acquired bits are the bit representation of NN parameters, as well as the number of neurons in the last layer, shown in the first 4 bits of the payload, as explained in Section III-B. Second, the receiver configures the NN with the architecture shown in Fig. 2, with the output layer size, k, acquired from the first 4 bits of the payload. Third, by calculating the total number of parameters using k (see Section II-B) the parameters are separated and inserted in the empty architecture. This architecture replaces the demapper block in the standard conventional receiver chain. From this point on, the demapper key is ready, and the future data packets are demapped using this key until a new

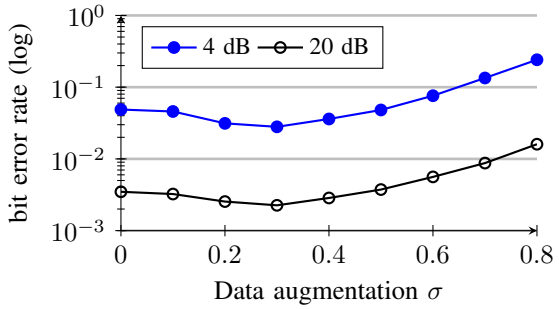

Fig. 4: Changes in BER with the demapper neural networks when different σ values are used for data augmentation at the training phase. The best BER is achieved when σ=0.31.

NN-key packet arrives. The reception of a new NN-key packet means the transmitter will use a new modulation scheme for the next data packets, and hence the demapper NN should be updated.

If the received packet is a data packet, the payload is pushed through the receiver processing chain, with the demapper NN in it, as shown in Fig. 1 (bottom).

IV. Evaluations

In this section, we provide receiver BER as a metric for the performance of the proposed demapper when: (*A*) data augmentation σ is varied, (*B*) custom modulation schemes are used, (*C*) the proposed demapper is quantized to different bit-widths, and (*D*) the quantized parameters experience bit errors during key communication.

To simulate the wireless medium, we use `TGn` channel model from MATLAB `Communications` toolbox that simulates an indoor office environment, cascaded with additive white Gaussian noise (AWGN) channel with SNRs chosen from 0 to 30 dB in steps of 2 dB. The demapper key tests are done with the correct demapper NN inserted in the receiver processing chain, as shown in Fig. 1 (bottom), and the LLRs are calculated as shown in Fig. 2. The final BER is calculated after the conventional decoder in the receiver chain.

A. Data Augmentation Standard Deviation (σ)

To study how different standard deviations (σs) in data augmentation at the transmitter affect the demapping performance at the receiver, we train 9 demappers with transmitter symbols modulated with a 16QAM-like custom schemes with standard symbol positioning, but custom symbol to bit mapping (scheme *a* in Fig. 1), using 9 σs in the set $\{0, 0.1, 0.2, \ldots, 0.7, 0.8\}$. We test the trained demappers individually in the receiver chain under SNRs 4 dB and 20 dB, as examples of low and high SNRs, respectively. As shown in Fig. 4, regardless of the test SNR level, the best performance is achieved when σ equals 0.31. This matches our expectation that the best demapper performance is achieved when σ equals half the shortest distance between transmitter constellation points. For both SNR levels, with increasing the σ beyond 0.31, the BER increases upto 0.5.

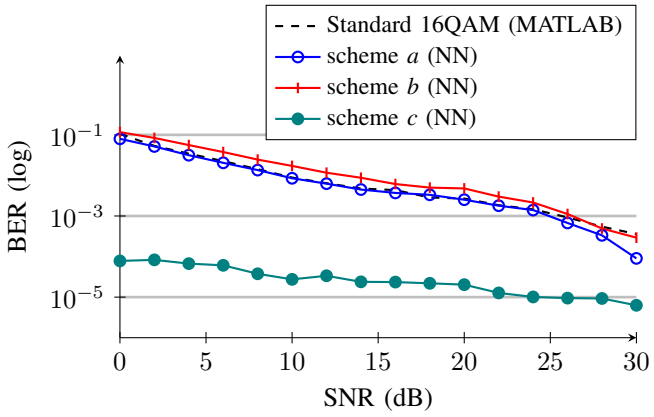

Fig. 5: Performance comparison between MATLAB demapper on standard 16QAM, and demapper NN on different modulations schemes shown in Fig. 1. Scheme *a* has custom bit to symbol mapping, scheme *b* has custom symbol positioning and custom bit to symbol mapping, and scheme *c* is out of the standard in use, but potentially part of other standards. The neural network is trained with proper σ value selected as 0.31, 0.22, and 0.66 for scheme *a*, *b*, and *c*, respectively.

B. Demapping From Custom Modulation Schemes

To show how the proposed demapper NN performs in demapping signals with custom symbol positioning or custom bit to symbol mapping, we modify the standard WiFi transmitter chain in MATLAB and generate packets with 3 custom mapping schemes of *a*, *b*, and *c*, illustrated in Fig. 1. Among these modulation schemes, scheme *a* has custom symbol to bit mapping, scheme *b* has custom symbol positioning and mapping, and scheme *c* is a custom scheme that is potentially part of other standards, but not the standard in use. We train separate NNs with the architecture shown in Fig. 2 to demap these schemes. The shortest distance between two neighbouring symbols in the transmitter constellation of schemes *a*, *b*, and *c* equals 0.62, 0.44, and 1.32, respectively, and therefore the best augmentation σs of 0.31, 0.22, and 0.66 (half the distances) are used to train each network. We assume no quantization and perfect transmission of the NN parameters to the receiver, which means the parameters are not corrupted. We test these 3 demappers individually in the receiver chain using receiver packets with corresponding modulation schemes under SNRs 0 to 30 dB. We further create a 16QAM baseline by passing signals modulated with standard 16QAM, through a receiver chain with conventional MATLAB blocks, without NNs. The final BER for custom schemes of *a*, *b*, and *c* along with the MATLAB conventional BER for standard 16QAM is illustrated in Fig. 5. Among these custom schemes, scheme *a* is of order 16 with the same symbol positioning as the standard 16QAM, and therefore, BERs are in the same range as those of 16QAM. As expected, BER for scheme *b* is larger than scheme *a*, since the symbols are closer to each other in scheme *b*. Expectedly, custom scheme *c* shows the lowest BER since it has the smallest order (4 compared to 16 for others), and hence, the largest distance between constellation points.

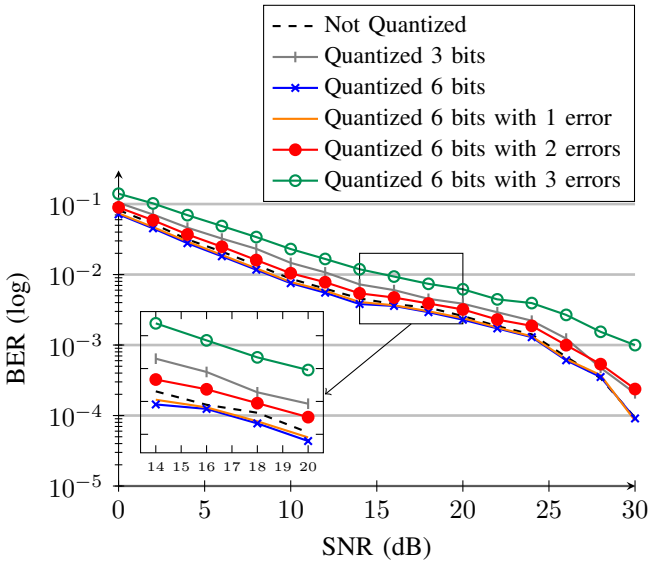

Fig. 6: Demapper neural network performance when its parameters are quantized, and when the quantized parameters are subject to 1, 2, or 3 bit errors.

C. Compressing (Quantizing) the NN-Key

Without compression, the NN parameters have 32-bit floating-point representation, which corresponds to 4608 bits for a total of 144 parameters. We use quantization to reduce the parameters bit-widths, and hence, the length of NN-key packet. We quantize the demapper with different widths and measure the BER degradation in each case. As an example, as shown in Fig. 6, 3-bit quantization degrades the performance of the NN by 76% on average. We find out that the shortest bit-width that does not degrade the BER is 6. The 6-bit quantization reduces NN-keys packet length, and hence the system overhead, by 81% compared to the non-quantized case.

D. Transmitting the NN-Key in Noisy Environments

In order to study the impact of bit errors in the demapper parameters on the demapper performance, we corrupt the 864 bits representing the 6-bit-quantized demapper key for the 16QAM-like modulation (scheme *a* shown in Fig. 1), with different rates. In separate experiments, we flip 1, 2, and 3 bits in the NN-key packet to emulate bit corruption with rates of 0.11%, 0.23%, and 0.34%, respectively. We insert the corrupted demapper back in the receiver chain, and test it under SNRs 0 to 30 dB. We do the experiment for each error rate 10 times, to account for the random positions where the errors appear, and average the performance results. We observe that the 6-bit-quantized NN-key packet can tolerate 1 bit error (0.11% BER in the NN-key packet) with only a 3% increase in the demapper data BER. However, 2 and 3 bit errors in the NN-key packet impose 46% and 241% increase in the demapper BER, respectively.

V. Conclusion

In this paper, we proposed a secure and adaptable communication system that uses an NN-based key to demap OFDM symbols. In our system, the transmitter hops among custom modulation schemes with different symbol positioning, different bit to symbol mapping, and different orders to accommodate the security and adaptability requirements. The transmitter trains a key with quantized parameters for each scheme, and boosts its performance through data augmentation during training. The trained NN parameters are sent over the air to the legitimate receiver as a demapping key, which the latter uses to demap future data packets. Besides incurring low overhead, one of the advantages of our scheme is the compliance with standard-defined packet structures, which makes our packets indistinguishable from regular packets to the eavesdroppers. We further analyzed the potential imperfect delivery of the demapping key, and measured the tolerance of the proposed demapper against errors in its parameters.

Acknowledgement

This work is supported by DARPA SPiNN HR00112090055 and NSF CNS 1923789.

References

[1] N. Soltani, G. Reus-Muns, B. Salehihikouei, J. Dy, S. Ioannidis, and K. Chowdhury, "RF Fingerprinting Unmanned Aerial Vehicles with Non-standard Transmitter Waveforms," *IEEE Transactions on Vehicular Technology*, 2020.

[2] G. Shen, J. Zhang, A. Marshall, and J. Cavallaro, "Towards Scalable and Channel-Robust Radio Frequency Fingerprint Identification for LoRa," *arXiv preprint arXiv:2107.02867*, 2021.

[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 Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1550–1573, 2014.

[4] E. Güvenkaya, J. M. Hamamreh, and H. Arslan, "On physical-layer concepts and metrics in secure signal transmission," *Physical Communication*, vol. 25, pp. 14–25, 2017.

[5] Y. Yang, M. Ma, S. Aïssa, and L. Hanzo, "Physical-Layer Secret Key Generation via CQI-Mapped Spatial Modulation in Multi-Hop Wiretap Ad-Hoc Networks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1322–1334, 2020.

[6] H. M. Furqan, J. M. Hamamreh, and H. Arslan, "Secure and Reliable IoT Communications Using Nonorthogonal Signals' Superposition with Dual-Transmission," in *IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1–6, IEEE, 2020.

[7] J. Zhang, A. Marshall, R. Woods, and T. Q. Duong, "Design of an OFDM physical layer encryption scheme," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2114–2127, 2016.

[8] Y. Lee, H. Jo, Y. Ko, and J. Choi, "Secure Index and Data Symbol Modulation for OFDM-IM," *IEEE access*, vol. 5, pp. 24959–24974, 2017.

[9] K. Li, N. Lu, J. Zheng, P. Zhang, W. Ni, and E. Tovar, "A Practical Secret Key Management for Multihop Drone Relay Systems based on Bluetooth Low Energy," in *2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–2, IEEE, 2021.

[10] L. Bonati, S. D'Oro, F. Restuccia, S. Basagni, and T. Melodia, "Stealte: Private 5g cellular connectivity as a service with full-stack wireless steganography," *arXiv preprint arXiv:2102.05606*, 2021.

[11] N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More Is Better: Data Augmentation for Channel-Resilient RF Fingerprinting," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 66–72, 2020.