COLLABORATIVE LEARNING FOR LESS ONLINE RETRAINING OF NEURAL RECEIVERS

Tianxin Wang[1], *Shuo Wang*[1], *Xudong Wang*[1,*], *and Geoffrey Ye Li*[2]

[1]Shanghai Jiao Tong University, China
[2]Imperial College London, U.K.

ABSTRACT

Offline-trained neural receivers achieve significant performance gains. Yet, online retraining is required to sustain such gains in a new environment. Instead of retraining whenever a new channel environment arises, a multi-cell collaborative learning framework is designed to enable the neural receivers to generalize to unseen scenarios, thus preventing frequent retraining. This framework features two key designs: 1) the personalized federated learning paradigm is exploited to strike a generalization-personalization balance, with each model sharing a global representation network and personalizing the local head network; 2) an online data filtering mechanism is designed to filter out low-impact data samples. According to simulations, the collaboratively-learned receivers outperform the traditional ones by over 3 dB and improves the generalization performance by 5.2 dB in the unseen scenarios.

Index Terms— Personalized federated learning, neural receiver, online collaboration

1. INTRODUCTION

The sixth-generation (6G) communication networks are expected to be AI-empowered. An increasing amount of research shows that machine learning (ML) plays a significant part on the physical layer of wireless networks [1, 2]. Particularly, the neural-network-based wireless receiver, named **neural receiver**, is one of the most promising solutions for 6G receiver design [2], which features an end-to-end learning and fully data-driven methodology. As shown in Fig. 1, in a neural receiver, the channel estimator and detector are replaced by a neural network (NN) that takes the frequency-domain receive signals as inputs and outputs the soft-detection bits. By implicitly capturing highly-nonlinear channel characteristics [2], neural receivers can significantly outperform the traditional ones.

In this paper, the uplink neural receivers located at base stations (BS) are considered, given the fact that BSs usually have sufficient computing and storage resources [3]. These neural receivers receive the uplink signals from user terminals. As a common practice, a neural receiver is offline pretrained and then deployed online [4]. Nevertheless, the pretrained model may not generalize well in online channel environments, as there exist discrepancies in data distributions between offline and online environments [4, 5]. To this end, online knowledge acquisition for neural receivers becomes a necessity to alleviate online performance degradation.

To address the offline-online discrepancy, some research efforts have been made in [4–7]. In [4], the NN model architectures are redesigned such that it is sufficient to retrain a subset of parameters online. The meta learning algorithms are used in [6, 8] to retrain the neural receiver with fewer training steps. In [5, 7], the retraining performance each time is enhanced by data augmentation. The above studies focus on how to adapt the neural receiver efficiently for a new online environment. However, they cannot reduce the frequency of online retraining when facing with time-varying environments. Specifically, whenever an unseen online environment arises, the neural receiver needs to repeat another round of retraining. This leads to high costs in the long run, as the receiver has to endure performance loss during each retraining [5]. Thus, the key problem lies in how to reduce the frequency of such online retraining.

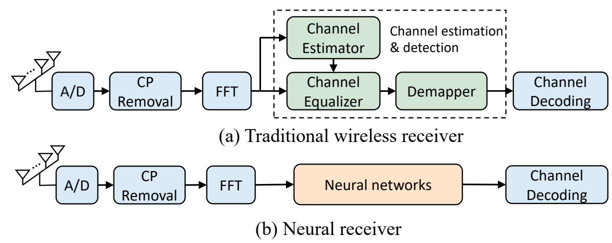


Fig. 1. Uplink neural receiver.

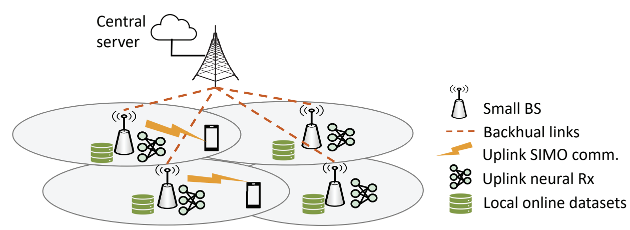


Fig. 2. Multi-cell network with heterogeneous channel scenarios.

To tackle the above problem, the core insight is that if the neural receiver enriches its online knowledge by learning from other cells' data distribution, it can generalize to the locally unseen environments and avoid frequent retraining. Therefore, a **collaborative learning** framework is proposed: each BS collaborates with other BSs to learn its neural receiver, such that the knowledge accumulated in the entire multi-cell network can be exploited. As shown in Fig. 2, in a multi-cell network, multiple small BSs are connected to a central server via backhaul links [9]. In each cell, the BS has multiple antennas for receiving the uplink orthogonal frequency-division multiplexing (OFDM) signals from terminals. The local labeled datasets are built up by the transmission of pilot-based training sequences.

How to fulfill the proposed collaborative learning remains a challenge. As a simple way of collaboration, centralized learning gathers multi-cell data in one server. Yet, it is infeasible practically

Corresponding author: Xudong Wang; Email: wxudong@ieee.org.

due to the privacy concerns [10]: 1) if the BSs belong to various service providers (SP), it violates each SP's privacy policy to expose data to others; 2) even if the same SP regulates all the BSs, the data still contain user-related information, e.g., user trajectories and behavior modes [11]. Therefore, it is more practical for each BS keeping its data locally. This can be realized by conventional federated learning (FL) [12, 13], where a global model is learned with locally-held data. However, due to the data heterogeneity across cells in reality, learning a global model may sacrifice the local performance at each individual BS, which indicates a lack of personalization [14, 15]. This generalization-personalization trade-off needs to be addressed in multi-cell collaboration. In addition, to ease the burden of online data storage locally, it is critical to filter out those pilot samples that have little contribution to collaborative learning. Thus, an online data filtering mechanism must be designed.

To tackle the above challenges, a personalized federated learning (PFL) framework named ***pFedRx*** is designed for collaboratively expanding online knowledge of uplink neural receivers and thus reducing online retraining frequency. It features two designs.

First, to address the generalization-personalization trade-off, the neural receiver is split into two consecutive parts, i.e., the representation network and the local head network, and multiple BSs collaborate to learn a globally-shared representation network and personalized local heads. Specifically, the received uplink pilots over the OFDM resource grid at each antenna can be interpreted as a multi-channel image and representation learning is conducted over this 'image' to extract the common features shared by heterogeneous wireless channels. Based on the shared representation layers, the personalized classification is performed by local head layers via casting the features to binary soft bits. In this way, each BS strikes a balance between learning from others and preserving the local knowledge. Second, the pilot filtering mechanism is designed by defining an impact metric of each online sample and evaluating it on the currently deployed neural receiver. The impact metric is the classification loss weighted by the received SNR. This is because the loss observed in the low-SNR regime is much higher than that in the high-SNR regime. To move this this SNR-induced bias, the received SNR is taken as a weighting factor, such that the data samples in the high-SNR regime can have relatively higher losses. We filter out and discard those data samples with impact metrics below a threshold.

The main contributions of this paper are summarized as follows:

- A PFL-based collaborative learning framework is designed to enrich online knowledge of neural receivers and prevent frequent retraining.
- Within the framework, multiple BSs collaborate with each other to learn a shared representation network and personalized local heads, which strikes the generalization-personalization balance. The online data filtering mechanism reduces the local data storage burden by discarding those less impactful data samples.

The rest of this paper is organized as follows. The system model is provided in Sec. 2. The overall framework is presented in Sec. 3, and the design details are elaborated in three aspects in Sec. 4. The designed scheme is evaluated in Sec. 5, and the conclusions are drawn in Sec. 6.

2. SYSTEM MODEL

In this section, first, the single-input-multiple-output (SIMO) OFDM communication system is characterized. Then, the key elements of the supervised learning problem (i.e., learning a neural receiver) are stated.

2.1. SIMO-OFDM Communications

The uplink SIMO OFDM communication is considered with N_r receive antennas at the BS and one transmit antenna at the user. One OFDM frame spans a transmission time interval (TTI) and includes N_s OFDM symbols and N_f subcarriers. Each resource element (RE) has one symbol time and one subcarrier. Let $\mathbf{Y}, \mathbf{H}, \mathbf{N} \in \mathbb{C}^{N_\mathrm{r}\times N_\mathrm{s}\times N_\mathrm{f}}$ be the received signals, the channel coefficients, the additive white Gaussian noise, respectively. The transmit signal matrix is $\mathbf{X} \in \mathbb{C}^{N_\mathrm{s}\times N_\mathrm{f}}$. Let k be the index of OFDM symbol and l be the index of subcarriers. We define the following vectors $\mathbf{y}_{kl} = \mathbf{Y}[:,k,l], \mathbf{h}_{kl} = \mathbf{H}[:,k,l], \mathbf{n}_{kl} = \mathbf{N}[:,k,l]$ and they all have the dimension of N_r. The frequency-domain received signal on each RE during one TTI is

$$\mathbf{y}_{kl} = \mathbf{h}_{kl}x_{kl} + \mathbf{n}_{kl}, \tag{1}$$

where $x_{kl} \in \mathbf{X}, \forall k = 1, ..., N_\mathrm{s}, j = 1, ..., N_\mathrm{f}$.

2.2. Elements of Supervised Learning Problem

There are M base stations in a multi-cell network. Each BS m builds up an online dataset $\mathcal{D}_m$ locally via the transmission of pilot-based training sequences [2], which is denoted by $\mathcal{D}_m = \{(\mathbf{T}_m^{(i)}, \mathbf{L}_m^{(i)})\}_{i=1}^{N_m}$ with N_m data samples, where $\mathbf{T}_m^{(i)}$ and $\mathbf{L}_m^{(i)}$ are the input and labels of the i-th data sample. Let $\mathbf{p} = [p_1, ..., p_M]$ be the quantity distribution vector, i.e., $p_m = N_m/N$ where N is the total number of data samples. The empirical error function F_m for the neural receiver on BS m over the local dataset is defined as

$$F_m(h_\mathbf{w}) = \frac{1}{N_m}\sum_{i=1}^{N_m} \ell\left(h_\mathbf{w}\left(\mathbf{T}_m^{(i)}\right), \mathbf{L}_m^{(i)}\right), \tag{2}$$

where hypothesis h is parameterized by model parameters $\mathbf{w}$, and $\ell(\cdot)$ is the loss function. For ease of notation, we use $F_m(\mathbf{w}) = F_m(h_\mathbf{w})$. The detailed realizations of $\mathbf{T}_m, \mathbf{L}_m$, and $\ell(\cdot)$ will be stated in Sec. 4.1.

3. OVERALL FRAMEWORK OF COLLABORATIVE LEARNING

During collaborative learning, the working procedure of each BS is illustrated in Fig. 3. Each BS optimizes its neural receiver locally based on its online dataset. Specifically, the local online dataset is obtained via the periodic transmission of pilot sequences within a fixed time interval named as collaboration period. The periodically received pilot samples are filtered and stored. The collaboration period has a large timescale as its purpose is to enrich online knowledge on the long term and to prevent frequent retraining. The learning phases on two different timescales are elaborated as follows.

On a large timescale, multiple BSs participate in the collaborative learning process based on the online local datasets. To start with, the neural receiver at each BS holds an initial model $\mathbf{w}_m$ consisting of two parts: the representation network $\boldsymbol{\theta}_m$ and the local head $\boldsymbol{\phi}_m$, i.e., $\mathbf{w}_m = \{\boldsymbol{\theta}_m, \boldsymbol{\phi}_m\}$. The initial model can be the offline-pretrained model, or the collaborative model from the last collaboration period. As a result of collaborative learning, a globally shared representation network $\boldsymbol{\theta}^\mathrm{new}$ is obtained that extracts the common features across multiple cells, and each BS learns a personalized head $\boldsymbol{\phi}_m^\mathrm{new}$ to capture the local distribution. Thus, the learned

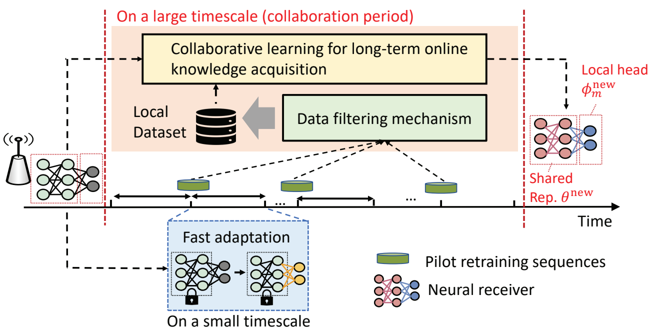


Fig. 3. pFedRx: working process of each BS in collaboration.

collaborative models $\mathbf{w}_m^{\text{new}} = \{\boldsymbol{\theta}^{\text{new}}, \boldsymbol{\phi}_m^{\text{new}}\}, \forall m = 1, ..., M$ are deployed into each BS's neural receiver.

On a small timescale, fast adaptation can be conducted for multiple times based on the most recent pilot frames during the collaboration period. This adapts the neural receiver to the current channel. It is realized by updating the local heads for a few gradient descent steps while keeping the representation network fixed. Note that other efficient online adaptation methods in [5, 7] can be used in combination with collaborative learning, which is not the focus of this paper.

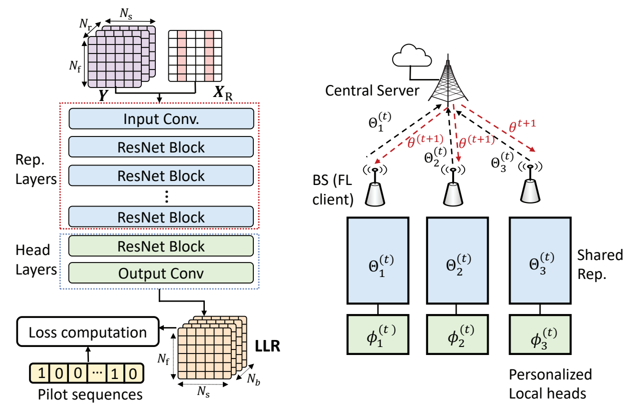
It is stressed that the collaborative learning process is in parallel with real-time model inference and fast adaptation. Collaborative learning is a long-term online task [3] running in the background. The generalization performance of neural receivers can be significantly improved after collaborative learning, as the multi-cell online knowledge is well learned.

4. COLLABORATIVE LEARNING FOR MULTI-CELL NEURAL RECEIVERS

In this section, we first present the design details of neural receivers. Then, the PFL algorithm for multi-cell collaborative learning is elaborated, followed by the mechanism design of online data filtering.

4.1. Design of Neural Receiver

As shown in Fig. 4(a), the input of neural receiver consists of two parts: received OFDM frame $\mathbf{Y} \in \mathbb{C}^{N_\text{r} \times N_\text{s} \times N_\text{f}}$ and demodulation reference symbols (DMRS) $\mathbf{X}_\text{R} \in \mathbb{C}^{N_\text{s} \times N_\text{f}}$. The configuration of $\mathbf{X}_\text{R}$ is aligned with third generation partnership project (3GPP) standards. It contains zeros at non-DMRS positions as in [2]. With these two parts stacked, we have $\mathbf{T}_m \in \mathbb{R}^{(2N_\text{r}+2) \times N_\text{s} \times N_\text{f}}$ (omitting the data sample index) by treating their real and imaginary parts as two separate channels. The output $h_\mathbf{w}(\mathbf{T}_m) \in \mathbb{C}^{N_\text{s} \times N_\text{f} \times N_\text{b}}$ is the log-likelihood ratios (LLR) to be fed into channel decoders, where N_b is the number of bits per symbol. The labels are the transmitted bit sequence of length $n_\text{t} \cdot N_\text{b}$ with n_t as the number of transmitted symbols, i.e., $\mathbf{L}_m \in \mathbb{R}^{n_\text{t} \cdot N_\text{b}}$ (omitting the data sample index). Note that the online labels are required for supervised retraining. There are typically two ways of retrieving labels [5]: 1) pilot-based sequence transmission (where $\mathbf{L}_m$ is known by the receiver); 2) on-the-fly label recovery [7]. Although the pilot transmission incurs a data rate loss, the label recovery method can be easily impacted by the erroneous labels. In this paper, we adopt the former method but our design can be easily extended to the later one. At last, the soft bit detection task is cast as a binary classification problem. The binary sigmoid cross-entropy loss is used to compute the distance between model outputs and labels as in [2].



(a) Fully convolutional arch.   (b) PFL method

Fig. 4. Model architecture of a neural receiver and multi-cell collaborative learning.

For the model architecture of the neural receiver, a fully convolutional neural network (CNN) consisting of multiple preactivation ResNet blocks is employed as in [2]. More details can be referred in [2]. As shown in Fig. 4(a), CNN model $\mathbf{w}_m$ at each BS m is partitioned into two parts before collaboration: representation layers $\boldsymbol{\theta}_m$ and local head layers $\boldsymbol{\phi}_m$. Based on the studies in representation learning [16], the early convolutional layers correspond to the representation shared across different tasks while the last few convolution layers tend to be task-specific. In Sec. 5.2, we will show that the early convolutional layers indeed learn the similar representations and the best partitioning will be decided through empirical studies.

4.2. Personalized Federated Learning Algorithm for Multi-Cell Neural Receivers

As depicted in Fig. 4(b), model $\mathbf{w}_m$ at BS m is partitioned into two parts, representation layers $\boldsymbol{\theta}_m$ and local head layers $\boldsymbol{\phi}_m$. During collaborative learning, multiple BSs collaborate to learn the global representation layers $\boldsymbol{\theta}$ and to personalize the local classifier layers $\boldsymbol{\phi}_m, \forall m = 1, ..., M$. In this way, each neural receiver captures the common knowledge shared by the multi-cell network as well as the local knowledge. The optimization formulation of this collaborative learning problem is expressed as follows:

$$\min_{\boldsymbol{\theta}, \{\boldsymbol{\phi}_m\}_{m=1}^M} \sum_{m=1}^{M} p_m F_m(\boldsymbol{\theta}, \boldsymbol{\phi}_m), \tag{3}$$

where the empirical function $F(\cdot)$ (defined in Sec. 2) is weighted by the quantity distribution vector $\mathbf{p} = [p_1, ..., p_M]$.

To solve the above optimization problem, the PFL algorithm FedRep [14] is adapted to our multi-cell collaboration problem with two distinct features. First, the model splitting in [14] has a straightforward pattern, i.e., separating convolutional layers from fully-connected layers, while different splitting cases must be explored for fully convolutional models in neural receivers. Second, the number of local iterations is increased significantly to reduce the backhaul communication costs between the BSs and the central server, as the backhual links can be wireless and use the same radio resources as the access network. Before diving in the algorithm, the batch gradient descent operation is defined as $g(F_m(\boldsymbol{\psi}, \boldsymbol{\gamma}), \boldsymbol{\psi}, \eta, K)$, which means conducting K batch gradient descents on the model parameters $\boldsymbol{\psi}$ with the objective $F_m(\boldsymbol{\psi}, \boldsymbol{\gamma})$ and the learning rate η.

Algorithm 1 pFedRx: Collaborative learning of neural receivers based on personalized federated learning

Require: Initial models $\boldsymbol{\theta}^{(0)}, \{\boldsymbol{\phi}_m^{(0)}\}_{m=1}^M$, number of communication rounds T, number of local iterations K_p for the personalized head, number of local iterations K_r for the shared representation, set of BSs $\mathcal{S} = \{1, 2, ..., M\}$.

1: **for** $t = 0$ **to** $T-1$ **do**
2: The server broadcasts the shared representation $\boldsymbol{\theta}^{(t)}$ to each BS in $\mathcal{S}$,
3: **for** each BS $m \in \mathcal{S}$ in parallel **do**
4: Set $\boldsymbol{\theta}_m^{(t)} = \boldsymbol{\theta}^{(t)}$,
5: $\boldsymbol{\phi}_m^{(t+1)} \leftarrow g\left(F_m(\boldsymbol{\theta}_m^{(t)}, \boldsymbol{\phi}_m^{(t)}), \boldsymbol{\phi}_m^{(t)}, \eta_\mathrm{p}, K_\mathrm{p}\right)$,
6: $\boldsymbol{\theta}_m^{(t+1)} \leftarrow g\left(F_m(\boldsymbol{\theta}_m^{(t)}, \boldsymbol{\phi}_m^{(t+1)}), \boldsymbol{\theta}_m^{(t)}, \eta_\mathrm{r}, K_\mathrm{r}\right)$,
7: BS m sends $\boldsymbol{\theta}_m^{(t+1)}$ back to the server,
8: **end for**
9: The server conducts aggregation on the representation layers $\boldsymbol{\theta}^{(t+1)} = \sum_{m\in\mathcal{S}} p_m \boldsymbol{\theta}_m^{(t+1)}$,
10: **end for**
11: Each BS m learns a personalized neural receiver $\mathbf{w}_m = \{\boldsymbol{\theta}^{(T-1)}, \boldsymbol{\phi}_m^{(T-1)}\}$.

Especially, the one-step gradient descent is explicitly expressed as

$$g(F_m(\boldsymbol{\psi}, \boldsymbol{\gamma}), \boldsymbol{\psi}, \eta, 1) = \boldsymbol{\psi} - \eta \nabla_{\boldsymbol{\psi}} F_m(\boldsymbol{\psi}, \boldsymbol{\gamma}). \tag{4}$$

The algorithm of pFedRx is elaborated in Alg. 1. Initial models $\mathbf{w}_m = \{\boldsymbol{\theta}_m^{(0)}, \boldsymbol{\phi}_m^{(0)}\}, \forall m = 1, ..., M$ are deployed at the BSs and the initial shared representation network is $\boldsymbol{\theta}^{(0)}$. At the beginning of each round t, the server broadcasts last-round shared representation $\boldsymbol{\theta}^{(t)}$ to each BS m, which is used to initialize all the representation layers of neural receivers (i.e., $\boldsymbol{\theta}_m^{(t)} = \boldsymbol{\theta}^{(t)}$). For local training, each BS first conducts batch gradient descent on personalized local head $\boldsymbol{\phi}_m^{(t)}$ for K_p iterations while keeping the representation layers fixed, and obtains $\boldsymbol{\phi}_m^{(t+1)}$. Then, each BS conducts batch gradient descent on $\boldsymbol{\theta}_m^{(t)}$ for K_r iterations, with learned personalized head $\boldsymbol{\phi}_m^{(t+1)}$ fixed. At the end of local training, each BS sends their locally updated representation $\boldsymbol{\theta}_m^{(t+1)}$ back to the central server. Such T communication rounds of training are performed until each BS m obtains a well-learned personalized neural receiver with multi-cell collaborative knowledge, i.e., $\mathbf{w}_m = \{\boldsymbol{\theta}^{(T-1)}, \boldsymbol{\phi}_m^{(T-1)}\}$.

4.3. Online Data Filtering Mechanism

As mentioned in Sec. 3 and 2, each BS constructs a local dataset for collaborative learning via the periodic transmission of pilot sequences. If all these received training data are saved, the storage redundancy will be incurred, as not all the data samples contain useful information for learning new environments. Some of the data samples can be more informative and representative than others. Therefore, to alleviate the above redundancy, an online data filtering mechanism is designed by defining an impact metric of each online sample and evaluating it on the currently deployed neural receiver.

The impact metric is the classification loss of one sample weighted by the achievable rate under the received SNR. This is because SNR induces a bias in training data samples, as the classification loss observed in the low-SNR regime is much higher than that in the high-SNR regime. To remove this bias, a weighting factor related to SNR is taken into account. Let $\alpha^{(i)}$ be the impact metric of the i-th data sample on the current model $\mathbf{w}$. Then we have

$$\alpha^{(i)}(\mathbf{w}) = \log_2\left(1 + \gamma^{(i)}\right) \ell\left(h_\mathbf{w}\left(\mathbf{T}_m^{(i)}\right), \mathbf{L}_m^{(i)}\right), \tag{5}$$

where $\gamma^{(i)}$ is the SNR for the i-th sample and the loss is weighted by the achievable Shannon rate per unit bandwidth. At each BS m, with the pre-set threshold α_th, those online samples that have $\alpha^{(i)}(\mathbf{w}_m) \geq \alpha_\mathrm{th}$ are selected and saved locally while the others are filtered out.

Table 1. Heterogeneous multi-cell environments

Cell ID	Channel model	Delay spread	Velocity range
1	TDL-B/C	0-50 ns	15-20 m/s
2	TDL-B/C	400-500 ns	0-5 m/s
3	TDL-B/C	200-300 ns	15-20 m/s
4	TDL-B/C	400-500 ns	15-20 m/s
5	TDL-D/E	0-50 ns	0-5 m/s
6	TDL-D/E	400-500 ns	15-20 m/s

5. PERFORMANCE EVALUATION

In this section, our simulation set-up is first presented. Then, the simulation results of the ablation studies and the comparisons with several baselines are stated.

5.1. Simulation Set-up

For SIMO communication, we assume two receive antennas at the BS without loss of generality, and one antenna at the user. The designs can be extended to SIMO systems with more receive antennas. There are 6 BSs in the multi-cell collaboration. We assume heterogeneous channel environments across cells. The channel models adopt the 3GPP tapped delay line (TDL) channel models [17], i.e., non-line-of-sight scenarios (TDL-A/B/C) and line-of-sight scenarios (TDL-D/E). To showcase the offline-online distribution discrepancy, the neural receiver is pretrained based on the TDL-A channel model at a delay spread of 0-50ns and on a velocity range of 0-5m/s. The online heterogeneous channel environments in six cells are shown in Tab. 1, and the uplink channel realizations are randomly generated in each cell.

For the periodic pilot transmission, one pilot-carrying OFDM frame (1 ms) is transmitted every 128 data-carrying OFDM frames, which incurs around 0.8% pilot overheads. Note that the online label recovery method that directly uses data frames for training in [7] can be used to further reduce the pilot overheads. Each BS builds up a local dataset with 1000 batches of pilot frames (32 samples per batch), which corresponds to a collaboration period of 1.15 hours. In other words, after 1.15-hour online deployment of neural receivers, multiple BSs collaboratively retrain their neural receivers with multi-cell channel knowledge. The critical simulation parameters are as follows. For uplink communications, the carrier frequency is 4 GHz with the subcarrier spacing 15 kHz; the modulation scheme is 16-QAM; the 5G LDPC channel coding is adopted with the code rate as $658/1024$; $N_\mathrm{f} = 72, N_\mathrm{s} = 14, n_t = 720$. For model learning, the Adam optimizer with 0.001 learning rate is used, and 80% of the local datasets are used for training while the remaining 20% are for validation and testing. The PFL parameters are $T = 20, K_\mathrm{r} = 2K_\mathrm{p} = 1600$.

The metric for evaluating receiver performance is coded bit-error rate (BER). Our designed method pFedRx is compared with

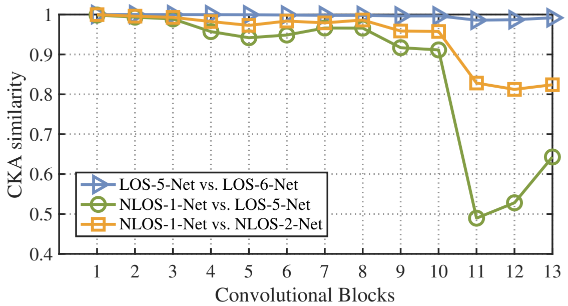
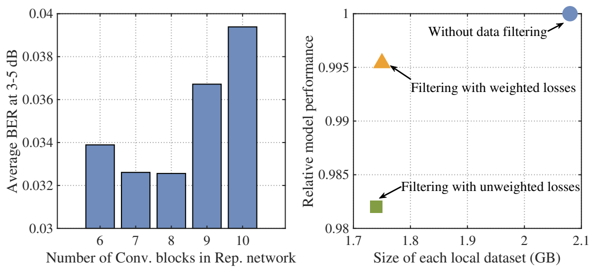


Fig. 5. Representation comparison based on CKA similarity.



(a) Size of Rep. network (b) Local storage

Fig. 6. Ablation studies for pFedRx.

the following baselines: 1) practical LMMSE receiver [2]; 2) genie-aided LMMSE receiver (with full and exact channel coefficients); 3) locally-trained neural receiver; 4) FedAvg-trained global neural receiver [12]; 5) Ditto-trained [15] personalized receiver. Ditto is one of the most effective PFL algorithms. The performance of the genie-aided LMMSE receiver is considered as the upper bound in interference-free cases.

5.2. Ablation Study

To justify the design of a shared representation network among cells, we compare the centered kernel alignment (CKA) similarity index [16] between the models trained with heterogeneous channel environments. The CKA similarity is widely used to compare representations learned by different trained models and their layers. The neural receiver learned in pFedRx has one input convolutional layer, 11 ResNet blocks, and one input convolutional layer, i.e., 13 convolutional blocks in total [2]. The models trained in cell 1 and 2 are named NLOS-1-Net and NLOS-2-Net, and the models trained in cell 5 and 6 are named LOS-5-Net and LOS-6-Net. As shown in Fig. 5, NLOS-1-Net and LOS-5-Net develop similar representations in their early layers while in the last few convolutional blocks the learned representations begin to diverge. This similar pattern also lies in the comparison of NLOS-1-Net and NLOS-2-Net.

By varying the number of convolutional blocks in the shared representation network, the best way of the model splitting is studied. We evaluate pFedRx with various model splitting schemes as shown in Fig. 6(a) and the average BER between 3 and 5 dB is computed. When there are 8 convolutional blocks in the shared representation (and 5 convolutional blocks in each personalized head), the designed algorithm achieves the best performance. The effectiveness of the online data filtering mechanism is evaluated in Fig. 6(b). The relative model performance (average BER at 6dB) against the unfiltered case is presented. With data filtering based on weighted losses, pFedRx achieves similar performance as the unfiltered scheme but

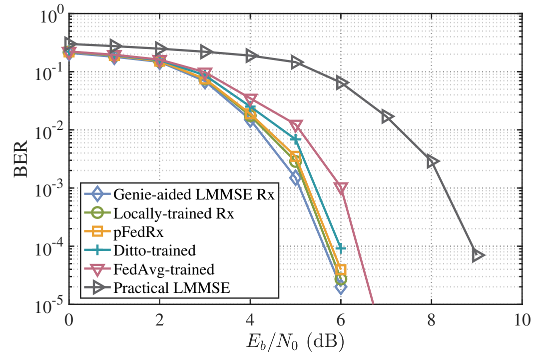


Fig. 7. Local performance of neural receivers.

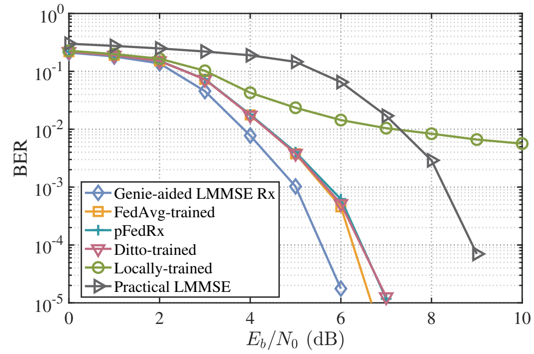


Fig. 8. Generalization performance of neural receivers.

reduces the local storage by 16.8%.

5.3. Comparison with Other Baselines

In the local test of neural receivers (i.e., in-distribution test), the learned neural receiver at each BS is evaluated with its local data distribution and the average BER across 6 cells is presented. As different cells have heterogeneous channel environments, local learning is focused on fitting the local data distribution, and thus it can outperform others in the local tests with sufficient online data, as shown in Fig. 7. In this case, our method pFedRx achieves approximately the same BER performance as the locally-trained neural receiver, both approaching the upper bound. The pFedRx scheme also achieves 0.7-1 dB gain over FedAvg and 3-3.2 dB gain over the practical LMMSE receiver.

In the generalization test of neural receivers (i.e., out-of-distribution test), the learned neural receiver at each BS is evaluated with a locally unseen data distribution (yet seen in other cells). Due to the lack of collaborative knowledge form other cells, the locally-trained neural receiver cannot generalize to unseen distributions with a significant BER drop. For the collaborative learning case, all three FL schemes achieve notable generalization performance, with the BER gains of 5.2 dB (at BER = 0.004) compared to the locally-trained receiver. Our method pFedRx slightly outperforms other FL schemes by 0.3 dB in the high-SNR regime. Based on two type of tests, pFedRx strikes the generalization-personalization trade-off: the trained neural receiver retains high local performance and generalizes well to unseen scenarios at the same time.

With the improved generalization ability, the online retraining frequency is significantly reduced by deploying the trained neural receivers in pFedRx. Considering a neural receiver in one cell, we assume the online channel environment is time-varying with 15

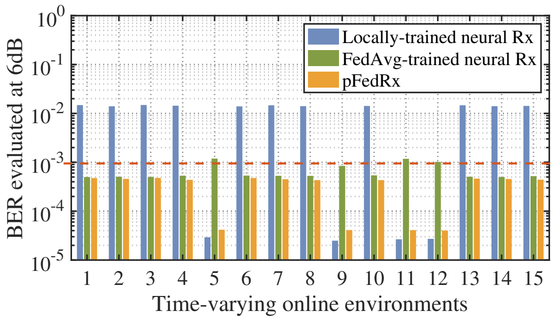


Fig. 9. Online BER in time-varying channel environments.

randomly generated configurations. Their distributions are either learned before or never seen. We assume that online retraining is conducted if the detected BER is lower than 10^{-3}. As shown in Fig. 9, the locally-trained receiver requires 11 times of retraining among 15 environments while the receiver in pFedRx does not require any retraining, as its online knowledge is enriched by the multi-cell collaboration. The FedAvg scheme requires retraining twice throughout the environmental changes. Also, the pFedRx scheme achieves a lower BER than the FedAvg scheme across all the environments.

6. CONCLUSION

A multi-cell collaborative learning framework (pFedRx) was designed for enriching online knowledge and preventing frequent retraining of neural receivers. Via the design of PFL paradigm, the neural receivers retained high local performance and achieved notable generalization to unseen scenarios as well. This is the first PFL framework designed for neural receivers. The storage burden at each BS was alleviated by the online data filtering mechanism. For future studies, it remains a challenge to adaptively adjust the model splitting of pFedRx. How to further reduce the pilot overheads and computation costs of collaborating learning is to be investigated.

7. REFERENCES

[1] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wirel. Commun. Lett.*, vol. 7, no. 1, pp. 114–117, 2017.

[2] M. Honkala, D. Korpi, and J. M. Huttunen, "DeepRx: Fully convolutional deep learning receiver," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 6, pp. 3925–3940, 2021.

[3] M. Polese, L. Bonati, S. D'oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges," *IEEE Commun. Surv. Tutor.*, vol. 25, no. 2, pp. 1376–1411, 2023.

[4] P. Jiang, T. Wang, B. Han, X. Gao, J. Zhang, C.-K. Wen, S. Jin, and G. Y. Li, "AI-aided online adaptive OFDM receiver: Design and experimental results," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 11, pp. 7655–7668, 2021.

[5] M. B. Fischer, S. Dörner, F. Krieg, S. Cammerer, and S. ten Brink, "Adaptive NN-based OFDM receivers: Computational complexity vs. achievable performance," in *IEEE Asilomar Conf. Signals Syst. Comput.*, 2022, pp. 194–199.

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

[7] M. B. Fischer, S. Dörner, S. Cammerer, T. Shimizu, H. Lu, and S. Ten Brink, "Adaptive neural network-based OFDM receivers," in *IEEE Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, 2022, pp. 1–5.

[8] O. Wang, J. Gao, and G. Y. Li, "Learn to adapt to new environments from past experience and few pilot blocks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 9, no. 2, pp. 373–385, 2022.

[9] T. Wang, S. Chen, Y. Zhu, A. Tang, and X. Wang, "Linkslice: Fine-grained network slice enforcement based on deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 8, pp. 2378–2394, 2022.

[10] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46–51, 2020.

[11] Y. Cui, J. Guo, C.-K. Wen, and S. Jin, "Communication-efficient personalized federated edge learning for massive mimo csi feedback," *IEEE Trans. Wirel. Commun.*, 2023.

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artif. Intell. Stat.*, 2017, pp. 1273–1282.

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

[14] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 2089–2099.

[15] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 6357–6368.

[16] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 3519–3529.

[17] 3rd Generation Partnership Project (3GPP), "Study on channel model for frequencies from 0.5 to 100 GHz," TS 38.901, 2024, v17.1.0.