FedPDA: Collaborative Learning for Reducing Online-Adaptation Frequency of Neural Receivers

Shuo Wang*, Tianxin Wang*, Xudong Wang†
*Shanghai Jiao Tong University, Shanghai, China
†Hong Kong University of Science and Technology Guangzhou, Guangzhou, China

***Abstract*—Wireless neural receivers provide a promising alternative to conventional receivers. To perform well in different channel environments, online adaption is required. However, during this process, performance remains low. Thus, an approach called federated collaborative learning with pruned-data aggregation (FedPDA) is developed to reduce online-adaptation frequency. The basic idea is that, upon online adaptation, mobile terminals further update their neural receivers collaboratively via federated learning. To reduce memory consumption, neural receivers follow a main-side network architecture where only the side network needs retraining during collaborative learning. To avoid catastrophic forgetting during continual learning, local data on terminals are pruned, with only a small percent sent to the base station. With such data, the base station also trains a neural receiver before conducting model aggregation. FedPDA is distinct with several features: 1) small memory footprint and no storage burden on terminals; 2) no catastrophic forgetting issue; 3) low communication cost. Performance results show that FedPDA reduces online adaptation by more than 90% and memory footprint by 70%. It achieves comparable performance as centralized schemes, but reducing communication cost by 78%. Compared to vanilla federated learning, FedPDA resolves the catastrophic forgetting issue without storage burden, and also reduces the communication cost by 50%.**

I. Introduction

Machine learning (ML) and deep learning (DL)-based wireless physical layer design serve as the cornerstone of 6G networks. Typical examples of machine learning for wireless communications include channel estimation [1]–[3], channel decoding [4], [5], beamforming design [6], [7], radio resource allocation [8], [9], etc. Particularly, neural network-based wireless receivers have attracted increasing attention, because it has been shown that they can achieve promising performance [10]–[23]. In this paper, the term "neural receiver" refers to a receiver that uses neural networks to replace specific function modules or to completely substitute the entire receiver. Compared to conventional receivers, neural receivers offer the following advantages: 1) reduced pilot overhead: neural receivers can achieve comparable performance with fewer demodulation reference signals [11], [24], [25], particularly in high-frequency communication scenarios [26]; 2) implicit but accurate modeling of real-world wireless channels: by directly learning from transmitted and received data, neural receivers eliminate the need of explicit channel modeling and effectively handle complex factors such as inter-cell interference [11], clipping noise [27], power amplifier distortion [23], [28], and finite resolution quantization of analog-to-digital converters [29], [30]; 3) graceful tradeoff between complexity and performance: neural receivers can address the limitations in scenarios where non-linear receivers become impractical when the number of antennas is large or the linear receivers suffer from poor performance on ill-conditioned wireless channels [31].

In a mobile network, neural receivers can be deployed at a base station (BS) for uplink communications or at a mobile terminal for downlink communications. This paper is focused on downlink communications. A neural receiver is pre-trained offline and then deployed on mobile terminals for online use. Usually, it is impractical for the offline pre-training dataset to encompass all possible channel conditions, so the pre-trained neural receiver may not perform well in varying channel environments. Thus, it is indispensable for the neural receiver to conduct online adaptation to enhance demodulation performance [13]–[15], [17], [18], [20], [21], [27]. However, during the process of online adaptation, the neural receiver can only deliver a low data rate due to degraded demodulation performance, which negatively impacts user experience. To ensure positive user experience, it is crucial to reduce the frequency of online adaptation for neural receivers. To this end, a key insight can be considered: if the neural receiver of a terminal can acquire channel knowledge from other terminals while retaining its own, then it does not need online adaptation under a new channel environment that have been encountered before either by other terminals or itself.

The above insight can be realized through collaborative learning among neural receivers at different mobile terminals, which illustrates the advantage of neural receiver than a conventional receiver. More specifically, after a neural receiver has completed online adaptation in response to deteriorated performance, it further collaborates with other neural receivers to share channel knowledge of all environments encountered by these receivers. Such collaborative learning can be conducted via a centralized approach. However, uploading the entire dataset of online adaption to the BS leads to a huge volume of uplink communication data, which is unacceptable for mobile terminals due to high cost and power consumption. Thus, a decentralized collaborative learning method is preferred over a centralized approach. Considering the point-to-multipoint setup of a BS and its associated mobile terminals, a natural

Corresponding author: Xudong Wang (Email: wxudong@ieee.org). The research was sponsored by the National Natural Science Foundation of China (NSFC) under Grant No. 62371292.

choice of decentralized collaborative learning is federated learning, i.e., each terminal trains its neural receiver locally but the model parameters are sent to the BS for aggregation. However, methods based on vanilla federated learning [12] cannot be adopted, as they lack a few capabilities that are critical for efficient collaborative learning among terminals. First, the neural network architecture must support efficient retraining. Particularly, retraining of a neural receiver cannot demand a large memory footprint, as it is carried out on a resource-constrained mobile terminal. Second, continual learning must be achieved implicitly, as a neural receiver keeps learning new dataset corresponding to different channel environments due to mobility. Thus, the catastrophic forgetting issue, which refers to the tendency of forgetting previously acquired knowledge while learning from new training instances [32], needs to be addressed properly. Keeping all datasets for a neural receiver can avoid the catastrophic issue, but is apparently not acceptable, because of limited storage on a mobile terminal.

To this end, an approach called federated collaborative learning with pruned-data aggregation (FedPDA) is developed in this paper. To reduce memory consumption, the structure of a neural receiver is designed by considering a main-side neural network architecture [33]. The design goal is to achieve memory-efficient training of the neural receiver. Thus, a side-assisted neural receiver called SaRx is developed such that only the side network of the neural receiver needs retraining during collaborative learning, while the main network keeps same parameters of a pre-trained model. In this way, the training process of a neural receiver only requires a small memory footprint.

To resolve the catastrophic forgetting issue without causing storage burden to terminals, it is necessary to seek help from the federated server (on the BS or co-located with the BS) to remember the previous channel knowledge of all neural receivers on terminals. The existing federated learning framework does not hold such a capability, so it must be renovated in two aspects: 1) the federated server needs to obtain the previous channel knowledge from terminals; 2) the server needs to maintain its own neural receiver model to capture the obtained channel knowledge. The first aspect of renovation must be communication-efficient, so the dataset on a terminal cannot be fully uploaded to the server. Thus, a data pruning method is designed to select the top data instances whose importance is measured by its inference loss with respect to the latest global model. In this way, only a small percent of data is sent to the base station, and the remaining data is purged from local storage. In the second aspect of renovation, the server must use the same neural network structure as that of mobile terminals, and also follows the same retraining method as that in SaRx. Moreover, model aggregation on the server must include the side networks from mobile terminals and the sever itself. With the above two renovations, FedPDA gracefully resolves the catastrophic forgetting issue with low communication cost in uplink and no storage burden on terminals. It should be noted that, even though this solution compromises the data privacy of federated learning in principle, it is not a concern, as sharing channel knowledge between the BS and mobile terminal is a common practice in wireless communications.

Overall, FedPDA is distinct with several features: 1) small memory footprint, as only a side network of the neural receiver needs retraining on terminals; 2) no catastrophic forgetting issue, as continual learning is achieved by a renovated federated learning framework; 3) no storage burden, as the dataset on terminals is continuously pruned and purged; 4) low communication cost, as only model parameters and a small percent of channel data are sent to the BS.

The following contributions are made in this paper:

- A federated collaborative learning approach called FedPDA is developed to share channel knowledge of different neural receivers. It can significantly reduce occurrence frequency of online adaptation of neural receivers, which thus greatly improves performance of neural receivers under varying channel environments.
- A side-assisted neural network structure called SaRx is incorporated into FedPDA to enable memory-efficient training of neural receivers.
- The federated learning framework in FedPDA is renovated to resolve the catastrophic forgetting issue for continual learning with high storage efficiency.

Additionally, simulation experiments are carried out to validate effectiveness of FedPDA. Performance results show that, FedPDA reduces memory footprint by 70% via the SaRx structure. Furthermore, the occurrence frequency of online adaption is decreased by more than 90%. FedPDA achieves comparable performance as centralized schemes, but reduce communication cost by 78%. As compared to vanilla federated learning schemes, FedPDA can carry out continual learning without catastrophic forgetting and does not impose storage burden on terminals, with a price of a slightly increased communication cost.

The rest of this paper is organized as follows. The system model and setup is presented in Section II. The detailed design of FedPDA is described in Section III. Performance evaluation is conducted in Section IV. The paper is concluded in Section V.

II. System Model and Setup

A. OFDM Transceiver and Channel Model

Overall, single-input single-output (SISO) downlink communication scenario is considered, which implies that both the number of transmit antennas and receive antennas are limited to one. A conventional OFDM transmitter is adopted. Signals propagating in a wireless channel are impacted by two components: 1) channel attenuation, which can be characterized by a model such as 3GPP tapped delay line (TDL) models [34], and 2) Gaussian white noise. As shown in Fig. 1, a frequency-domain end-to-end neural receiver structure is adopted. It replaces the entire demodulation module of conventional receivers by a single neural network.

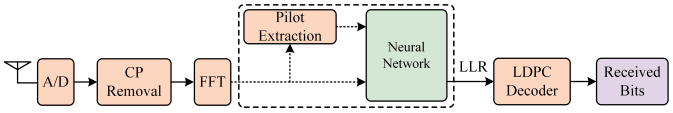


Fig. 1: Illustration of a fully learned neural receiver.

B. Online Adaptation Process

The actual process of online adaptation is separate from the work of this paper. However, for the purpose of clarity, a simple procedure of online adaptation is provided as follows. In general, online adaptation of a neural receiver is comprised of three steps. In the first step, when a terminal encounters a new channel condition where the bit error rate (BER) exceeds a predefined threshold, it sends an adaptation request to the BS. In the second step, after the BS receives the adaptation request, it sends the adaptation reference signals (ARSs) back to the terminal. The original form of these signals can be represented as $\{\mathbf{Q}^{(i)}, i = 1, 2, \ldots, N_q\}$, where N_q denotes the total number of ARSs. In fact, the ARSs are pre-defined bit streams that are transmitted in some allocated resource blocks, interleaving with regular communication data. In the third step, the terminal gets the received signal $\mathbf{Y}^{(i)}$ as well as the demodulation reference signal $\mathbf{P}^{(i)}$, both in frequency domain. Thus, it can form a new dataset $\mathcal{D} = \{(\mathbf{Z}^{(i)}, \mathbf{Q}^{(i)})\}_{i=1}^{N_q}$, where $\mathbf{Z}^{(i)}$ is the concatenation of $\mathbf{Y}^{(i)}$ and $\mathbf{P}^{(i)}$. Based on this new dataset, the neural receiver is retrained.

After the parameters of the neural receiver are updated, BER is expected to decrease below the specified threshold, and finally online adaptation is completed.

C. Channel Knowledge Fusion Process

For the sake of simplicity, we consider a scenario where N_u terminals are communicating with a single base station $\mathcal{S}$, but it should be noted that the approach developed in this paper can also be applied to scenarios with multiple base stations. The overall timeline is divided into intervals denoted as $[t_0, t_1, t_2, ...]$. The offline pre-trained model is obtained before time t_0, and each neural receiver is deployed with a model $\boldsymbol{\theta}_{G,t_0}$ at t_0. During the time interval from t_{k-1} to t_k, the terminals who have completed online adaptation and possess new channel knowledge for sharing are referred to as **critical terminals**, denoted by $\mathcal{G}_{t_k}$. The proportion of critical terminals among all is known as the **critical ratio**, denoted by $\gamma_{t_k} = \frac{N_{\mathcal{G}_{t_k}}}{N_u}$, where $N_{\mathcal{G}_{t_k}}$ is the number of critical terminals during $t_{k-1} \sim t_k$.

During each time interval, all critical terminals that have completed online adaptation follow an approach to share and fuse their channel knowledge. This approach is expected to obtain an updated global model, denoted as $\boldsymbol{\theta}_{G,t_k}$, for all neural receivers. The process and the necessity of channel knowledge fusion are explained in a simple scenario shown in Fig. 2 where one time interval is illustrated in details.

Assuming that, prior to time t_0, the BS has completed offline training using a dataset collected from channel environment E_0, and subsequently has distributed the pre-trained model parameters $\boldsymbol{\theta}_{G,t_0}$ to all terminals.

During the time interval from t_0 to t_1, terminals denoted as group $\mathcal{G}_1$ move from environment E_0 to environment E_1, and the rest remain in E_0. Since terminals arriving at the new environment E_1 may experience performance degradation, online adaptation is needed, as illustrated by the first box in interval $t_0 \sim t_1$. Upon online adaptation, these terminals work together to share their latest channel knowledge with the BS and other terminals, and then an updated global model with parameters $\boldsymbol{\theta}_{G,t_1}$ is obtained at the BS, which is illustrated by the second box in interval $t_0 \sim t_1$. Finally, the updated model is distributed to all terminals in environments E_0 and E_1, as shown in the third box in interval $t_0 \sim t_1$. Thus, starting at t_1, neural receivers on all terminals are expected to perform well in environments E_0 and E_1, even if the terminals that remain in E_0 have not yet experienced the new environment E_1. In other words, those terminals will not need online adaptation again if they arrive in the new environment E_1. As the above process continues, online adaptation frequency is significantly reduced.

To enable such a promising capability, it is critical to develop a collaborative learning approach to fuse channel knowledge of critical terminals and obtain a global neural receiver model for all terminals located in different channel environments.

III. Federated Collaborative Learning with Pruned-Data Aggregation

As mentioned in Section I, collaborative learning for channel knowledge fusion needs to consider several design requirements, i.e., small memory footprint, no catastrophic forgetting, no extra storage burden, and low communication cost. Neither centralized schemes nor vanilla federated learning schemes are effective to satisfy all such requirements. To this end, an approach called federated learning with pruned data aggregation (FedPDA) is developed in this section to handle all such requirements effectively.

A. Overall Architecture

As explained in Section I, FedPDA is designed based on the federated learning framework but with two renovations. First, the neural network structure is designed with a main-side architecture, such that retraining of a neural receiver is memory efficient. Second, local dataset used for training a neural receiver at a terminal is pruned to a small percent and then sent to the BS, while the rest is purged. Such a small percent of data accumulate at the BS and help the BS learn the knowledge of all previous channel environments encountered by terminals.

Based on these two renovations, FedPDA is developed with the following key functionalities: 1) side network-assisted neural receiver (SaRx) for memory-efficient model retraining; 2) channel knowledge fusion, in which the BS and the critical terminals collaborate to fuse latest and previous channel knowledge; 3) global dataset updating, in which local dataset at terminals are pruned and purged, but the global dataset at the BS is enriched with pruned data from terminals.

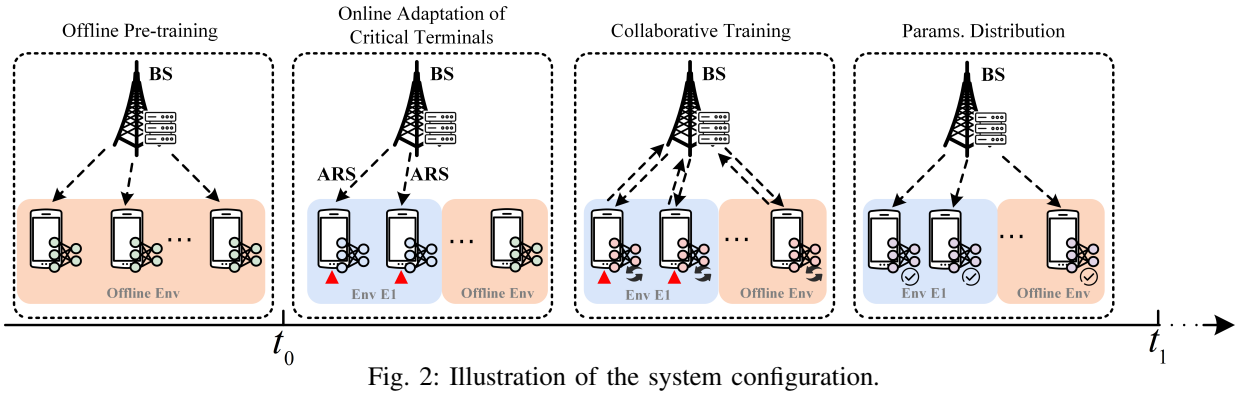
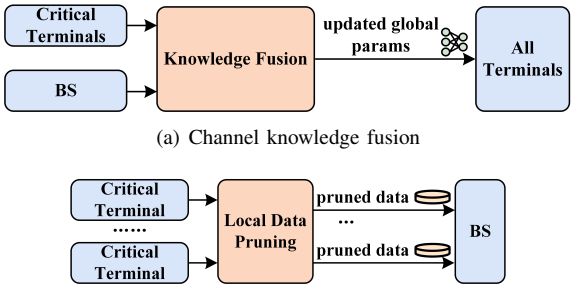


Fig. 2: Illustration of the system configuration.



(a) Channel knowledge fusion

(b) Global dataset updating

Fig. 3: The overall procedure of FedPDA

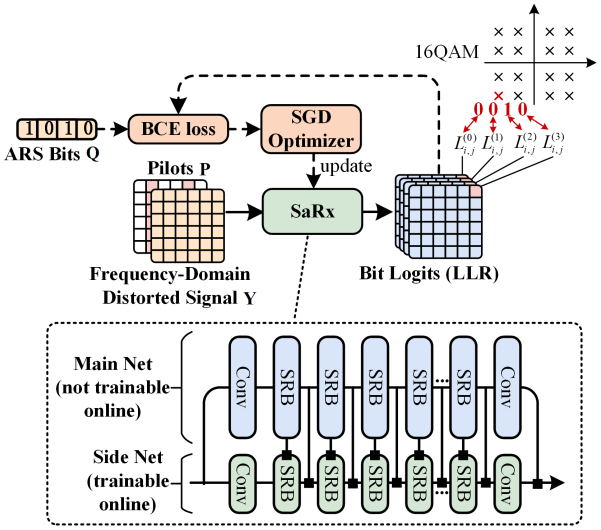


Fig. 4: Illustration of the side-assisted neural receiver structure.

To consolidate these functionalities, the overall procedure of FedPDA is illustrated in Fig. 3. During the channel fusion stage (Fig. 3(a)), the critical terminals conduct training collaboratively following a federated learning paradigm, but with two distinct features. First, all neural receivers should follow the SaRx structure. Second, the BS also trains its own neural receiver using the global dataset. Through model aggregation between the BS and critical terminals, a global model is obtained and then distributed to all terminals. During the global dataset updating stage (Fig. 3(b)), each critical user prunes its local dataset obtained in online adaptation. The pruning process follows a certain metric such as inference loss to select the most important instances, and then uploads the selected instances to the BS. In order to keep communication cost low, the data pruning process must ensure a desired data pruning ratio λ is satisfied. Once the BS receives the updated data instances, it merges such pruned data with previous ones on the global dataset. As a result, the global dataset is enriched to record the channel knowledge of all historical channel environments.

B. Side-Assisted Neural Receiver Structure

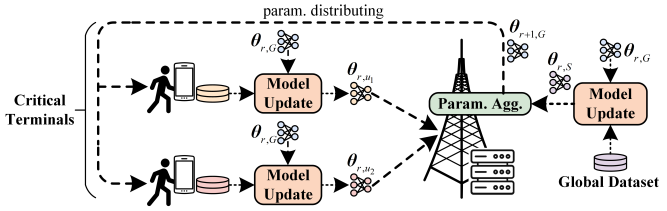
As shown in Fig. 4, SaRx consists of two parts: the main network $\boldsymbol{f}_m(\cdot;\boldsymbol{\theta}^m)$ and the side network $\boldsymbol{f}_s(\cdot;\boldsymbol{\theta}^s)$. In this paper, both the main network and the side network in SaRx adopt the fully convolutional neural network structure. The main network consists of 1 input convolutional layer, 11 separable residual blocks (SRB) [35], and 1 output convolutional layer. Each SRB consists of 2 depthwise separable convolutional layers and 2 layer normalization (LN) layers [11]. The side network is a lightweight version of the main network, where the size of the convolutional kernels in the side network are $\frac{1}{\epsilon}$ times of those in the main network (ϵ is the reduction factor, e.g., $\epsilon = 2, 4, 8$).

The main network and side network are connected by the skip connections, i.e., 1×1 trainable convolutional layers, which are located at the output of each convolutional layer except the input layer as shown in Fig. 4. The input of SaRx (denoted as $\mathbf{Z}$) is the concatenation of the frequency domain received signal $\mathbf{Y}$ and the frequency domain DMRS $\mathbf{P}$. It is then fed into both the main network and side network, respectively. The final output is obtained by aggregating the outputs of these networks, expressed as

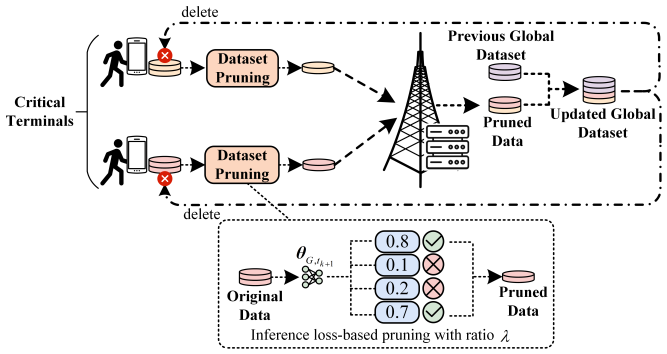
$$\boldsymbol{f}(\mathbf{Z};\boldsymbol{\theta}^m,\boldsymbol{\theta}^s) = \boldsymbol{f}_m(\mathbf{Z};\boldsymbol{\theta}^m) + \boldsymbol{f}_s(\mathbf{Z};\boldsymbol{\theta}^s). \tag{1}$$

The main network is pre-trained using offline collected dataset and its model parameters $\boldsymbol{\theta}^m$ are fixed after being deployed; Only the model parameters of the side network $\boldsymbol{\theta}^s$ are updated during collaborative learning.

Since the memory usage during neural network training is mainly determined by the amount of intermediate activations in gradient back propagation [33], the amount of intermediate activations in SaRx is reduced by approximately $\frac{1}{\epsilon}$ times by employing smaller convolution kernels in the side network.



(a) Channel knowledge fusion stage.



(b) Global dataset updating stage.

Fig. 5: Illustration of the developed FedPDA approach.

SaRx is trained with the binary cross-entropy (BCE) loss, which calculates the difference between each ARS and its corresponding output log-likelihood ratios (LLRs) as follows:

$$\mathcal{J}_{\mathrm{BCE}}^{(m)} = \frac{1}{N_t N_f K} \sum_{i=0}^{N_t-1} \sum_{j=0}^{N_f-1} \sum_{k=0}^{K-1} (\mathbf{Q}_{i,j,k}^{(m)} \log(\hat{\mathbf{L}}_{i,j,k}^{(m)}) + (1-\mathbf{Q}_{i,j,k}^{(m)}) \log(1-\hat{\mathbf{L}}_{i,j,k}^{(m)})) \tag{2}$$

where m is the sample instance, K represents the modulation order, $\mathbf{Q}_{i,j,k}^{(m)}$ is the k-th transmitted bit on the resource element (i,j), N_t is the number of OFDM symbols, and N_f is the number of subcarriers. $\hat{\mathbf{L}}_{i,j,k}^{(m)}$ is given as $\hat{\mathbf{L}}_{i,j,k}^{(m)} = \mathrm{sigmoid}(\mathbf{L}_{i,j,k}^{(m)}) = \frac{1}{1+e^{-\mathbf{L}_{i,j,k}^{(m)}}}$, where $\mathbf{L}_{i,j,k}^{(m)}$ is the LLR of k-th transmitted bit on the resource element (i,j).

C. Channel Knowledge Fusion

The channel knowledge fusion stage is illustrated in Fig. 5(a). At the beginning of round r, the BS selects all the critical terminals $\mathcal{G}_{t_k}$ to participate in channel fusion. Subsequently, each critical terminal $u \in \mathcal{G}_{t_k}$ updates the global model of previous round $\boldsymbol{\theta}_{r,G}^s$ using its local adaptation dataset $\mathcal{D}_u$ to obtain $\boldsymbol{\theta}_{r,u}^s$. Meanwhile, BS updates the global model of last round $\boldsymbol{\theta}_{r,G}^s$ using the global dataset $\mathcal{D}_{\mathcal{S},t_{k-1}}$ and then obtains $\boldsymbol{\theta}_{r,\mathcal{S}}^s$.

Subsequently, the critical terminals send the updated models $\{\boldsymbol{\theta}_{r,u}^s, u \in \mathcal{G}_{t_k}\}$ to the BS where these models are aggregated together with the BS-side model $\boldsymbol{\theta}_{r,\mathcal{S}}^s$ to obtain the updated global model $\boldsymbol{\theta}_{r+1,G}^s$, as

$$\boldsymbol{\theta}_{r+1,G}^s = \alpha_{\mathcal{S}} \boldsymbol{\theta}_{r,\mathcal{S}}^s + (1-\alpha_{\mathcal{S}}) \sum_{u \in \mathcal{G}_{t_k}} \frac{|\mathcal{D}_u|}{\sum_{u \in \mathcal{G}_{t_k}} |\mathcal{D}_u|} \boldsymbol{\theta}_{r,u}^s, \tag{3}$$

where $\alpha_{\mathcal{S}} \in (0,1)$ is the BS-side model aggregation ratio. These steps are repeated for R_P rounds, and finally an updated global model $\boldsymbol{\theta}_{G,t_k}^s$ is obtained.

D. Global Dataset Updating

The global dataset updating stage is illustrated in Fig. 5(b). When the channel knowledge fusion stage is completed, the global model parameters $\boldsymbol{\theta}_{G,t_k}^s$ incorporating the latest channel knowledge are distributed to all terminals. Subsequently, each critical terminal $u \in \mathcal{G}_{t_k}$ needs to remove the unimportant training instances from its local adaptation dataset $\mathcal{D}_u$ to obtain a pruned adaptation dataset $\mathcal{D}_u^{'}$. The BS collects the pruned adaptation dataset $\{\mathcal{D}_u^{'}, u \in \mathcal{G}_{t_k}\}$ from all critical terminals and then merges it with the current global dataset $\mathcal{D}_{\mathcal{S},t_{k-1}}$. Afterwards, the global dataset is updated to $\mathcal{D}_{\mathcal{S},t_k}$.

To measure the importance of a training instance, its inference loss with respect to the latest global model $\boldsymbol{\theta}_{G,t_k}^s$ is employed. The underlying rationale is that training instances with larger inference loss hold greater significance, as they are more likely to be misclassified by the model and can provide more information about the wireless channel. Research in [32] demonstrates that training instances with higher inference loss exhibit a greater likelihood of being forgotten by the model.

As shown in Fig. 5(b), each training instance $(\mathbf{Z}_u^{(i)}, \mathbf{Q}_u^{(i)})$ of user u is fed into the latest global model to get the inference loss $\mathcal{J}_u^{(i)}$, which is computed based on Eq. (2). Denote the indices of training instances in $\mathcal{D}_u$ as $[n] = [0, 1, \ldots, |\mathcal{D}_u|]$. The pruned adaptation dataset $\mathcal{D}_u^{'}$ is obtained by selecting the top K_{p} instances with the largest inference loss, i.e.,

$$\mathcal{D}_u^{'} = \{(\mathbf{Z}_u^{(j)}, \mathbf{Q}_u^{(j)}), j \in \arg \max_{I \subset [n]: |I| = K_{\mathrm{p}}} \sum_{i \in I} \mathcal{J}_u^{(i)}\}, \tag{4}$$

where $K_{\mathrm{p}} = (1-\lambda)|\mathcal{D}_u|$, and $\lambda \in (0,1)$ is the data pruning ratio.

E. Parameter Selection for Communication Efficiency

With the above design, FedPDA is advantageous over the vanilla federated learning schemes in memory efficiency and continual learning. However, how to ensure communication efficiency of FedPDA depends on appropriate selection of key parameters such as iteration rounds of federated learning, model size of the side network, the data pruning ratio, etc. Otherwise, the communication cost can be much higher than that of vanilla federated learning schemes or even higher than centralized schemes.

Given N_u terminals, the cost involved in uplink and downlink communications during the time interval between t_{k-1} and t_k is analyzed for different schemes. Without loss of generality, the total number of ARSs N_q for online adaptation is assumed to be constant, and adaptation dataset $\mathcal{D}_u$ for each terminal is assumed to have the same size $S_D = |\mathcal{D}_u|$.

Considering FedPDA, if the number of critical terminals in the given time interval is $\mathcal{G}_{t_k}$ and the critical ratio is γ_{t_k}, then the communication cost $C_{\gamma_{t_k}}^{\mathcal{P}}$ can be derived as

$$\begin{aligned} C_{\gamma_{t_k}}^{\mathcal{P}} &= 2R_P |\mathcal{G}_{t_k}| |\boldsymbol{\theta}^s| + |\mathcal{G}_{t_k}| (1-\lambda) S_D \\ &= \gamma_{t_k} N_u \left(2R_P |\boldsymbol{\theta}^s| + (1-\lambda) S_D\right), \end{aligned} \tag{5}$$

where R_P is the number of iteration rounds in federated learning, and λ is the data pruning ratio, which is assumed to be same for all terminals. Moreover, $|\boldsymbol{\theta}^s|$ is the model size of the side network in FedPDA.

For a centralized scheme, its communication cost $C^{\mathcal{C}}_{\gamma_{t_k}}$ is simply given as

$$C^{\mathcal{C}}_{\gamma_{t_k}} = \sum_{u \in \mathcal{G}_{t_k}} S_D = \gamma_{t_k} N_u S_D. \tag{6}$$

Considering a vanilla federated learning scheme, its communication cost $C^{\mathcal{F}}_{\gamma_{t_k}}$ can be derived as

$$C^{\mathcal{F}}_{\gamma_{t_k}} = 2 N_u R_F \left|\boldsymbol{\theta}^f\right|, \tag{7}$$

where $\left|\boldsymbol{\theta}^f\right|$ is the model size of the entire neural receiver, and R_F represents the number of iteration rounds in federated learning.

To ensure the communication cost of FedPDA is lower than that of the centralized scheme, we need $C^{\mathcal{P}}_{\gamma_{t_k}} < C^{\mathcal{C}}_{\gamma_{t_k}}$. From Eq. (5) and Eq. (6), we know that the pruning ratio λ should satisfy

$$\lambda > 1/\xi_{\mathcal{P}}, \tag{8}$$

where $\xi_{\mathcal{P}} = \frac{S_D}{2R_P|\boldsymbol{\theta}^s|}$.

Since $\lambda < 1$, Eq. (8) is always satisfied as long as $\xi_{\mathcal{P}} > 1$, which means the model size of the side network should satisfy the following condition

$$|\boldsymbol{\theta}^s| < S_D / 2R_P. \tag{9}$$

According to the ablation study mentioned in Section IV, $R_P = 30$ is sufficient even under highly diverse channel conditions. Furthermore, the ablation study shows that a minimum of $N_q = 6400$ ARSs is required to ensure optimal online adaptation performance. This leads to an adaptation dataset size $S_D = 616.4$ MB for a typical 5G resource allocation with 14 OFDM symbols and 264 subcarriers. Since the model size $|\boldsymbol{\theta}^s|$ is usually much less than 10 MB, the condition in Eq. (9) can be easily satisfied.

To make sure the communication cost of FedPDA is lower than that of a vanilla federated learning scheme, we need $C^{\mathcal{P}}_{\gamma_{t_k}} < C^{\mathcal{F}}_{\gamma_{t_k}}$. Thus, from Eq. (5) and Eq. (7), we know that the critical ratio γ_{t_k} should satisfy

$$\gamma_{t_k} < \frac{2R_F \left|\boldsymbol{\theta}^f\right|}{2R_P |\boldsymbol{\theta}^s| + (1-\lambda) S_D}. \tag{10}$$

Obviously, the above condition is not always satisfied. However, by using the values found in Section IV for parameters R_P, S_D, R_F, $|\boldsymbol{\theta}^s|$, and $\left|\boldsymbol{\theta}^f\right|$, the condition in Eq. (10) can be satisfied when the data pruning ratio λ exceeds 0.90. For a lower data pruning ratio, e.g., $\lambda = 0.80$, the communication cost of FedPDA is slightly higher than that of a vanilla scheme.

IV. Performance Evaluation

Experiments are carried out in three aspects. The first one is to validated if FedPDA can effectively reduce online-adaptation frequency. The second aspect is the evaluation of memory footprint for the SaRx structure. Comparisons between our proposed SaRx structure and the original structure are illustrated in terms of memory usage. Our strategy of pruning skip-connections between the main network and the side network is also studied to further reduce the memory footprint. The third aspect is to evaluate overall performance of FedPDA. Comparisons with both centralized and vanilla federated learning schemes are also conducted. Two types of vanilla federated learning are considered: 1) "Vanilla Federated-N", i.e., the vanilla federated learning scheme that only records the channel data of the most recent environment; 2) "Vanilla Federated-S", i.e., the federated learning scheme that stores channel data of all encountered environments. In addition, the conventional receiver (LMMSE channel estimator and LMMSE constellation equalizer) and the receiver with perfect channel state information (CSI) are also considered as benchmarks.

A. Experiment Setup

All of our experiments are conducted using the open-source link-level simulation library Sionna 0.14.0 [36]. Training and testing of a neural receiver model are conducted using TensorFlow 2.11.1 [37], and we use the TensorFlow Profiler to track the hardware resource consumption (time and memory) of neural receiver models. BER is considered as the performance metric for evaluating different types of receivers. More specifically, the coded BER is considered, which is obtained by passing LLRs through the 5G low-density parity-check (LDPC) decoder and then comparing the decoded bits to the original bit streams.

The channel environments are simulated following a type of TDL channel model. The channel impulse response of such a model is expressed as

$$h(t,\tau) = \sum_{i=1}^{n} a_i(t)\delta(\tau - \tau_i), \tag{11}$$

where $a_i(t)$ represents the amplitude of the ith tap at the delay of τ_i, which is generated using a sum-of-sinusoids model [38]. Given such a channel model, three distinct channel environments are simulated: 1) The offline environment E_0, which follows the indoor office normal delay profile; 2) The online environment E_1, which follows the Urban Macrocell (UMa) normal delay profile; 3) The online environment E_2, which follows the Rural Macrocell (RMa) normal delay profile. The detailed simulation parameters of these channels are shown in Table I.

B. Reduction of Online-Adaptation Frequency

To demonstrate the effectiveness of reducing online-adaptation frequency by FedPDA, a metric called reduction ratio is adopted. It stands for the percentage of online adaptations that are reduced by FedPDA. we assume that there are

TABLE I: Simulation Parameters

Parameter	Offline Env E_0	Online Env E_1	Online Env E_2
Carrier Frequency	4GHz		
Channel Model	TDL-A	TDL-B	TDL-E
RMS Delay Spread	$10ns \sim 30ns$	$370ns \sim 400ns$	$20ns \sim 50ns$
Max. User Speed	$0m/s$	$2m/s - 5m/s$	$17m/s - 20m/s$
Eb/N0	$0dB \sim 15dB$		
Number of PRBs	23 (276 subcarriers)		
Subcarrier Spacing	15kHz		
OFDM Symbol Duration	$71\mu s$		
Length of TTI	14 OFDM symbols/ms		
Modulation Scheme	16QAM		
Code Rate	658/1024		
No. of TX Antennas	1		
No. of RX Antennas	1		
DMRS configuration	block type		

TABLE II: The Configuration of Model Structures in Various Schemes

	E2E	SaRx-FSC	SaRx-RSC	SaRx-LSC	SaRx-NSC
No. of Model Params	851,592	851,592	796,984	798,024	798,252
No. of Trainable Params	851,592	190,404	135,796	136,836	137,064
Indices of Skip Connections	$1 \sim 23$	$1 \sim 23$	$12 \sim 23$	$1 \sim 11$	23

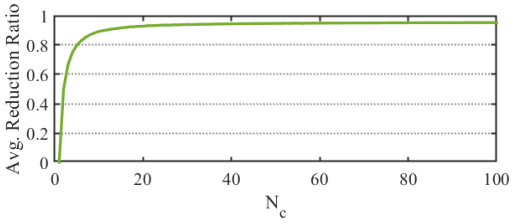


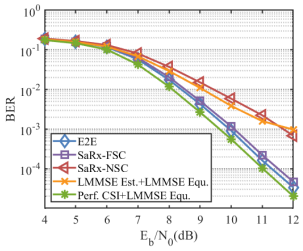
Fig. 6: Online adaptation frequency versus the number of environment change N_c.

$N_u = 24$ terminals, and each terminal can only encounter a single online channel environment (whether new or old) during each time interval $t_{k-1} \sim t_k$, where k is the index of an interval. At the end of time interval t_k, collaborative learning is conducted among all terminals if online adaptation occurs in time interval $t_{k-1} \sim t_k$. The average reduction ratio of online adaptations for one terminal is shown in Fig. 6. It indicates that, after experiencing a sufficient number of different channel environments (e.g., 10), the average reduction ratio of online adaptations can quickly reach 90%.

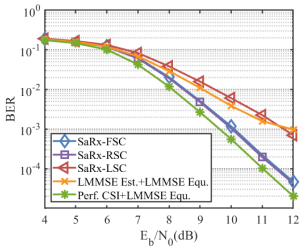
C. Memory Footprint

In this experiment, only one neural receiver is considered, and memory consumption is evaluated while the neural receiver is being retrained. The experiment is conducted on a server with Intel (R) Core (TM) i9–10900KF CPU @3.70GHz and NVIDIA GeForce GTX 3090 GPU. Moreover, only two environments are considered, i.e., the offline environment E_0 and online environment E_1. The offline pre-trained dataset consists of $108,800$ transmission time intervals (TTIs), which are randomly generated based on the offline environment E_0. The online adaptation dataset consists of 6400 TTIs that are randomly generated using the online environment E_1. The main network is pre-trained using the offline pre-trained dataset(using Adam optimizer with learning rate 0.001, batch size 32, and 30 epochs).

The SaRx with full skip connections scheme (SaRx-FSC) is considered first, where each convolution layer of the side network is connected to the corresponding layer of the main network through a skip connection. In addition, two baselines are considered for comparison: 1) the entire network scheme (E2E), where both the model parameters of the main network and side network need to be trained online; and 2) the SaRx with no skip connections scheme (SaRx-NSC), which means that there are no intermediate skip connections between the main network and side network, and only the outputs of the main network and side network are added to generate the final output. The SaRx-NSC scheme has the same number of trainable parameters as SaRx-FSC, ensuring a fair comparison. The configurations for the model structure of these three schemes are listed in Table. II. The Adam optimizer is utilized with a learning rate of 0.001, a batch size of 32, and 30 epochs for all these schemes.



(a) Testing results of the E2E, SaRx-FSC, and SaRx-NSC schemes.



(b) Testing results of the SaRx-FSC, SaRx-LSC, and SaRx-RSC schemes.

Fig. 7: Performance of various SaRX schemes.

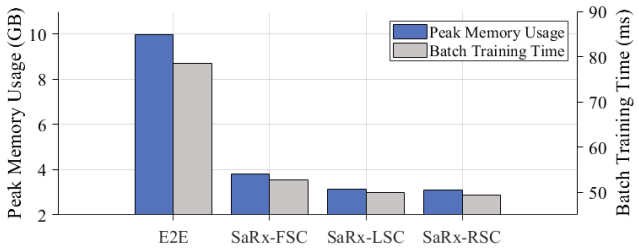


Fig. 8: The testing results of peak memory usage and batch training time.

The BER testing results of E2E, SaRx-FSC, and SaRx-NSC are depicted in Fig. 7(a). The testing dataset consists of $28,800$ randomly generated TTIs based on the online environment E_1. The results indicate that the E2E performs the best among all schemes, while SaRx-FSC achieves nearly identical performance to E2E. The SaRx-NSC scheme exhibits the poorest performance due to its lack of information from the main network. This findings suggest that the information from the main network is crucial for learning in the side network, and by incorporating this information, SaRx with full skip connections can achieve comparable performance as the E2E scheme. The peak memory usage and average batch computation time results of SaRx-FSC scheme and the E2E scheme are shown in Fig. 8. The results indicate that the SaRx-FSC scheme exhibits an average reduction of 61.96% in memory footprint and achieves an average reduction of 32.70% in average batch computation time, as compared to the E2E scheme.

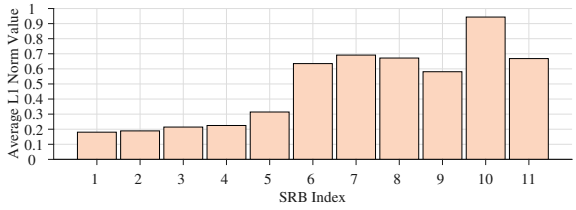


Fig. 9: $\ell 1$-norm value for updated model parameters of each SRB in the side network.

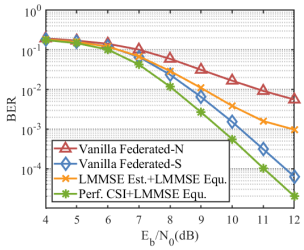


Fig. 10: Catastrophic forgetting of vanilla federated approaches.

To further evaluate the SaRx-FSC scheme with respect to different SRBs, we obtain the average $\ell 1$-norm of the updated model for each SRB in the side network as

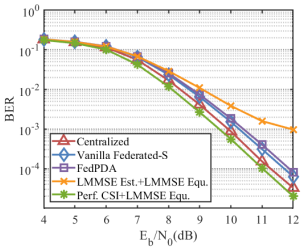
$$\chi_k = \frac{\|\boldsymbol{\theta}^{(n)}_{\mathrm{SRB}_k} - \boldsymbol{\theta}^{(0)}_{\mathrm{SRB}_k}\|_1}{|\boldsymbol{\theta}^{(n)}_{\mathrm{SRB}_k}|}, \tag{12}$$

where $\boldsymbol{\theta}^{(0)}_{\mathrm{SRB}_k}$ represents the initial parameters of the k-th SRB, and $\boldsymbol{\theta}^{(n)}_{\mathrm{SRB}_k}$ denotes the final model parameters of the k-th SRB after $n = 30$ epochs of training. The results are shown in Fig. 9, indicating that the right-sided SRBs have a higher number of updates compared to the left-sided SRBs. Therefore, the right-sided skip connections are considered more important in ensuring that the side network can receive sufficient information about the high-level features from the main network.

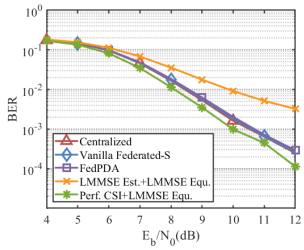
Based on these results, SaRX-FSC can be further simplified by pruning less important SRBs. Two schemes are considered: 1) the left-sided skip connection (SaRx-LSC) scheme, in which only the left-sided skip connections are retained; and 2) the right-sided skip connection (SaRx-RSC) scheme, in which only the right-sided skip connections are retained. The BER testing results depicted in Fig. 7(b) demonstrate that SaRx-RSC exhibits comparable performance to SaRx-FSC, while the SaRx-LSC scheme demonstrates poor performance. Furthermore, the results depicted in Fig. 8 demonstrate that the SaRx-RSC scheme can further reduce memory usage (by approximately 68.69%) and average batch computation time (by approximately 36.97%), as compared to the SaRx-FSC scheme.

D. Overall Performance Evaluation

All three environments E_0, E_1, and E_2 are considered in this set of experiments. Offline pre-trained dataset consists of $108,800$ TTIs, which are randomly generated based on offline environment E_0. Moreover, 24 terminals originally stay in



(a) Testing results on environment E_1.



(b) Testing results on environment E_2.

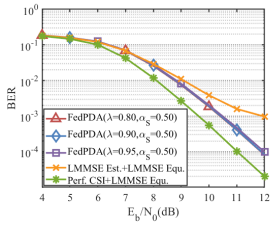
Fig. 11: Performance of mitigating catastrophic forgetting.

environment E_0. At time t_0, 12 terminals (as the first group) move to environment E_1, and at time t_1, the remaining 12 terminals (as the second group) move from E_0 to E_2. Thus, $\gamma_{t_1} = \gamma_{t_2} = 0.5$. For each terminal in both groups, the online adaptation dataset consists of $N_q = 6400$ TTIs of data, resulting in a dataset size of $S_D = 616.40$ MB. For FedPDA, the number of federated rounds in the channel knowledge fusion stage is set to $R_P = 30$, and each terminal conducts 5 local iterations in every federated round. For the vanilla federated approach, the number of federated rounds R_F is also set to 30, and each user conducts 5 local iterations in every federated round. The above values of N_q, R_P, and R_F are determined based on a separate set of experiments, but details are not included here due to page limit.

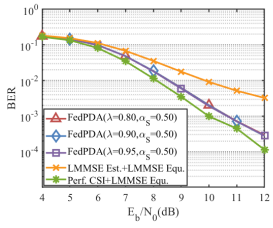
The neural network in FedPDA employs the SaRx-RSC structure, resulting in a side network model size of 0.522 MB. SaRx is not considered in the centralized scheme and the vanilla scheme. Three different data pruning ratios are considered for FedPDA: $\lambda = 0.80$, $\lambda = 0.90$, and $\lambda = 0.95$.

1) Catastrophic Forgetting and BER Performance: To evaluate the impact of catastrophic forgetting, the final updated global model at time t_2 is tested in environment E_1. The results in Fig. 10 show that the performance of a neural receiver based on federated learning can be significantly impaired by catastrophic forgetting, if previous channel knowledge is not retained (i.e., using "Vanilla Federated-N").

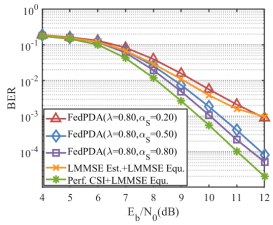
To compare performance of different approaches to mitigating the catastrophic forgetting, testing results of the neural receiver in environments E_1 and E_2 are illustrated in Fig. 11(a) and Fig. 11(b), respectively. The testing datasets of environments E_1 and E_2 consist of $28,800$ randomly generated TTIs, respectively. Results in Fig. 11(a) show that FedPDA ($\lambda = 0.80$, $\alpha_{\mathcal{S}} = 0.50$) achieve comparable performance as the vanilla federated, even if it does not store any data of previous channel knowledge. Moreover, the performance of FedPDA is only slightly lower than the centralized approach. Results in Fig. 11(b) show that FedPDA ($\lambda = 0.80$, $\alpha_{\mathcal{S}} = 0.50$) can achieve the same performance as both the centralized approach and the federated approach. These results indicate that all three approaches can effectively integrate the knowledge of environment E_2 while retaining previous channel knowledge of environment E_1. Thus, when terminals located in environment E_2 now move to environment E_1, real-time adaptation is not



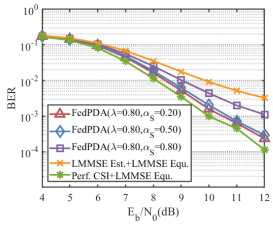
(a) Testing results on E_1 with different pruning ratios.



(b) Testing results on E_2 with different pruning ratios.



(c) Testing results on E_1 with different aggregation ratios.



(d) Testing results on E_2 with different aggregation ratios.

Fig. 12: The testing results of FedPDA with respect to different pruning ratios and model aggregation ratios.

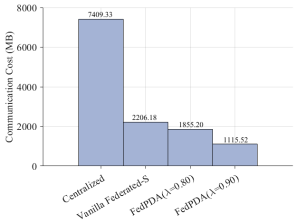
required, thus reducing the frequency of online adaptation.

Next, we investigate the impact of data pruning ratio λ on the performance of mitigating catastrophic forgetting for FedPDA. Three different values of λ (0.80, 0.90 and 0.95) are considered. The corresponding results shown in Fig. 12(a) demonstrate that FedPDA remains unaffected even when up to 90% of the local dataset is pruned. A pruning ratio of 95% leads to a noticeable decline in performance, but it remains within an acceptable range. The testing performance on environment E_2 is not sensitive to the data pruning ratio, as shown in Fig. 12(b). Hence, a large data pruning ratio up to 0.95 can be applied with little impact on the overall performance.

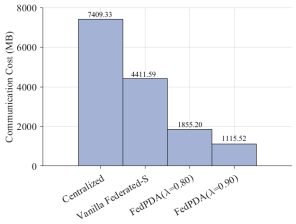
The impact of aggregation ratio α_S on the performance of FedPDA is also investigated. As indicated in Fig. 12(c), the effectiveness of preventing catastrophic forgetting is limited when α_S is small. This is because a small model aggregation ratio can result in inadequate integration of channel knowledge from previous environment E_1 into the global model. When α_S is larger, such as 0.8, prevention of catastrophic forgetting works well; however, there is a degradation in the performance of the final global model on environment E_2, as depicted in Fig. 12(d). Therefore, the performance of FedPDA is related to the aggregation ratio at the BS model, making it an important hyper-parameter.

2) Communication Cost and Storage Occupancy: The previous section demonstrates that all three approaches effectively address the issue of catastrophic forgetting, but their demands on communication cost and local storage are different.

The communication cost of different approaches is illustrated in Fig. 13. Both uplink and downlink communications are included. It can be observed that FedPDA exhibits significantly lower communication cost as compared to the



(a) Communication cost from time $t_0 = 0$ to $t_1 = 1h$.



(b) Communication cost from time $t_1 = 1h$ to $t_2 = 2h$.

Fig. 13: Communication costs of three approaches.

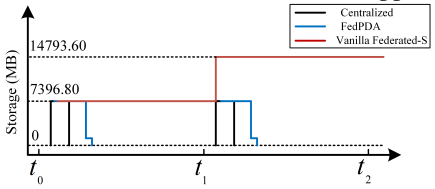


Fig. 14: Training data storage in three approaches.

centralized approach (e.g., 74.9% reduction with $\lambda = 0.80$, and 84.9% reduction with $\lambda = 0.90$ in Fig. 13(b). It also outperforms the vanilla federated approach by highly reducing the communication cost, e.g., 50.0% with $\lambda = 0.90$ in Fig. 13(a) and 74.7% with $\lambda = 0.90$ in Fig. 13(b). The performance gain over the vanilla approach is achieved, mainly because of using a much smaller neural network model (i.e., the SaRx structure).

The amount of stored channel data on terminals is shown in Fig. 14. Initially, the local data storage is low for all three approaches. However, as one round of channel fusion is completed, data storage of the vanilla federated approach doubles, since it needs to record channel data of all encountered environments. Thus, even though Vanilla Federated-S mitigates catastrophic forgetting, but imposes a large storage burden on terminals. In contrast, FedPDA does not have such an issue.

V. Conclusion

A federated learning-based approach with pruned data aggregation called FedPDA was developed to facilitate collaborative training among terminals. It effectively prevents forgetting of previously learned channel knowledge. FedPDA is distinct from existing approaches with three features: 1) much smaller memory footprint because of a novel neural receiver structure named SaRx; 2) much lower communication cost, as compared to the centralized approach; 3) no storage burden on terminals and lower communication cost, as compared to the vanilla federated approach. More diverse channels and real-world datasets with different modulation schemes will be evaluated in the future work, which helps reveal more benefits of FedPDA. Additionally, how to incorporate model personalization into FedPDA is an interesting topic for future research. The authors have provided public access to their code at https://github.com/pocketmaster123/FedPDA.

REFERENCES

[1] C.-J. Chun, J.-M. Kang, and I.-M. Kim, "Deep learning-based channel estimation for massive MIMO systems," *IEEE Wireless Communications Letters*, vol. 8, no. 4, pp. 1228–1231, 2019.

[2] E. Balevi, A. Doshi, A. Jalal, A. Dimakis, and J. G. Andrews, "High dimensional channel estimation using deep generative networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 18–30, 2021.

[3] D. Luan and J. S. Thompson, "Achieving robust generalization for wireless channel estimation neural networks by designed training data," *ICC 2023 - IEEE International Conference on Communications*, pp. 3462–3467, 2023.

[4] K. Chahine, N. Ye, and H. Kim, "DeepIC: coding for interference channels via deep learning," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 01–06.

[5] M. Yang, C. Bian, and H.-S. Kim, "OFDM-guided deep joint source channel coding for wireless multipath fading channels," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 584–599, 2022.

[6] T. Lu, H. Zhang, and K. Long, "Joint beamforming and power control for MIMO-NOMA with deep reinforcement learning," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–5.

[7] S. A. Hamza, M. G. Amin, and B. K. Chalise, "Phase-only reconfigurable sparse array beamforming using deep learning," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 4913–4917.

[8] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for radio resource allocation with diverse quality-of-service requirements in 5G," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2309–2324, 2021.

[9] F. Ortiz, N. Skatchkovsky, E. Lagunas, W. A. Martins, G. Eappen, S. Daoud, O. Simeone, B. Rajendran, and S. Chatzinotas, "Energy-efficient on-board radio resource management for satellite communications via neuromorphic computing," *IEEE Transactions on Machine Learning in Communications and Networking*, pp. 1–1, 2024.

[10] Z. Zhou, L. Liu, and H.-H. Chang, "Learning for detection: MIMO-OFDM symbol detection through downlink pilots," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 3712–3726, 2020.

[11] M. Honkala, D. Korpi, and J. M. J. Huttunen, "DeepRx: fully convolutional deep learning receiver," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3925–3940, 2021.

[12] M. B. Mashhadi, N. Shlezinger, Y. C. Eldar, and D. Gündüz, "Fedrec: federated learning of universal receivers over fading channels," in *2021 IEEE Statistical Signal Processing Workshop (SSP)*, 2021, pp. 576–580.

[13] Z. Zhou, L. Liu, S. Jere, J. Zhang, and Y. Yi, "RCNet: incorporating structural information into deep RNN for online MIMO-OFDM symbol detection with limited training," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3524–3537, 2021.

[14] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to demodulate from few pilots via offline and online meta-learning," *IEEE Transactions on Signal Processing*, vol. 69, pp. 226–239, 2021.

[15] K. M. Cohen, S. Park, O. Simeone, and S. Shamai, "Bayesian active meta-learning for reliable and efficient AI-based demodulation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 5366–5380, 2022.

[16] M. B. Fischer, S. Dörner, F. Krieg, S. Cammerer, and S. t. Brink, "Adaptive NN-based OFDM receivers: Computational complexity vs. achievable performance," in *2022 56th Asilomar Conference on Signals, Systems, and Computers*, 2022, pp. 194–199.

[17] Y. Liang, L. Li, Y. Yi, and L. Liu, "Real-time machine learning for symbol detection in mimo-ofdm systems," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, pp. 2068–2077.

[18] O. Wang, J. Gao, and G. Y. Li, "Learn to adapt to new environments from past experience and few pilot blocks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 9, no. 2, pp. 373–385, 2023.

[19] D. Garcia, R. Ruiz, J. O. Lacruz, and J. Widmer, "High-speed machine learning-enhanced receiver for millimeter-wave systems," in *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*, 2023, pp. 1–10.

[20] T. Raviv, S. Park, O. Simeone, Y. C. Eldar, and N. Shlezinger, "Online meta-learning for hybrid model-based deep receivers," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6415–6431, 2023.

[21] T. Raviv and N. Shlezinger, "Data augmentation for deep receivers," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2023.

[22] T. Raviv, S. Park, O. Simeone, Y. C. Eldar, and N. Shlezinger, "Online meta-learning for hybrid model-based deep receivers," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6415–6431, 2023.

[23] J. Pihlajasalo, D. Korpi, M. Honkala, J. M. J. Huttunen, T. Riihonen, J. Talvitie, A. Brihuega, M. A. Uusitalo, and M. Valkama, "Deep learning OFDM receivers for improved power efficiency and coverage," *IEEE Transactions on Wireless Communications*, vol. 22, no. 8, pp. 5518–5535, 2023.

[24] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018.

[25] F. Ait Aoudia and J. Hoydis, "End-to-end learning for ofdm: From neural receivers to pilotless communication," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1049–1063, 2022.

[26] Y. Liao, N. Farsad, N. Shlezinger, Y. C. Eldar, and A. J. Goldsmith, "Deep neural network symbol detection for millimeter wave communications," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[27] X. Gao, S. Jin, C.-K. Wen, and G. Y. Li, "ComNet: combination of deep learning and expert knowledge in OFDM receivers," *IEEE Communications Letters*, vol. 22, no. 12, pp. 2627–2630, 2018.

[28] S. Zheng, S. Wu, H. Li, C. Jiang, and X. Jing, "Deep learning-aided receiver against nonlinear distortion of HPA in OFDM systems," in *2021 13th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2021, pp. 1–6.

[29] D. H. N. Nguyen, "Neural network-optimized channel estimator and training signal design for MIMO systems with few-bit ADCs," *IEEE Signal Processing Letters*, vol. 27, pp. 1370–1374, 2020.

[30] L. Xu, F. Gao, T. Zhou, S. Ma, and W. Zhang, "Joint channel estimation and mixed-ADCs allocation for massive MIMO via deep learning," *IEEE Transactions on Wireless Communications*, vol. 22, no. 2, pp. 1029–1043, 2023.

[31] M. Goutay, F. A. Aoudia, J. Hoydis, and J.-M. Gorce, "Machine learning for MU-MIMO receive processing in ofdm systems," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2318–2332, 2021.

[32] M. Toneva, A. Sordoni, R. T. des Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," in *Proceedings of the International Conference on Learning Representations*, 2019.

[33] Y.-L. Sung, J. Cho, and M. Bansal, "LST: Ladder side-tuning for parameter and memory efficient transfer learning," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 12 991–13 005.

[34] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.901, 11 2020, version 16.1.0.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[36] J. Hoydis, S. Cammerer, F. Ait Aoudia, A. Vem, N. Binder, G. Marcus, and A. Keller, "Sionna: An open-source library for next-generation physical layer research," *arXiv preprint arXiv:2203.118540*, 2022.

[37] M. Abadi, "Tensorflow: Learning functions at scale," ser. ICFP 2016. New York, NY, USA: Association for Computing Machinery, 2016, p. 1.

[38] C. Xiao, Y. R. Zheng, and N. C. Beaulieu, "Novel sum-of-sinusoids simulation models for rayleigh and rician fading channels," *IEEE Transactions on Wireless Communications*, vol. 5, no. 12, pp. 3667–3679, 2006.