2020 29th Wireless and Optical Communications Conference (WOCC)

Reservoir Computing Meets Wi-Fi in Software Radios: Neural Network-based Symbol Detection using Training Sequences and Pilots

Lianjun Li[1], Lingjia Liu[1], Jianzhong (Charlie) Zhang[2], Jonathan D. Ashdown[3], and Yang Yi[1]
[1]Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA
[2]Standards and Mobility Innovation Lab., Samsung Research America, Plano, TX, USA
[3]Information Directorate, U.S. Air Force Research Lab., Rome, NY, USA

***Abstract*—In this paper, we introduce a neural network (NN)-based symbol detection scheme for Wi-Fi systems and its associated hardware implementation in software radios. To be specific, reservoir computing (RC), a special type of recurrent neural network (RNN), is adopted to conduct the task of symbol detection for Wi-Fi receivers. Instead of introducing extra training overhead/set to facilitate the RC-based symbol detection, a new training framework is introduced to take advantage of the signal structure in existing Wi-Fi protocols (e.g., *IEEE* 802.11 standards), that is, the introduced RC-based symbol detector will utilize the inherent long/short training sequences and structured pilots sent by the Wi-Fi transmitter to conduct online learning of the transmit symbols. In other words, our introduced NN-based symbol detector does not require any additional training sets compared to existing Wi-Fi systems. The introduced RC-based Wi-Fi symbol detector is implemented on the software defined radio (SDR) platform to further provide realistic and meaningful performance comparison against the traditional Wi-Fi receiver. Over the air experiment results show that the introduced RC-based Wi-Fi symbol detector outperforms conventional Wi-Fi symbol detection methods in various environments indicating the significance and the relevance of our work.**

***Index Terms*—Wi-Fi, symbol detection, neural network, reservoir computing, USRP, and limited training.**

I. Introduction

Wi-Fi technologies [1] are important in modern wireless systems to provide low-cost/high-throughput reliable connections for wireless local area networks (WLAN). Modern Wi-Fi systems adopt orthogonal frequency division multiplexing (OFDM) as their physical layer multi-access strategy. Transmit symbol detection is one of the most important modules in the receive processing chain of Wi-Fi systems to recover the transmitted signal from received signals. Conventional model-based symbol detection methods usually estimate the underlying wireless channel first. Based on the estimated channel and the received signal, the receiver will conduct various signal processing techniques to recover the transmitted signal. A major drawback of this approach is that the underlying signal processing techniques need to incorporate the underlying model information of the input-output relationship between the transmitted signal and the received signal. However, the exact model of the transceiver chain is difficult to obtain due to the presence of radio frequency (RF) non-linearities, either from the hardware (e.g. power amplifier) or the wireless channel. Therefore, traditional model-based schemes suffer from model mismatch and channel estimation errors calling for more robust and adaptive methods.

Neural networks (NNs) provide learning-based methods that do not need an explicit model of the underlying problem providing an ideal approach to resolve the issue of model mismatch in communication systems. Recently, NN-based approaches have been applied in many communication problems such as channel estimation, symbol detection, resource allocation, and fault detection [2], [3]. For example, in [4] a 3-hidden-layer multilayer perceptron (MLP) is designed for OFDM symbol detection without explicitly estimating the underlying wireless channel. An extreme learning machine-based symbol detection method is introduced in [5] for OFDM system against fading channel and power amplifier nonlinear distortion, simulation results show it outperforms the LMMSE method. Meanwhile, [6] introduces model-based and data-driven NNs for OFDM receiver, performance of those two designs are evaluated by simulation as well as the over-the-air test. Due to the convolution feature of wireless channel, convolutional neural networks (CNNs) are adopted in [7] for symbol detection achieving better performance than expert OFDM receiver in lower to middle SNR regimes of Rayleigh channels. On the other hand, due to the inherent temporal correlation exhibits in communication signals recurrent neural networks (RNNs) have also been utilized in many studies of communication systems [8], [9]. However, it is important to note that all these learning-based algorithms require extremely large number of training samples to achieve good performance, making it almost impossible to implement these schemes in practical and relevant communication systems.

In wireless communication systems, control and training overhead are extremely costly which negatively impact the underlying system throughput [2]. This is completely different from other fields such as the computer vision where tens of thousands of training sets are available. Therefore, introducing NN-based approaches with reduced training overhead is of

The work of L. Li, L. Liu, and Y. Yi are partially supported by U.S. National Science Foundation (NSF) under grants ECCS-1811497 and CCF-1937487. Approved for public release (reference number: 88ABW-2020-1532). The corresponding author is L. Liu (ljliu@ieee.org).

crucial importance to realize the promise of learning-based communication strategies in realistic and relevant scenarios. Ideally, we want to invent communication strategies that are having the same or less training/control overhead compared to traditional model-based signal processing techniques. To realize this objective, reservoir computing (RC), which is a special category of RNNs that can significantly reduce the training overhead by only optimizing the output layer weights, has been introduced in our previous work [10]–[12] to conduct symbol detection for OFDM systems with very limited training overhead. Echo state network (ESN), which is one type of RC, is adopted in [10] to show promising performance with very limited training overhead. The following work in [11] introduced the windowed ESN (WESN) to improve ESN-based strategy by adding a sliding window to its input. In [12], a deep NN-based DeepRC is introduced to further improve the system performance by taking advantage of the structural information of OFDM signals.

In this paper, instead of focusing on generic OFDM signals we will apply RC-based symbol detection to Wi-Fi systems. To be specific, we are going to completely relying on the existing system overhead (long and short training sequences, comb pilot symbols defined in current Wi-Fi protocols) to conduct online training of the RC-based symbol detector. To provide relevant and convincing comparison, we will implement our scheme on software defined radio (SDR) platform for performance evaluation. To our best knowledge, this is the first learning-based scheme that can be directly adopted in Wi-Fi systems in a plug-and-play manner without any pre-training procedure and additional training overhead.

II. Wi-Fi Physical Layer

A. Wi-Fi Transceiver Procedure

The family of *IEEE* 802.11 standards specify the media access control (MAC) and physical layer (PHY) protocols of Wi-Fi technologies [1]. Within the family, 802.11a/g are the most widely adopted technologies in modern wireless systems, which are also the main focus of this paper. In the rest of this section, we briefly introduce the PHY procedure of 802.11a/g transceiver, as shown in Fig. 1.

In the PHY, OFDM is adopted by 802.11a/g as the multi-access strategy. The frame structure of a Wi-Fi frame can be illustrated in Fig. 2. On the transmitter side, payload message from the MAC layer is subsequently processed as following:

- **OFDM data generation and carrier allocation:** First, payload information bits are encoded and mapped to quadrature amplitude modulation (QAM) symbols where QAM symbols are allocated to OFDM sub-carriers forming OFDM symbols. Specifically, 64 sub-carriers are used in one OFDM symbol, as shown in Fig. 3, where 48 of them carry data; 4 of them are placed with known pilot symbols, which are supposed to be utilized by receiver for channel estimation and symbol detection; the rest 12 are null sub-carriers. All OFDM **data** symbols (in the frequency domain) are generated in this step where the exact number of generated OFDM data symbols depends on the length of payload as well as the modulation and coding scheme (MCS) selected.
- **Signal field:** 1 OFDM **signal** symbol is added as the prefix to the OFDM data symbols. It informs the receiver about the length and MCS information of the following OFDM data symbols. The signal symbol itself is modulated by binary phase shift keying (BPSK) and encoded by rate $1/2$ code.
- **Training sequences:** Short training sequences (STS) and long training sequences (LTS), each with the length of two OFDM symbols, are added on top of the OFDM **signal** symbol. STS and LTS are known symbols aiming to facilitate frame detection and symbol synchronization as well as coarse channel estimation on receiver side.

Once the Wi-Fi frame is generated in the frequency domain, an Inverse Fast Fourier Transform (IFFT) of size 64 is performed to convert it to time domain. A cyclic prefix (CP) of length 16 is prepended to each OFDM symbol. On the receiver side, the PHY procedure is the following:

- **Frame detection and symbol synchronization:** For detecting the start of a frame, receiver correlates the received samples with the STS, once the correlation exceeds a certain threshold, the frame start can be identified. OFDM symbol synchronization is performed through the LTS. To be specific, received samples are passed through a matched filter (matched to the LTS) where the peak of the filter output is used to locate the start of the OFDM symbol.
- **Remove CP and FFT:** The time domain symbols are converted to the frequency domain by first removing the CP and then applying a FFT with size 64.
- **Symbol detection and demodulation:** Compared with the transmitted symbols, the received ones are distorted by the channel and hardware circuits in the transceiver chain. Symbol detection technique is used to recover the transmitted symbols from the distorted observations. The detected QAM symbols will be mapped back to bits to go through the decoding procedure where the decoded payload will be delivered to the MAC layer.

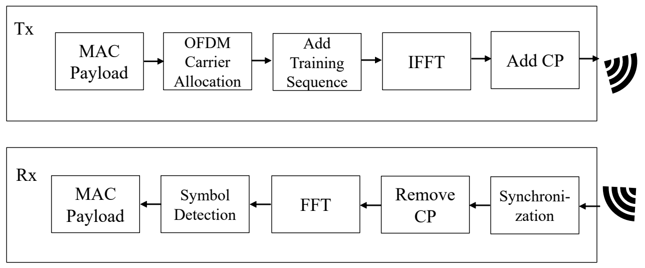

Fig. 1. Wi-Fi transceiver procedure

B. Conventional Symbol Detection Methods in Wi-Fi

In this section, we briefly introduce the symbol detection methods used in Wi-Fi systems. At transmitter, let $\boldsymbol{X_i}$ denotes the ith **frequency** domain OFDM symbol in a frame

$$\boldsymbol{X_i} \triangleq [X_i^0, \cdots, X_i^k, \cdots, X_i^{N_{sc}-1}], \qquad (1)$$

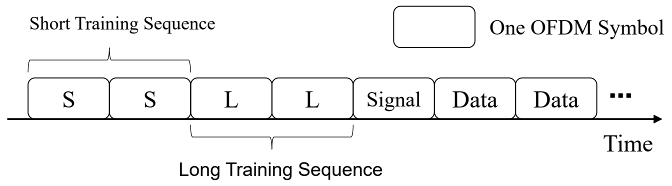

Fig. 2. Wi-Fi frame structure

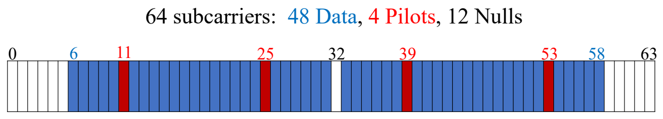

Fig. 3. OFDM carrier allocation

where X_i^k is the QAM symbol at the kth sub-carrier of the ith frequency domain OFDM symbol, $N_{sc} = 64$ is the total number of sub-carriers. Correspondingly, the ith **time** domain OFDM symbol is denoted as $\boldsymbol{x_i}$

$$\boldsymbol{x_i} \triangleq [x_i^0, \cdots, x_i^n, \cdots, x_i^{N_{cp}+N_{sc}-1}], \tag{2}$$

where x_i^n is the nth sample of the ith time domain OFDM symbol, $N_{cp} = 16$ is the CP length. Note $\boldsymbol{X_i}$ can be obtained from $\boldsymbol{x_i}$ by removing the CP and conducting a FFT.

Similarly, at receiver side, let $\boldsymbol{y_i}$ and $\boldsymbol{Y_i}$ denote the ith received **time** and **frequency** domain OFDM symbol in a frame, respectively,

$$\boldsymbol{y_i} \triangleq [y_i^0, \cdots, y_i^n, \cdots, y_i^{N_{cp}+N_{sc}-1}], \tag{3}$$
$$\boldsymbol{Y_i} \triangleq [Y_i^0, \cdots, Y_i^k, \cdots, Y_i^{N_{sc}-1}]. \tag{4}$$

Now we introduce three conventional symbol detection methods adopted in Wi-Fi systems [13]: the Least Squares (LS) method, the Least Mean Square (LMS) method, and the Comb pilots interpolation method (Comb). The LS method utilizes LTS symbols (the 3rd and 4th OFDM symbol in a frame) to estimate channel $\boldsymbol{H}$ in frequency domain, denoting the estimated channel as $\hat{\boldsymbol{H}}$, the channel at kth sub-carrier is estimated as

$$\hat{H}^k = \frac{1}{2}\left(\frac{Y_3^k}{X_3^k} + \frac{Y_4^k}{X_4^k}\right). \tag{5}$$

This estimated channel is then used to recover all the rest received symbols through

$$\hat{X}_i^k = \frac{Y_i^k}{\hat{H}^k} \tag{6}$$

where $\hat{X}_i^k$ is the recovered QAM symbol at the kth sub-carrier of the ith OFDM symbol. LS is a simple and computational efficient algorithm, however it only works well when the frame length is small or the channel has long coherence time.

The LMS method resolves the drawback of LS by updating the estimated channel at each OFDM symbol. It starts with the same initial channel estimates as LS. The QAM symbols at the ith OFDM symbol are recovered by the channel estimates at the $(i-1)$th OFDM symbol

$$\hat{X}_i^k = \frac{Y_i^k}{\hat{H}_{i-1}^k}, \tag{7}$$

then $\hat{X}_i^k$ is mapped to the nearest QAM constellation[1], denoted as $\tilde{X}_i^k$, finally the channel estimates at current OFDM symbol are updated by

$$\hat{H}_i^k = (1-\alpha)\hat{H}_{i-1}^k + \alpha\frac{Y_i^k}{\tilde{X}_i^k}, \tag{8}$$

where $\alpha \in [0, 1]$ is an averaging factor of the channel updating procedure. Note LMS uses decision feedback to estimate channel, the performance suffers when decision is erroneous.

As the name suggests the Comb method uses the channel estimated at four pilot positions to interpolate the whole channel in frequency domain. Comb also starts with the same initial channel estimate as LS, at ith OFDM symbol, the channel at four pilot sub-carriers are estimated by least square

$$\hat{H}_i^k = \frac{Y_i^k}{X_i^k}, \quad k \in \{11, 25, 39, 53\}, \tag{9}$$

then the channel at sub-carrier 0 and 63 are set to the mean of the four estimated values

$$\hat{H}_i^0 = \hat{H}_i^{63} = \mathbb{E}[\hat{H}_i^k], \quad k \in \{11, 25, 39, 53\}. \tag{10}$$

The channel at rest sub-carriers are estimated by linearly interpolating with above six values where a time averaging similar to (8) is applied to update the channel.

III. Reservoir Computing Based Symbol Detection

A. *Reservoir Computing and Echo State Network*

As discussed in Section I, RC [14] is a special category of RNN where the RNN is used as a reservoir mapping the input signals to very high dimensional dynamic states. An output layer then maps those states to desired outputs. The signature of RC is only the output weights are trainable, the input and recurrent weights are initially randomly generated and then fixed. In general, there are two types of RC: ESN and Liquid State Machine (LSM) [15]. Like our previous work, we adopt ESN in this work where its architecture can be illustrated in Fig. 4. The network dynamics is governed by the following state update equation

$$\boldsymbol{s}(t) = f\big(\boldsymbol{W}\boldsymbol{s}(t-1) + \boldsymbol{W}^{in}\boldsymbol{i}(t) + \boldsymbol{W}^{fb}\boldsymbol{o}(t-1)\big) + \boldsymbol{n}(t) \tag{11}$$

where $\boldsymbol{s}(t) \in \mathbb{C}^{N_s}$ is the reservoir state, N_s is the number of neurons in the reservoir. $\boldsymbol{i}(t) \in \mathbb{C}^{N_i}$ is the ESN input, N_i is the input size. $\boldsymbol{o}(t) \in \mathbb{C}^{N_o}$ is the ESN output, N_o is the output size. $\boldsymbol{W} \in \mathbb{C}^{N_s \times N_s}$ is the state transition matrix, $\boldsymbol{W}^{in} \in \mathbb{C}^{N_s \times N_i}$ is the input weight matrix, $\boldsymbol{W}^{fb} \in \mathbb{C}^{N_s \times N_o}$ is the output feedback matrix, it can be nulled when feedback is not required. $\boldsymbol{n}(t)$ is the noise regulation term. f is the state activation function. The ESN output is calculated by

$$\boldsymbol{o}(t) = g(\boldsymbol{W}^{out}\boldsymbol{z}(t)) \tag{12}$$

[1]For example $\tilde{X}_i^k \in \{1+j, -1+j, 1-j, -1-j\}$ if using QPSK.

where $\boldsymbol{z}(t) = [\boldsymbol{s}(t), \boldsymbol{i}(t)]$ is called extended system state. $\boldsymbol{W}^{out} \in \mathbb{C}^{N_o \times (N_s+N_i)}$ is the output weights matrix. g is the output activation function.

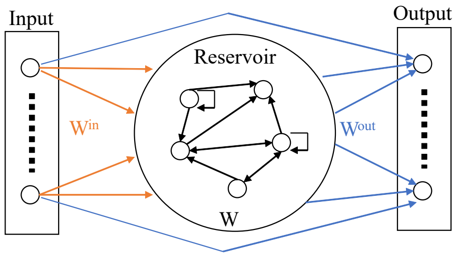

Fig. 4. ESN architecture

Assuming the training data size is T, the learning of the output weights can be formulated using the following

- Feeding training input $\boldsymbol{I} = [\boldsymbol{i}(0), \cdots, \boldsymbol{i}(T-1)]$ to ESN to generate extended states $\boldsymbol{Z} = [\boldsymbol{z}(0), \cdots, \boldsymbol{z}(T-1)]$ by (11), and outputs $\boldsymbol{O} = [\boldsymbol{o}(0), \cdots, \boldsymbol{o}(T-1)]$ by (12).
- Obtaining $\boldsymbol{W}^{out}$ by minimizing the loss between outputs $\boldsymbol{O}$ and training label $\boldsymbol{D} = [\boldsymbol{d}(0), \cdots, \boldsymbol{d}(T-1)]$

$$\boldsymbol{W}^{out} = \underset{\boldsymbol{W}^{out}}{\operatorname{argmin}} L(\boldsymbol{O}, \boldsymbol{D}) \tag{13}$$

When g in (12) is the identity function and L is the Frobenius norm, the minimization problem can be rewritten as

$$\boldsymbol{W}^{out} = \underset{\boldsymbol{W}^{out}}{\operatorname{argmin}} \|\boldsymbol{D} - \boldsymbol{W}^{out}\boldsymbol{Z}\|_F^2 \tag{14}$$

which can be solved by invoking the Moore-Penrose inverse (denoted by $\cdot^{\dagger}$) of $\boldsymbol{Z}$

$$\boldsymbol{W}^{out} = \boldsymbol{D}\boldsymbol{Z}^{\dagger} \tag{15}$$

this is an offline learning method. Many online learning methods can also be used to calculate the output weights, such as Recursive Least Square (RLS) [16], Backpropagation Decorrelation (BPDC) [17], and FORCE learning [18].

B. ESN-based Symbol Detection for Wi-Fi Systems

The goal of ESN-based symbol detection is to recover the frequency domain symbols $\boldsymbol{X}_i$ from the time domain observations $\boldsymbol{y}_i$. Here we use supervised learning to train the underlying ESN. The training labels come from the STS and LTS symbols at the beginning of a frame, and 4 pilot sub-carriers in each data symbol. This means we will not utilize any extra training sets/overhead compared to the existing Wi-Fi receiver. This completely distinguishes our work from other NN-based methods where significantly large training overhead is needed. The training procedure is described in the following where the main idea is using STS and LTS to train the initial ESN output weights, and then utilizing pilots in each data symbol to update those weights in real-time:

- Training through STS and LTS: As described in II-A, at the begging of each frame, the first 4 OFDM symbols (STS and LTS) are known symbols used for synchronization and initial channel estimation. We can also utilize those symbols for the initial training of ESN. The training set can be represented by the input-label tuple:

$$\begin{aligned}\Phi_{ini} &\triangleq \{\boldsymbol{I}_{ini}, \boldsymbol{D}_{ini}\} &(16)\\ &= \{[\boldsymbol{y_1}, \boldsymbol{y_2}, \boldsymbol{y_3}, \boldsymbol{y_4}], [\boldsymbol{X_1}, \boldsymbol{X_2}, \boldsymbol{X_3}, \boldsymbol{X_4}]\}\\ &\cong \{[\boldsymbol{y_1}, \boldsymbol{y_2}, \boldsymbol{y_3}, \boldsymbol{y_4}], [\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \boldsymbol{x_4}]\} &(17)\end{aligned}$$

where $\cong$ represents "equivalently defined as". The initial ESN output weights can be calculated by (15).
- Training through pilots: As the channel is changing dynamically, the ESN weights need to be updated in real-time to track the channel. We utilize the 4 pilot sub-carriers in each OFDM symbol to achieve this. For each data symbol (the ith symbol, $i \geq 4$), the training tuple $\Phi_i \triangleq \{\boldsymbol{I}_i, \boldsymbol{D}_i\}$ is prepared as:

$$\boldsymbol{I}_i = \boldsymbol{Y}_i\boldsymbol{P}\boldsymbol{F}^H, \qquad \boldsymbol{D}_i = \boldsymbol{X}_i\boldsymbol{P}\boldsymbol{F}^H$$

where $\boldsymbol{F}^H$ is the inverse Fourier transform matrix, $\boldsymbol{P}$ is a $N_{sc} \times N_{sc}$ diagonal matrix with the nth diagonal element p_n defined as:

$$p_n = \begin{cases} 1, & n = 11, 25, 39, 53; \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

It can be seen both the input and the label are the time domain OFDM waveforms purely determined by pilot symbols. In terms of the training method, instead of the one-shot matrix inversion used for initial training, RLS online training [16], [19] is more suitable here, because it gradually updates the ESN weights based on new training samples achieving more robust performance by utilizing channel's temporal correlation. Similar ideas also have been adopted by conventional symbol detection methods, which average the channel in time to gain better performance (8). After the output weights have been updated by new training sample Φ_i, ESN will take the ith received symbol $\boldsymbol{y}_i$ as input to inference the transmitted symbol.

IV. Implementation on SDR Platform

To evaluate the performance of our algorithm in real system, we implemented the introduced RC-based symbol detection method on the SDR platform. The software adopted is GNU Radio[2], which is an open-source development kit for implementing signal processing functions. The hardware used is Ettus USRP N210[3], which provides radio frequency (RF) front-end functionalities.

We utilize the Wi-Fi transceiver framework provided in [20], which is open sourced on GitHub[4]. Since the Wi-Fi transmitter is relatively simple and fully specified by the standards, we adopted the one provided by the framework, with the modification of adding a block to save transmitted OFDM symbols on local memory for bit error rate (BER) calculation on the receiver side. The GNU Radio block diagram of transmitter is depicted in Fig. 5, and the workflow is described below:

[2]https://www.gnuradio.org/
[3]https://www.ettus.com/all-products/un210-kit/
[4]https://github.com/bastibl/gr-ieee802-11

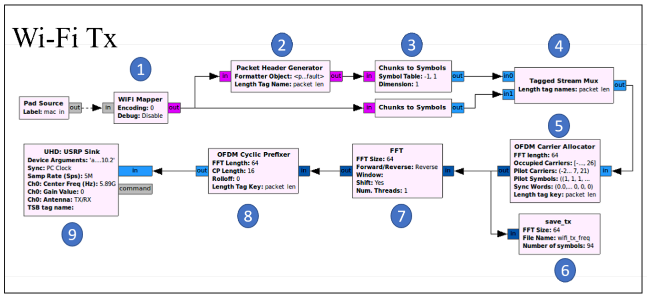

Fig. 5. GNU Radio diagrams for the Wi-Fi transmitter

(1). The payload bits from MAC layer are encoded by convolutional code with rate 1/2.
(2). The packet header generator creates signal field of the frame, including information such as MCS and frame length to facilitate receiver procedure.
(3). Data bits are mapped to transmitted QAM symbols based on the underlying modulation scheme. The information in the **signal** field is modulated with BPSK.
(4). Signal field is prepended to the data field.
(5). STS and LTS symbols are added to the Wi-Fi frame while pilot symbols being inserted to data symbols.
(6). OFDM symbols at the transmitter are saved to local memory for later BER calculation.
(7). FFT is used to convert the signal to the time domain.
(8). CP of legth 16 is added for each OFDM symbol for dealing with the frequency selective fading channel.
(9). The time domain samples are transmitted by USRP.

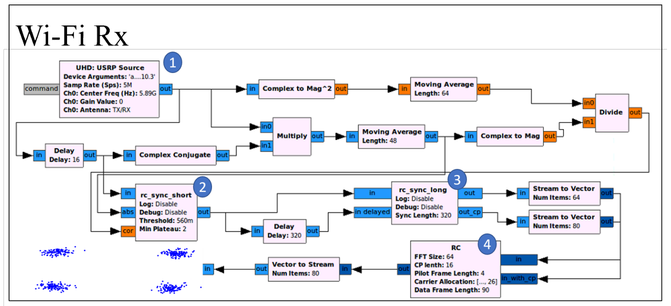

Fig. 6. GNU Radio diagrams for the RC-based Wi-Fi receiver

On the receiver side, we modified and added various receiving components based on the framework. To be specific, the synchronization block is modified and the RC-based symbol detection block is implemented. Conventional Wi-Fi symbol detection methods discussed in Section II-B are also implemented for performance comparison. The receiver block diagram is illustrated in Fig. 6 with the following workflow:

(1). USRP converts received RF signals to the baseband, and outputs them to following blocks.
(2). The frame start is detected in this block using STS.
(3). OFDM symbol alignment is achieved using LTS. Note the original block removes STS and CP from its output, however, we utilize them for ESN training. Therefore, we modified this block to include STS and CP as its outputs.
(4). This block is the core of the receiver where the RC-based symbol detection is implemented. The raw received symbols are ported into this block and the output are recovered symbols.

The SDR testbed is shown in Fig. 7 where a computer equipped with GNU Radio v3.7 is connected with two USRP N210s through Ethernet cable and a switch. One USRP is serving as the Wi-Fi transmitter and the other is the receiver. They communicate with each other over the air.

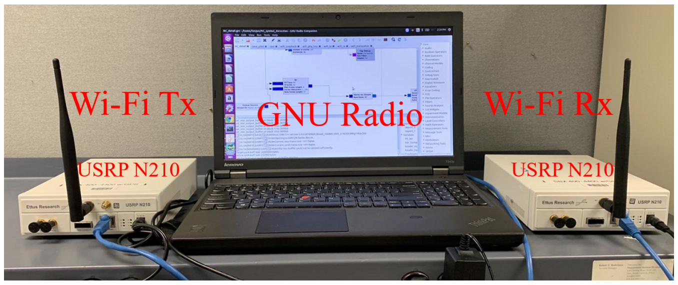

Fig. 7. SDR testbed

V. Experiment Results

The performance of RC-based symbol detection method and other conventional methods are evaluated by conducting experiments on the SDR platform. In this section we first introduce the details of the setup of the experiments, and then present the performance evaluation results.

For the Wi-Fi PHY layer configuration, one frame is set to include 94 OFDM symbols where the first 4 symbols are STS and LTS, the 5th OFDM symbol is the **signal** symbol, and the rests are data symbols. In terms of MCS, quadrature phase shift keying (QPSK) is adopted for OFDM data symbols. Even though convolutional coding with rate $1/2$ is utilized, we simply ignored this part to compute the error performance using bits that are before decoding. The sub-carrier allocation is same as described in Section II-A: 4 pilot sub-carriers, 48 data sub-carriers, and 12 null sub-carriers, the FFT size is 64, CP length is 16. For the ESN, the number of neurons N_s is set to be 16, the input weights $\boldsymbol{W}^{in}$ is randomly generated from a uniform distribution, the state transition matrix $\boldsymbol{W}$ is also randomly generated, and the the spectrum radius of the matrix is set to be 0.2 to satisfy the echo state property [21]. Feedback is not used in state update, so $\boldsymbol{W}^{fb}$ is set to zero.

The experiments are conducted indoor under three different scenarios. The first scenario is line-of-sight with near distance (LOS_Near), where the transmitter and receiver have LOS transmission path and the distance between the transmitter and the receiver is only 10 inches. The second scenario is LOS with far distance (LOS_Far), where the distance between transmitter and receiver is increased to 10 feet, in addition, a 30dB attenuator is added to the receiver side to further decrease the signal strength. The last scenario is non-line-of-sight (NLOS), where no direct path exists between transmitter and receiver, and the distance between the transmitter and the receiver is 5 feet with a 30dB attenuator on the receiver side.

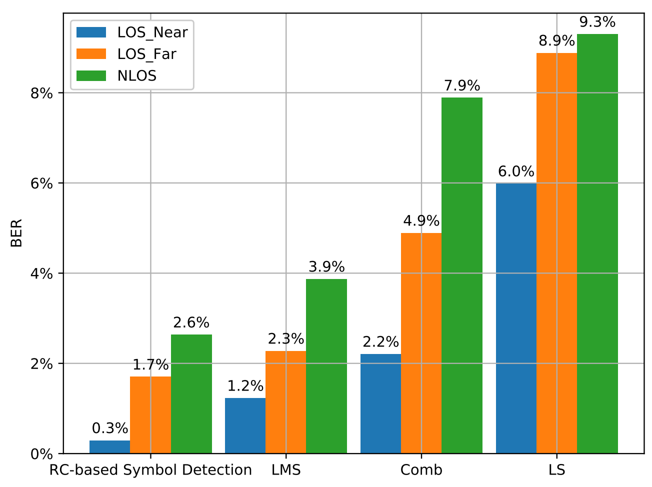

Fig. 8. Performance of RC-based symbol detection, LMS, Comb and LS

Performance of RC-based symbol detection, LMS, Comb, and LS under these 3 scenarios are shown in Fig. 8, each BER value is averaged over 20 frames[5]. It can be seen from the figure that as the experiment scenario changes from LOS_Near to LOS_Far, and then to NLOS, the wireless environment becomes more and more challenging. This results in BER degradation for all four algorithms. Among them, LS has the worst performance reflecting its drawback: LS suffers from low temporal correlation since it only keeps initial channel estimates. LMS achieves better BER performance than Comb in LOS_Near scenario because LMS estimates all data channels while Comb only estimates pilot channels. Meanwhile, in LOS_Far and NLOS scenarios, LMS still outperforms Comb. This is because even though LOS_Far and NLOS are more challenging environments, the resulting BER is still low (less than 5%) without causing error propagation. In future, we will evaluate the performance of LMS in other more challenging scenarios. As for our introduced RC-based symbol detection, it achieves the best performance in all scenarios. We believe this is due to RC's capability to efficiently learn a direct mapping from the input to the output (instead of going through channel estimation) and to adjust the underlying weights for RC in real-time to capture the underlying temporal correlation.

VI. Conclusion and Future Work

RC-based symbol detection for Wi-Fi systems using online training was introduced. It was further implemented on the SDR platform showing good BER performance under various scenarios. The introduced detector does not require any additional training overhead providing a convincing case to incorporate NN-based methods to practical wireless systems. Since multiple-input multiple-output (MIMO) is supported in many Wi-Fi standards, a nature future extension of the work is to MIMO-OFDM systems. Our previous work has shown promising performance of RC-based symbol detection in LTE/LTE-Advanced compatible systems [11], it is beneficial to evaluate the RC-based algorithm in MIMO Wi-Fi systems.

[5]Total number of data bits within 20 frames are: 20 frames $\times$ 89 data symbols $\times$ 48 data sub-carriers $\times$ 2 bits/symbol = 170,880.

References

[1] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *Std 802.11-2012*, 2012.
[2] R. Shafin, L. Liu, V. Chandrasekhar, H. Chen, J. Reed *et al.*, "Artificial intelligence-enabled cellular networks: A critical path to beyond-5G and 6G," *IEEE Wireless Commun. Mag.*, pp. 1–6, 2020.
[3] L. Li, H. Chen, H.-H. Chang, and L. Liu, "Deep Residual Learning Meets OFDM Channel Estimation," *IEEE Wireless Commun. Lett.*, pp. 1–1, 2019.
[4] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, 2017.
[5] J. Liu, K. Mei, X. Zhang, D. Ma, and J. Wei, "Online extreme learning machine-based channel estimation and equalization for OFDM systems," *IEEE Commun. Lett.*, vol. 23, no. 7, pp. 1276–1279, 2019.
[6] P. Jiang, T. Wang, B. Han, X. Gao, J. Zhang, C.-K. Wen, S. Jin, and G. Y. Li, "Artificial intelligence-aided OFDM receiver: Design and experimental results," *arXiv preprint arXiv:1812.06638*, 2018.
[7] Z. Zhao, M. C. Vuran, F. Guo, and S. Scott, "Deep-waveform: A learned OFDM receiver based on deep complex convolutional networks," *arXiv preprint arXiv:1810.07181*, 2018.
[8] N. Farsad and A. Goldsmith, "Neural network detectors for molecular communication systems," in *2018 IEEE 19th Intl Workshop on Signal Process. Advances in Wireless Commun. (SPAWC)*, 2018, pp. 1–5.
[9] B. Karanov, D. Lavery, P. Bayvel, and L. Schmalen, "End-to-end optimized transmission over dispersive intensity-modulated channels using bidirectional recurrent neural networks," *Optics express*, vol. 27, no. 14, pp. 19 650–19 663, 2019.
[10] S. S. Mosleh, L. Liu, C. Sahin, Y. R. Zheng, and Y. Yi, "Brain-Inspired Wireless Communications: Where Reservoir Computing Meets MIMO-OFDM," *IEEE Trans. Neural Netw.*, vol. 29, no. 10, pp. 4694–4708, 2018.
[11] Z. Zhou, L. Liu, and H. Chang, "Learning for Detection: MIMO-OFDM Symbol Detection through Downlink Pilots," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2020.
[12] Z. Zhou, L. Liu, V. Chandrasekhar, J. Zhang, and Y. Yi, "Deep Reservoir Computing Meets 5G MIMO-OFDM Systems in Symbol Detection," in *34th AAAI Conf. Artificial Intell.*, 2020.
[13] J. A. Fernandez, D. D. Stancil, and F. Bai, "Dynamic channel equalization for IEEE 802.11p waveforms in the vehicle-to-vehicle channel," in *2010 48th Annual Allerton Conf. on Commun., Control, and Comput.*, pp. 542–551.
[14] D. Verstraeten, B. Schrauwen, M. d'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," *Neural networks*, vol. 20, no. 3, pp. 391–403, 2007.
[15] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," *Bonn, Germany: German Natl Research Center for Inf. Technol. GMD Technical Report*, vol. 148, no. 34, p. 13, 2001.
[16] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *Advances in neural inf. process. syst. (NIPS)*, 2003, pp. 609–616.
[17] J. J. Steil, "Backpropagation-decorrelation: online recurrent learning with O (N) complexity," in *2004 IEEE Int. Joint Conf. on Neural Netw.*, vol. 2, 2004, pp. 843–848.
[18] D. Sussillo and L. F. Abbott, "Generating coherent patterns of activity from chaotic neural networks," *Neuron*, vol. 63, no. 4, pp. 544–557, 2009.
[19] B. Farhang-Boroujeny, *Adaptive filters: theory and applications*. John Wiley & Sons, 2013.
[20] B. Bloessl, M. Segata, C. Sommer, and F. Dressler, "An IEEE 802.11 a/g/p OFDM Receiver for GNU Radio," in *2nd Workshop on Software Radio Implement. Forum*, 2013, pp. 9–16.
[21] M. Lukoševičius, "A practical guide to applying echo state networks," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 659–686.