Federated Learning for Radar Gesture Recognition Based on Spike Timing Dependent Plasticity

Mengjun Zhang, Bin Li, Hongfu Liu, Chenglin Zhao
Beijing University of Posts and Telecommunications, Beijing, China

***Abstract*— Radar-based gesture recognition combined machine learning methods can achieve excellent performance and has been utilized in a wide variety of applications. However, large-scale radar signal processing suffers from the risk of privacy leakage and the problem of limited computing resources. In this paper, we propose a federated learning framework based on the spiking neural network to solve above limitations in radar gesture recognition. First, we build a distributed federated learning system with good privacy protection to process radar data collaboratively, which only exchanges the model parameters between clients to train the model, without the need for transferring the data itself. Second, we train the spiking neural network using the spike timing dependent plasticity with biological interpretability, which is more compatible with the biological characteristics and can significantly reduce the energy consumption. Our creative combination of low-power spiking neural network and federated learning thus address the issue of resource constraint of edge nodes in federated learning. Furthermore, we introduce the weight pruning technique to the federated learning training process to reduce the model's communication cost. We conduct experiments on the 8GHz radar gesture dataset and Google Soli dataset to test the proposed model's validity. We evaluate the model's performance across various participating devices and examine the spiking neural network's capacity to recognize gestures under independent identical distribution and non-independent identical distribution strategies in federated learning. We also analyze the energy consumption and communication cost of the proposed model, demonstrating the spike timing dependent plasticity-based algorithm is more energy efficient than the traditional backpropagation as well as the backpropagation through time algorithms.**

***Index Terms*— Radar gesture recognition, Federated learning, Spiking neural network, Spike timing dependent plasticity, Dada security.**

I. Introduction

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants U1805262 and U61971050.

Mengjun Zhang, Bin Li, Hongfu Liu and Chenglin Zhao are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, 100876, China (e-mail: Binli@bupt.edu.cn).

0018-9251 © 2022 IEEE

IN recent years, radar gesture recognition has attracted much attention, which is of great importance in the fields of security surveillance, human-computer interface, safe driving, medical health and sports and fitness [1], [2]. Instead of capturing video image information like traditional optical or other sensors, radar identifies and classifies the human behaviors through the Doppler frequencies, requiring less storage space and information processing capability. More important, radar echo is not susceptible to light conditions and occlusions when collecting signal due to its strong penetrability and weak environmental sensitivity [3]. Li et al. successfully applied radar technology to public security monitoring by automatically recognizing human posture through real-time analysis of human echo [4]. The human posture detection technology was also used in the passive fall detection of the elderly, which combined body posture and motion information to provide timely warning of potentially dangerous situations [5], [6]. In the intelligent transportation, the radar was used in combination with high-definition cameras as well as other equipments to measure the speed of vehicles. In addition, human posture recognition based on radar sensors also played an important role in disaster rescue, which can detect human movements behind obstacles such as walls [7], [8].

In real scenarios, the number of radar deployments is large and the data collected by radar is massive and fast. Therefore, the human posture recognition technology based on radar sensor cannot be separated from the support of deep learning model. Kim et al. used continuous wave radar to extract Doppler sign and trained a supported vector machine (SVM) to identify seven rhythmic human movements [9]. Li et al. creatively combined the sparsity-aware feature extraction method with the modified-Hausdorff-distance-based classifier, which extracted the micro-Doppler features using the orthogonal matching pursuit algorithm and then recognized the hand gestures by the nearest neighbor classifier [10]. Hong et al. extracted the dynamic range Doppler trajectories from the radar gesture signal and separated the human movement by a peak search method, which then combined with the machine learning methods to classify the human motion [11], [12]. The deep convolutional autoencoder (CAE) was proposed in [13] to distinguish human activities, which was shown to be more effective than other deep learning architectures. The deep convolutional neural network (DCNN) was applied for human activity classification based on micro-Doppler features, which directly took the spectrogram as the input and achieved a good classification performance [14], [15], [16]. The unique ultrasound system was proposed for person identification, which used CNN [17] or LSTM [18] to recognize human posture. In [19], only micro-Doppler signatures is used to investigate the feasibility of recognizing hand gestures with no range information is utilized, which investigated the feasibility of classifying them based on the measured signatures. Bryan et al. used the principal component analysis (PCA) to extract feature vectors from the radar echo signal, and then used the SVM classification algorithm to distinguish human postures successfully [20]. Kern et al. in [21] extracted the perframe feature from radar signal with PointNet and identified the traffic gestures with long short-term memory (LSTM), whose performance clearly outperformed the counterpart CNN.

Although previous studies have achieved significant advancements, it still have significant limitations. On the one hand, gesture recognition requires the aggregation of large-scale data, which can easily lead to the leakage of data privacy and pose serious security risks. On the other hand, the widely used DNN-based recognition methods require huge memory footprint and high energy consumption, making them unsuitable for ultra-low-power IoT devices. A distributed federated learning (FL) system can solve the privacy leakage problem when processing radar data on a large scale. In FL, client devices do not need to exchange data with one another. Instead, they train models independently at the edge and then upload the parameters to the central server for aggregation [22], [23]. In addition, parallel processing across multiple workers in FL can also accelerate the convergence and compensate the infeasibility of training enormous scale data on a single model [24]. However, the widely used recognition methods require huge memory footprint and high energy consumption, which may result in resource limitations at the edge. Furthermore, FL is still vulnerable to security assaults such as the gradient reconstruction attack, which offer certain security problems.

The spiking neural network (SNN), which delivers information via spike trains with certain frequencies, can effectively address the energy consumption issue in radar gesture detection. As the third generation neural network, SNN is inspired by the electrical activity in human brain [25]. Instead of activating the neuron in every iteration propagation, the neuron in SNN can only be activated when the membrane potential reaches a certain value, so the energy consumption and computational cost can be reduced effectively [26], [27], [28]. Currently, few researches have focused on using SNN to solve the radar classification problem. The SNN with convolutional structure was proposed for radar gesture recognition, which learned the network with a surrogate gradient method called backpropagation through time (BPTT) and reduced the power consumption significantly [29], [30]. Arsalan et al. proposed an energy efficient SNN-based radar gesture recognition system, which processed raw ADC data in end-to-end way and avoided the cost of FFTs [31]. The range-Doppler radar signal was obtained through preprocessed in [32], after that a signal to spike approach was used to encode the Doppler map into spike train and fed to SNN for analysis, whose readout was recognized by several classifiers. However, the training algorithms in SNN are not mature, which are mainly based on traditional BP algorithm for gradient approximation or network conversion, with lower recognition accuracy than traditional DNN model. Therefore, in order to achieve high performance in SNN-based radar recognition, additional classification algorithms such as SVM, random forest, and logistic regression are added [32], which is not conform to the inherent biological characteristics in SNNIn comparison, the spiking time dependent plasticity (STDP) rule, which is inspired by brain learning patterns and has good biological interpretability, has significant potential for SNN training. The STDP can modify the synaptic depending to the firing time of adjacent neurons, which does not require a whole inference process before computing the weight and eliminates gradient backpropagation to some extent [33]. Accordingly, it is meaningful to use the STDP rule based on biological interpretability in radar recognition scenarios.

When the model structure in FL is complex, its transmission between the central server and the client also requires significant communication costs. Based on this, neural network pruning can be used to address the energy consumption in resource-constrained nodes, which compresses the model by removing unnecessary weights from the network at the cost of only a marginal loss in accuracy [34]. Magnitude-based weight pruning technology was widely used in recent years, which took the magnitude of the weights as a criterion for the importance of the model connection and removed the weight with the smallest value to achieve network sparsity [35]. Han et al. used the iterative pruning to remove the redundant connections in deep neural networks and achieved the impressive results, whose idea was applied to a variety of implementations [36], [37]. Instead of the traditional training-and-iterating idea, the lottery ticket hypothesis (LTH) proposed in [38] stated that there exists an optimal substructure in the original model, whose direct training can achieve or even exceed the performance of the original model. However, the weight pruning methods are mainly used in traditional DNN networks, which are rarely used in SNNs due to the non-differentiability. Hence, the utilization of SNN-based pruning techniques in distributed FL framework holds great significance.

In this paper, we explore for the first time the collaborative training of FL with STDP to achieve the radar gesture recognition. The main contributions are as follows:

1) We propose an energy-efficient and privacy-preserving FL method for radar gesture recognition. Instead of exchanging the raw data of each radar, FL can learn the network collaboratively by sharing only the client model parameters, which protects the data. More importantly, we use SNN as the client model in FL, which can significantly reduce energy consumption by transmitting information through discrete spikes;
2) We present a biologically interpretable FL learning approach that employs STDP to train the SNN on the client-side of FL. This method updates the network weights based on the temporal correlation of the spike releases from neighboring neurons

within the SNN. Unlike the traditional BP algorithm, the STDP-based approach does not require back-propagation of gradients, which can lower neural network energy consumption and eliminate gradient attacks against FL;

3) We design a novel FL approach with distributed parameter pruning. We concentrate on random pruning, magnitude-based iterative pruning, and LTH, and employ these pruning strategies in SNN for the first time. By combining several pruning methods with the STDP rule, a sub-network that approximates the original network can be found and used for model parameter transmission, substantially reducing the communication cost of FL;
4) We conduct experiments to analyze the performance of the proposed model. By setting up several control experiments to observe the experimental results under different FL configurations, we demonstrate that the STDP-based FL has good accuracy and robustness. We analyze the computational complexity as well as the security of the proposed model using the BP-based Lenet and BPTT-based SNN models as benchmarks, and the experimental results show that our approach can reduce the energy consumption and effective against gradient-based model attacks. Furthermore, we examine the models' performance under various pruning approaches, indicating that the LTH strategy can further reduce network energy usage and model communication costs without sacrificing accuracy.

The rest of the paper is organized as follows. Section II introduces the fundamental theory of the radar system and the SNN. Section III presents the specific process of our proposed model. In Section IV, numerical experiments are carried out and the conclusion is drawn in Section V.

II. Methodology

In this section, we first introduce the basic theory of radar. The FL theory and its model updating mechanism are then briefly described. Afterwards, FL's client model SNN and its biologically interpretable learning mechanism are shown in detail. Finally, several pruning methods used in the article are outlined.

A. Radar System Theory

In this paper, the human gesture data is collected by the FMCW radar, which can continuously transmit chirp signals to measure the distance, angle and speed of the target. The structure of the FMCW radar system is demonstrated in Fig. 1. When detecting the target, the chirp signal is generated by synthesizer and emitted from the transmitting antenna(TX), which can be modeled as

$$s(t) = sin(2\pi f_c t + \pi\frac{B}{T}t^2) \tag{1}$$

where $\phi_t(t)$ can be defined as

$$\phi_t(t) = 2\pi f_c t + \pi\frac{B}{T}t^2 \tag{2}$$

In Eq. 2, f_c is the start frequency of chirp, which increases with slope $\frac{B}{T}$ over a certain time scale T. At time t, the instantaneous frequency of emitted signal can be expressed as

$$f_t(t) = f_c + \frac{B}{T}t \tag{3}$$

where B is the bandwidth of the chirp. The radar signal will be bounced when it reaches the target object, which will result an echo signal

$$\begin{aligned} s_r(t) &= \xi s(t-\tau) \\ &= \xi sin(2\pi f_c(t-\tau) + \pi\frac{B}{T}(t-\tau)^2) \end{aligned} \tag{4}$$

where ξ denotes the attenuation coefficient. The time delay between the transmitted and received signals can be demonstrated as $\tau = \frac{2d}{c}$, where d is the target from object to radar and c is the speed of light. By mixing the received signal $s_r(t)$ with a replica of $s(t)$ at the in-phase receiver of the radar chip, the IF signal with a new frequency is obtained as follow:

$$\begin{aligned} r(t) &= s(t)s_r(t) = \xi sin(\phi_t(t-\tau))sin(\phi_t(t)) \\ &= -\frac{\xi}{2}cos(2\pi\frac{B}{T}\tau t + 2\pi f_c\tau - \pi\frac{B}{T}\tau^2) \end{aligned} \tag{5}$$

The phase offset between transmitted and received signals is related to the echo delay τ, which is $\Delta\phi = 2\pi f_c\tau = \frac{4\pi d}{\lambda}$ and λ is the wavelength. For multiple targets, the RX will receive several echo signals, which have different delay and corresponding to respective frequency variance. In this case, the IF signal is a superposition of multiple single frequency signals, which Fourier transform can produce a spectrum with different peaks and each of them represents the target at a specific distance. Thus, the distance to the target can be calculated according to the frequency of each peak. For each frame of signal, the data is a matrix whose rows represent the fast time and columns represent the slow time. The range-Doppler frame can be obtained by an FFT on the columns of each frame [39]. By apply the Fourier transform on radar signal after adding a sliding window, i.e., short-time Fourier transform (STFT), we can capture the motion of the human body.

B. Federated Learning

FL can achieve the balance between data privacy protection and data sharing computing. As a distributed learning framework, FL carries out model training by multiple data terminals with local data, and it only requires to exchange model parameters without exchanging local individuals or sample data [40]. Federated learning system consists of N clients and a central server. Each of the client $\mathcal{N}_n, n = 1, 2, \ldots, N$ holds a local data set $\mathcal{D}_n$, which is allocated from the model data set randomly according to the storage and the computation capacity of the client. At the beginning of the training, the central

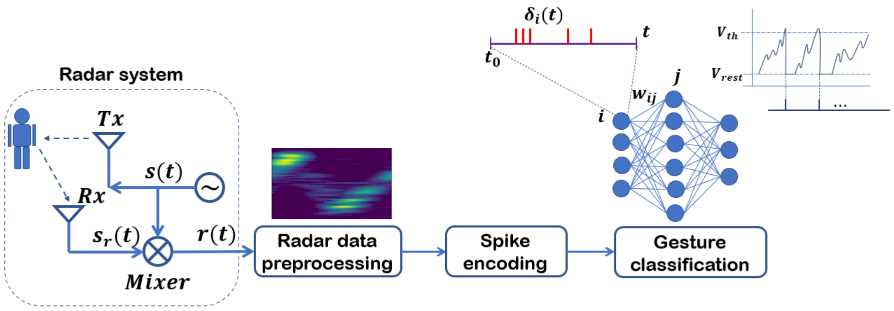

Fig. 1. The illustration of radar gesture recognition

server initializes the model uniformly and broadcasts the model M_0 to all clients. Based on this, the client updates parameters according to their local data set $\mathcal{D}_n$ and obtains a locally model M_n, which are uploaded to the central server for aggregation. More specifically, in each communication round r, a subset of clients $\mathcal{N}_{n_r}(n_r \subseteq N)$ are selected at random to participate the training of the model. After local training, the participating clients receive their updated model $\Delta M_c^r(c \in n_r)$, then the updated weights are aggregated in the central server [41], as shown in

$$\Delta M_r = \frac{1}{|n_r|}\sum_{c\in n_r}\Delta M_c^r \tag{6}$$

where $|n_r|$ demonstrates the total number of clients participating in training at round r. After that, the central server sends the new parameters to clients to start the next round training.

C. Spiking Neural Networks

As an effective method for complex spatiotemporal information processing, SNN uses the spiking neural model as the basic unit and transfers signal with precisely spike sequence.

The Integrate and Fire model (LIF) is used as the basic block in this paper, aiming to imitating the working process of spiking neural model. The post-neuron in LIF model receives the input signals from pre-neurons connected to it and integrates them through the function of synapses, then the output spike train is obtained according to the dynamic process of membrane voltage changing with time [42]. As shown in Fig. 1, the spike train $\delta_{i,t}$ in layer i is aggregated to layer j by weight w_{ij} and generates the current stimulation on it, which can be express as

$$I_{j,t} = \sum w_{ij}\delta_{i,t} \tag{7}$$

Stimulation of neurons will lead to the diffusion of ions on both sides of the cell membrane, forming the potential differences, which will accumulate gradually as time goes on. The dynamic function of membrane potential in LIF is

$$C_m\frac{dV_{j,t}}{dt} = -g_L(V_{j,t}-V_L) + I_{j,t} \tag{8}$$

where C_m is the membrane capacitance, g_L is the leaky conductance and the V_L is the leaky potential. Assume that the total excitatory conductance is g_E, which is dynamically changed according to the pre-synaptic neurons fire.

$$\tau_E\frac{dg_E}{dt} = -g_E + \sum w_{ij}\delta_{i,t} \tag{9}$$

where τ_E is the conductance decay of excitatory neurons. Thus, the Eq. (8) could be converted to

$$\tau\frac{dV_{j,t}}{dt} = -(V_{j,t}-V_L) + \frac{g_E}{g_L}(V_{j,t}-V_E) \tag{10}$$

in which V_E is the reversal potential and $\tau = \frac{C_m}{g_L}$.

When membrane potential exceeds a certain threshold V_{th}, the current neuron is activated and generates spike $\delta_{j,t}$, after which the potential will be rested to resting potential V_{rest}, waiting for the firing next time. It can be express as

$$if\ V_{j,t} > V_{th}, V_{j,t} = V_{rest}, \delta_{j,t} \tag{11}$$

Conversely, when the membrane voltage less than threshold, action potential will not be generated and the $V_{j,t}$ will declining to resting state according to the time constant τ if there is no spike stimulation for a long time.

The SNN's weight modification is related to the arrival time of presynaptic spikes and the firing time of postsynaptic neurons, which is the Spike Time Dependent Plasticity [43]. STDP was observed in natural circuits and considered as the basic rule for human brain works [43], [44], which consists of two opposite processes: long-term potentiation (LTP) and long-term depression (LTD). Within certain time window, when the presynaptic neuron spikes appear earlier than the postsynaptic neuron spikes, LTP will be generated. In this situation, the firing of adjacent neuron spikes is thought to have a causal relationship, and its connection weight is strengthened. Correspondingly, when the postsynaptic neuron spikes are earlier than the presynaptic spikes, or even the postsynaptic pulse is generated when there is no presynaptic stimulus, the LTD is triggered and its connection weight is weakened. The closer the adjacent spikes are, the greater the enhancing or weakening effect will be. The detailed weight adjustment process is demonstrated as the formulation.

$$\Delta W_{ij}(t_i,t_j) = \begin{cases}\Delta W_{ij}^{+} = A_+ e^{\frac{t_i-t_j}{\tau_+}}, & t_i < t_j\\ \Delta W_{ij}^{-} = -A_- e^{\frac{t_i-t_j}{\tau_-}}, & t_i > t_j\end{cases} \tag{12}$$

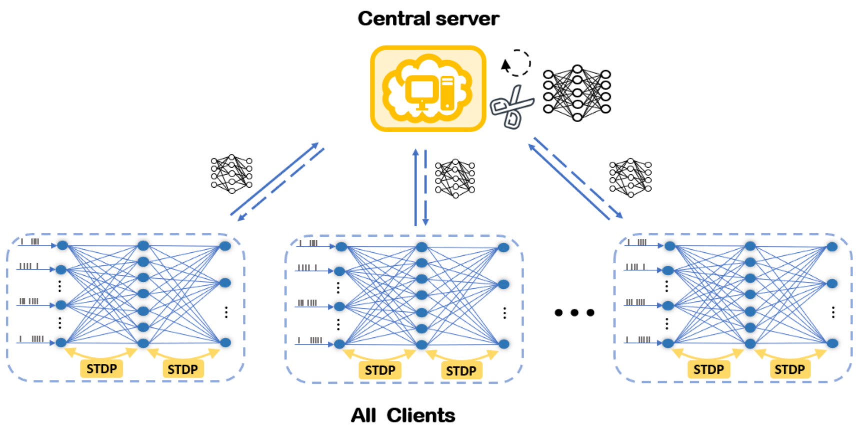

Fig. 2. The illustration of FL based on SNNs

where A_+ and A_- are the scaling factors, t_i and t_j are the firing time of neurons in layer i and layer j respectively. τ_+ and τ_- are the decay time of potentiation and depression.

D. Weight Pruning

The iterative pruning and LTH are used to compress the client model in FL to reach a predefined sparsity. It is worth noting that the iterative pruning directly iterates over the pruned weights, while the weight values retained after pruning in LTH have to be returned to the initial state. In addition, we also take random pruning into account, which removes a predetermined proportion of the model at random to achieve the desired sparsity [45].

III. Federated Radar Gesture Recognition Based on STDP

The detailed federated radar gesture recognition process is described in this section and its model framework is shown in Fig. 2. Firstly, the micro-Doppler feature set is obtained from the radar signal through STFT. Then, the preprocessed dataset is fed into the SNN-based FL framework, which updates the client-side model by the STDP algorithm. In order to further reduce the communication cost of the FL during parameter transmission, we propose the weight pruning technique based on STDP and apply it to the learning process of FL scenarios. To evaluate the energy consumption of the model, we use the BP-based Lenet and BPTT-based SNN as benchmarks to analyze the computational complexity of the proposed algorithm. In addition, we take the gradient-based data reconstruction attack as an example and validate the security of the proposed model.

A. Radar data preprocessing

The human posture IF signal $r(t)$ collected by radar is a three-dimensional data $L \times H \times x$, where L is the ADC number of spikes, H is the collection of chirps per frame and x is the total number of frames. The discrete signal $r[i]$ can be obtained by ADC sampling of IF signal as follow

$$r[i] = r(t = iT_s) \tag{13}$$

where T_s is the sampling period. For chirp $r_h[i]$, the range profile $R_h[k]$ for $h = 1, ..., H$ is acquired by DFT:

$$R_h[k] = \frac{1}{\sqrt{L}} \sum_{i=0}^{L} \mathcal{B}[i] r_h[i] e^{-j2\pi \frac{ki}{L}} \tag{14}$$

where $\mathcal{B}$ is the Blackman window [46]. Then the micro-Doppler can be obtained according to the STFT on the $R_h[k^*]$

$$\mathcal{D}[u, f] = \sum_{h=-\infty}^{\infty} R_h[k^*] \mathcal{H}_l[h - l_w R] e^{-j2\pi f h} \tag{15}$$

where $\mathcal{H}$ is the Hamming window and R is the hop size. k^* is the range bin which represents the micromotion of target. In STFT, the window length l_w is vital, which gives a trade-off between time bins N_T and frequency bins N_f. The time bins N_T can be calculated according to [47]

$$N_T = \lfloor \frac{nH - (l_w - R)}{R} \rfloor \tag{16}$$

where $l_w - R$ is the overlapping area of windows. To this extent, the micro-Doppler feature $\mathcal{D}$ of radar gesture dataset with size of $N_T \times N_f$ is obtained. In order to make the feature set better to input into the neural network for recognition and classification, we cut $\mathcal{D}$ along dimension u and divide it into several patches. In addition, the band-limiting method is adopted to reject the impact of high-frequency noise.

B. STDP-based Federated Learning with Weight Pruning

After preprocessing the radar human posture data, the micro-Doppler feature set $\mathcal{D}$ is obtained, which is then fed into the FL framework for precise recognition. We assume that a total of N clients are deployed in the FL for radar posture analysis. In each communication round r, partial clients are selected to participate in the

Algorithm 1 Federated Training Method based on STDP

```
Input: Set of $N$ clients; micro-Doppler dataset $\mathcal{D}$; number of rounds ($R$); number of local epochs ($E$); time step ($T$); client participation ratio ($p$); pruning ratio ($s$); pruning interval ($I_p$)
Output: The model parameter
 1: Server execute:
 2: Model initialization: Allocate the dataset to client $\mathcal{N}_n$; initialize the model weight as $M^0$; set the connectivity matrix to $m$ (elements are all 1).
 3: for round $r \leftarrow 0$ to $R$ do
 4:     broadcast the model from central server to all clients $M_n^r$
 5:     client selection $\mathcal{N}_{n_r} \leftarrow random\_choice(\mathcal{N}_n, p)$
 6:     for each participating client $\mathcal{N}_c (c \in n_r)$ do
 7:         calculate parameter updates $\Delta M_c^r$
 8:         update local model based on the connectivity matrix $M_c^r \leftarrow M_c^{r-1} + \mu * \Delta M_c^r \odot m$
 9:     end for
10:     upload client parameter and update global model
11:     if $r \mid I_p$ then
12:         update connectivity matrix $m$ by ratio $s$ according to the weight magnitude
13:     end if
14: end for
15: return
16: Client update:
17: encode input data into spike sequences in time period T $\delta^0 \leftarrow Poisson\_Generator$
18: for Layer $l \leftarrow 1$ to $L$ do
19:     calculate $\delta^l$ by forward propagation based on the dynamic process of neurons
20: end for
21: compare the spike release times of $\delta^l$ and $\delta^{l+1}$ and calculate the $\Delta W$ according to STDP
22: return
```

training, which can reduce the energy consumption of the whole network. Then the trained model is uploaded to the central server for parameter integration. To further reduce the computational cost of the network, we take the weight pruning to compress the whole model to a certain sparsity by iterating multiple times within a given interval. Specifically, when the model reaches stability, we judge the network connectivity based on the magnitude of the weight at the central server and obtain a mask to prune the model, which will be assigned to the client for the next round of training. The specific process is shown in Algorithm1.

Initialize the local model: Before training starts, we initialize the model parameter and connectivity matrix as M^0 and m, respectively. m indicates the connectivity importance of the network and $m \in \{0, 1\}$. Then the initialized parameter and micro-Doppler feature set $\mathcal{D}$ are assigned to the client, so the $n-th$ client has an initial model parameter $M_n^0 = M^0$ and a local private dataset $\mathcal{D}_n$

$$\mathcal{D}_n = \{(d_1, y_1), (d_2, y_2), ..., (d_i, y_i)\} \tag{17}$$

where y_i is the label.

Client Model Update: The client $\mathcal{N}_c (c = 0, 1, .., n_r)$ is randomly selected from all clients to participate the model training according to its local dataset, where $n_r (n_r \in N)$ is the number of participating clients. In each client, the input data is converted to a spike sequence under the certain encode rule and the information is transmitted forward through synaptic connections. Specifically, the post-neuron receives current stimulus from the pre-neuron, integrates it, and discharges when the membrane voltage meets the firing conditions, generating corresponding trigger spikes

$$\delta = \{\delta^1, ..., \delta^l, ..., \delta^L\} \tag{18}$$

where $l = 1, ..., l, ..., L$ indicates the number of layers in SNN. It is noteworthy that the update of the neurons in the output layer is a bit more complicated, which can be performed in two steps [48], [49]. In the inference step, the neurons act same with input layer according to the Eq. (9) and Eq. (10). And in the learning phase, the effect of target signal is added into the neuron firing as

$$\tau_E \frac{dg_E}{dt} = -g_E + \sum w_{ij} \delta_{i,t} + \beta (y_j - g_E) \tag{19}$$

where y is the jth target signal and β is the effect factor of the target signals. By comparing the spikes' firing order and time difference between the $i-th$ neuron in layer l and the $j-th$ neuron in layer $l+1$, the update of parameter M_{ij} can be obtained, which is described in Eq. (12). When $|t(\delta_i^l) - t(\delta_j^{l+1})| < t_w$, it indicates that the spike firing time of adjacent neurons is closer and can affect the weight adjustment; on the contrary, the firing time is far away and considered that there is no causal relationship between them. Specifically, when $(t(\delta_i^l) - t(\delta_j^{l+1})) < 0$

$$\Delta M_{ij} = A_+ e^{\frac{t(\delta_i^l) - t(\delta_j^{l+1})}{\tau_+}} \tag{20}$$

Conversely

$$\Delta M_{ij} = -A_- e^{\frac{t(\delta_i^l) - t(\delta_j^{l+1})}{\tau_-}} \tag{21}$$

At round r, the client $\mathcal{N}_c$ will obtain the model parameter

$$M_c^r = M_c^{r-1} + \mu * \Delta M_c^r \odot m \tag{22}$$

where μ is the learning rate of SNN, $\odot$ denotes the element-wise product and M_c^r is the accumulated update over the course of local training.

Central server parameters integration: After that, the participating clients will upload their parameters to the central server for integration and the parameter M^r of the global model is calculated according to $FedAvg$ algorithm.

$$M^r = \frac{1}{|n_r|} \sum_{c \in n_r} M_c^r \tag{23}$$

Update model pruning mask: When r is an integer multiple of I_p, we prune the current model with ratio of s and update m. Specifically, when magnitude-based

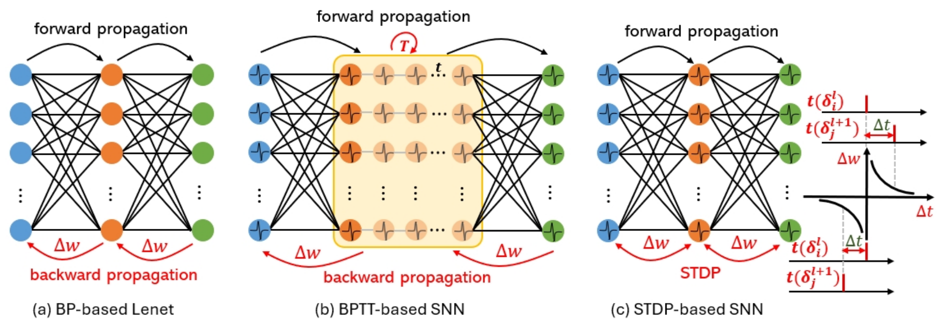

Fig. 3. The illustration of the BP-based Lenet, BPTT-based SNN and STDP-based SNN.

iterative pruning is adopted, the value in M^r is compared and the smallest $(1-s)\%$ is pruned to obtain mask m. After that, $m \odot M^r$ is assigned as a new model parameter to the client training. It is worth noting that m is an all-1 matrix in the initial stage, and the part to be pruned is set to 0 during the pruning iteration. The LTH expects to find a winner ticket in the initial model, which can achieve comparable accuracy with the original network in a similar number of iterations. Therefore, it takes $m \odot M^0$ as the sub-network parameter. Random pruning without any delicate pruning criteria, which removes $(1-s)\%$ of parameters randomly.

Finally, the central server sends the new model to clients to start the next round of training. This process repeats R rounds until the learning of the model is finished.

C. Computational Complexity Analysis

It is worth noting that backpropagation is still the most mainstream neural network optimization algorithm, which updates model parameter by propagating the error gradient in reverse. In order to verify the proposed FL framework more comprehensively, we take the BPTT-based SNN and BP-based Lenet as benchmarks, and observe their recognition performance on radar gesture data. Fig. 3 shows the difference between BP-based Lenet, BPTT-based SNN, and STDP-based SNN. Real values are used in Lenet to transmit information, and the error of the output layers is back-propagated to update the parameters. The SNN uses the spike trains to transmit the data, and BPTT-based approach gathers sequence information by unfolding Lenet in the time dimension and backpropagating layer by layer to obtain weights. The STDP-based technique computes the spike firing sequence of neurons within a time window and modifies synaptic size according to the sequence's temporal correlation. The update process of the BPTT and BP algorithms is similar, both of which obtain the signal's actual output $f(d_i)^L$ through forward feedback and then compare it with the expected output to obtain the loss function:

$$\mathcal{L}_i = \frac{1}{|\mathcal{D}_n|} \sum_{(d_i, y_i) \in \mathcal{D}_n} loss(y_i, f(d_i)^L) \tag{24}$$

then the weight updating can be shown as

$$\Delta M_c^r = \sum_{t=1}^{t} \frac{\partial \mathcal{L}_i}{\partial f(d_i)^l} \frac{\partial f(d_i)^l}{\partial f(d_i)^{l-1}} \cdots \frac{\partial f(d_i)^1}{\partial M_c} \tag{25}$$

Although the BP and BPTT methods have good performance, they are artificially defined and do not conform to brain's actual working process. The STDP-based SNN training algorithm can considerably reduce computational energy consumption while being biologically interpretability, making it suited for resource-constrained devices deployed at the edge in FL scenarios. The main manifestations are as follows:

1) The information in the Lenet is transferred through real values, resulting in a large number of matrix multiplication operations in the forward calculation process, whereas the operations in the SNN are primarily superposition due to its discrete pulses. Furthermore, the spike sparsity in SNN is strong, and the node is set to one when triggered and remains at zero otherwise, reducing energy consumption significantly;
2) The BP algorithm needs to broadcast the output layer's error back when updating the parameters, which surely doubles the computational complexity of the network. In contrast, the STDP rule only needs to compare the spike moments across adjacent nodes to optimize the model;

We use a three-layer fully connected network to compare the computational complexity of three methods, each of which has four components: feed-forward propagation from input layer to hidden layer (FF_1), feed-forward propagation from hidden layer to output layer(FF_2) and their corresponding feedback propagation FB_1 and FB_2. Assume that the number of neurons in the network's input, hidden, and output layers is m, n, and k, respectively, then the computational complexity of STDP-based SNN, BPTT-based SNN and BP-based Lenet can be demonstrated as follows:

$$\mathcal{O}_{STDP} = \mathcal{O}(FF_1) + \mathcal{O}(FF_2) = \mathcal{O}(mn) + \mathcal{O}(nk) \tag{26}$$

$$\begin{aligned} \mathcal{O}_{BPTT} &= \mathcal{O}(FF_1) + \mathcal{O}(FF_2) + \mathcal{O}(FB_1) + \mathcal{O}(FB_2) \\ &= \mathcal{O}(mn) + \mathcal{O}(nn+nk) + \mathcal{O}(nm) + \mathcal{O}(kn+nn) \end{aligned} \tag{27}$$

$$\begin{aligned}\mathcal{O}_{BP} &= \mathcal{O}(FF_1)+\mathcal{O}(FF_2)+\mathcal{O}(FB_1)+\mathcal{O}(FB_2)\\ &= \mathcal{O}(mn)+\mathcal{O}(nk)+\mathcal{O}(nm)+\mathcal{O}(kn)\end{aligned} \tag{28}$$

D. Communication Cost Analysis

In the following paragraph, we analyze the effect of weight pruning on the FL's communication cost. FL needs to transmit the model parameter between the central server and the client at each communication round. The communication cost of the original model is $\mathcal{O}_c = \mathcal{O}(mn+nk)$, and when the model is pruned with a rate of $s\%$, the ratio of the parameters' numbers after pruning to the original is $s\%$. However, in addition to network's weights, pruned FL also requires the transmission of parameter index information to the clients during model optimization. In our implementation, we use 32bit floating point numbers to store the values of the weights and 16bit integers to store the row and column indices of the non-zero elements [50]. Therefore, the model's communication cost after pruning is $\mathcal{O}_p = 2s * \mathcal{O}_c$.

E. Security Analysis

FL updates the model by transferring gradients rather than sharing data across clients, which protects data privacy. However, deep leakage from gradient (DLG) can still reconstruct data by stealing uploaded gradients, posing a security risk [51]. DLG is a classical data reconstruction attack that approximates the real gradient with a dummy gradient to bring the dummy data infinitely near to the real data. In the past, DLG attack was used in DNN-based FL. Specifically, for the input data and the label (d, y), the model can obtain the gradient ΔW. In DLG, the adversary creates a set of dummy label and data (d_m, y_m), as well as a corresponding gradient ΔW_m based on (d_m, y_m).

$$\Delta W_m = \frac{\partial \mathcal{L}(f(d_m, W_m), y_m)}{\partial W_m} \tag{29}$$

Then, the adversary takes the difference between dummy gradient and real gradient as the loss function, and optimize the dummy data d_m and dummy label y_m by L-BFGS optimizer [52] to make them approximate the real data (d, y)

$$\begin{aligned} d_m, y_m &= \arg\min_{d_m, y_m} \left\|\Delta W_m - \Delta W\right\|^2 \\ &= \arg\min_{d_m, y_m} \left\| \frac{\partial \mathcal{L}(f(d_m, W_m), y_m)}{\partial W_m} - \Delta W \right\|^2 \end{aligned} \tag{30}$$

In DNN, the signal propagates forward layer by layer through real values, and the entire process is continuous. Therefore, the input data in DNN has a non-linear mapping relationship with the weights and (d_m, y_m) can be optimized according to the minimization direction of $\|\Delta W_m - \Delta W\|^2$. The signal in SNN propagates forward through discrete spike sequences, and the dynamic processes of neurons make the information transmission discrete. Therefore, SNN lacks gradients and can calculate their weights based on the temporal correlation of neurons' spikes, but they are unable to utilize weights to infer changes in spikes in reverse. As a result, the attacker cannot reconstruct the input information from the exchanged gradient in STDP-based FL. It can be seen that our proposed model is more secure than the FL based on Lenet.

IV. Experiment

In this paper, we observe the classification performance of STDP-based spiking neural network for radar gesture in a federated learning framework.

A. Gesture dataset

8GHz radar dataset: The 5-class 8-GHz radar dataset [53] is used in the paper, which collects the human postures at a distance of 2m from the antenna via ultra-low-power SISI radar [54]. The human gesture contains the following parts: swinging the arm in the vertical direction (vertical), swinging the arm in the horizontal direction (horizontal), swinging the arm with the hand facing outward (hello), palms facing outward away from or near the radar (toward) and no movement of the above postures (background). The radar transmitted 192 chirps in each signal frame, while the number of ADC samples per chirp is 512. The time interval of the chirp transmitted is 1.2ms, and the emitted duration time is 41us, so the sampling time per frame is 238ms. We use a hamming window with a length of 192 to acquire the micro-Doppler characteristics by multiplying IF signals with the window function and performing the short-time Fourier transform (STFT). In order to balance the constructed radar gesture dataset, we take the background acquisition with the minimum number of frames as the benchmark, on which the STFT is performed and then cut the micro-Doppler feature set into patches of 48 time samples along the first dimension. Furthermore, we eliminate the first and last six examples altered to remove startup and ending parts, yielding 339 samples. Thus, we construct a balanced dataset of 1965 samples with five categories. Finally, limit the bandwidth of the Doppler characteristic to obtain a feature of size (100,48), which can be input into the SNN for recognition and classification [29].

Google Soli dataset: The Google Soli dataset collects the hand gestures from a 60GHz FMCW millimeter-wave radar, which composed of 12 classes with a total of 5500 CFAR-processed range-Doppler magnitude measurements [55]. The dataset, in particular, has 11 movements and a background, and each gesture creates sequences of 2D range-Doppler with a size of 32×32 across time [56]. The execution time of gestures varies, so the length of the sequence in the dataset is different and we used a fixed chunk for the experiment. The Soli radar IC includes four receive antenna elements, thus the dataset contains four channel and we choose one of them to validate the algorithm proposed in the paper.

B. Experiment Setup

In FL framework, two data distribution strategies are considered: (i) independent identical distribution (IID): each client has the same data distribution. (ii) non-independent identical distribution (non-IID): the data distribution among each client is unbalanced, which uses a Dirichlet distribution with concentration parameter f to sample the proportion of each class[57], [58]. The smaller the parameter f, the more obvious the class imbalance of the local training set between different clients. Compared with IID, non-IID is more conform to reality. The experiment sets a total of 100 communication rounds and each round conduct one local epoch training. After each round, the updated model parameters are integrated in the central server and redistributed to the clients, until the model is trained. In addition, the local datasets do not change between rounds. We use the k-folder cross-validation in experiment to observe the recognition accuracy, which is set as $k = 5$ in the initial stage. Considering the limited number of samples, we set a total of 12 clients to equally split the training data.

SNN is selected as the basic framework of FL and STDP-based algorithm is used to optimize the model. We set $\tau = 20ms$, $V_L = -70mV$, $V_{rest} = -80mV$ and $V_{th} = -54mV$. The preprocessed 8GHz human posture radar data is a matrix with the dimension of 100*48. By normalizing the signal and comparing it with the random sample value in (0,1) at T times, we can generate 4800 spike sequences with the length of T, which will be used as the input trains of SNN. We adopt a three-layer fully connection structure 4800-2000-5 for gesture recognition and the STDP rule is used to training the model. Similarly, when validating the performance of the model algorithm using the Google Soli dataset, a three-layer fully connected network with size 1024-2000-12 was constructed. We also carry out the comparative analysis of radar gesture recognition, which replaces the client-side SNN in the FL framework with BP-based Lenet and BPTT-based SNN of the same size. In addition, we add weight pruning during the FL training process, which can reduce the communication cost by setting the unimportant weight connections in the network to zero. We focus on the magnitude-based iteration pruning and LTH, and compare them with the random pruning to observe the performance of the network.

C. Experiment Results

Model performance comparison. Firstly, we compare the performance of FL in IID and non-IID when the client-side model is BP-based Lenet, BPTT-based SNN and STDP-based SNN. We set half of the clients to participate in the training.

8GHz radar dataset: The detailed training process for 8GHz radar dataset is shown in Fig. 4, which demonstrates the trend of test accuracy with communication round. In Fig. 4, the accuracy of radar gesture recognition based on BP training is the highest with $95.39\% \pm 0.56\%$ when the client data in FL obey IID distribution, followed by that based on STDP training with $94.26\% \pm 0.53\%$, and the recognition result based on BPTT training is the worst, which is $91.48\% \pm 1.1\%$. In addition, the model converges fastest when the client model is Lenet, followed by STDP and worst by BPTT. When the client data follows the non-IID distribution, we define the unbalance factor as $f = 0.5$. According to the experimental results, the convergence accuracies based on STDP and BP methods are similar, which are significantly better than the result of BPTT. Comparing the convergence curves under different distributions, we can find that the Lenet and BPTT-based algorithms are less stable, while the STDP-based model has a smoother convergence process.

Google Soli dataset: The detailed learning process for Google Soli dataset is shown in Fig. 5. It is obvious that the radar gesture recognition accuracy based on Lenet

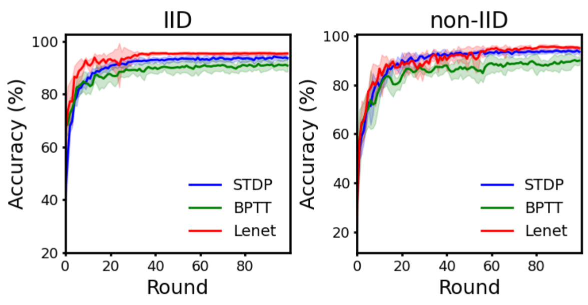

Fig. 4. 8GHz radar dataset: Comparison of the FL's training progress for 12 clents with half participating in each round.

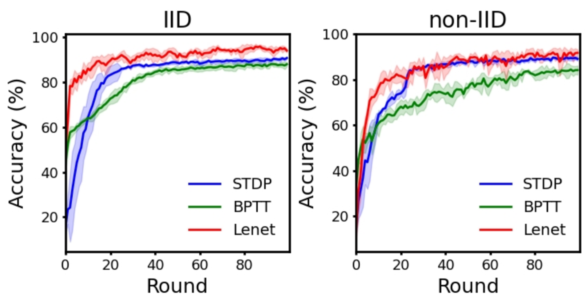

Fig. 5. Google Soli dataset: Comparison of the FL's training progress for 12 clents with half participating in each round.

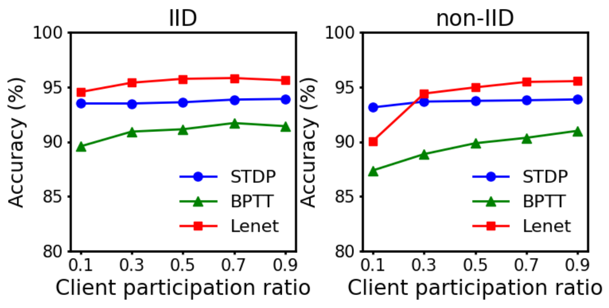

Fig. 6. 8GHz radar dataset: Comparison of radar gesture recognition accuracy under different client configurations with IID distribution and non-IID distribution

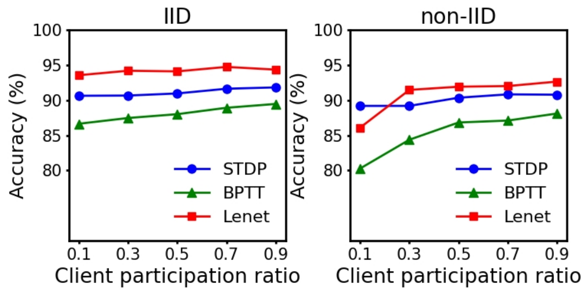

Fig. 7. Google Soli dataset: Comparison of radar gesture recognition accuracy under different client configurations with IID distribution and non-IID distribution

is highest when the client data obeys IID distribution, followed by that based on STDP and the recognition performance based on BPTT is worst. The convergence of the model is also optimal for Lenet and worst for BPTT. When the client data comes from a non-IID distribution with an unbalance factor of 0.5, the Lenet-based FL converges slightly faster than the STDP, and their recognition performance is comparable. Furthermore, the convergence accuracy under diverse data distributions demonstrates that the STDP-based FL is the most stable, but the BPTT-based models are highly affected by data distribution.

Impact of client configuration. Considering that the federation system in real life involves a large number of devices, the FL model in the experiment needs to take device scalability into account. To evaluate the detailed performance, we observe the recognition accuracy of the radar dataset under various participation ratios p.

8GHz radar dataset: As shown in Fig. 6, the accuracy drops gradually as the p decreases, and the performance of radar posture recognition in the three training modes keeps the same trend in the case of IID. When the client data obeys the non-IID distribution, the recognition accuracy of the model differs significantly from that of the IID case. Specifically, the performance of the FL with Lenet is better when the number of participating clients is higher, and it is significantly better than the other two. As the participation ratio p decreases, the performance of the Lenet-based FL drops dramatically and gradually becomes equal to or even lower than that in the STDP-based FL when p is less than 0.5, which both outperform the performance of BPTT. Comparing the results under different distributions, we can also find that the model with STDP has stronger robustness, which is less affected by the data distribution and the number of clients, while the Lenet- and BPTT-based networks have more variable results under different environmental configurations.

Google Soli dataset: The recognition accuracy of the Google Soli radar dataset under various participation ratios is shown in Fig. 7. When the data in the client obeys the IID distribution, the impact of client configuration on recognition performance is relatively mild. Overall, the FL model based on Lenet has the best performance, followed by STDP, and BPTT has slightly worse performance, and its classification accuracy gradually drops as the client participation ratio decreases. When the data is non-IID distribution, the Lenet-based FL performs better than the other two when a significant number of clients participate in the training, and classification accuracy declines rapidly when the p is less than 0.3. Furthermore, under non-IID distribution, the STDP-based FL performs steadily even when the client participation ratio fluctuates.

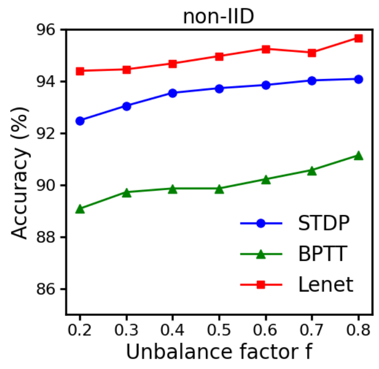

Fig. 8. 8GHz radar dataset: Effect of unbalance dataset among the clients

Impact of data distribution. We also observe how FL performs when the distribution of client data changes. As the non-IID data are generated through the Dirichlet distribution, we modify the unbalance of the data set distribution by adjusting f. We consider the participation rate p is 0.5 and perform the impact of different unbalance factors.

8GHz radar dataset: In Fig. 8, we gradually increase f from 0.2 to 0.8, it can be shown that as the data becomes more non-IID, the accuracy of radar gesture recognition based on FL gradually deteriorates under all three algorithms, but the range of variation is relatively slow without large fluctuations, indicating that the model is robust to unbalanced data.

Google Soli dataset: Fig. 9 depicts the gesture recognition performance of the Google Soli dataset on various data unbalance factors. It can be shown that the classification accuracy declines with increasing non-IID data, and Lenet- and STDP-FL have relatively strong robustness to data distribution, however the BPTT-based method is quite bad.

Energy consumption comparison. According to the energy calculations described in [59], we estimate the energy consumption on traditional hardware based on a 45nm CMOS for processing 32-bit integer arithmetic operations, in which the energy consumption of multiply and add operations are $E_{mul} = 3.1pJ$ and $E_{add} = 0.1pJ$ respectively. We compare the energy consumption of STDP, BPTT, and Lenet in a simple three-layer fully connected network. Note that, only Multiply and Accumulate (MAC) operations are taken into account, and the memory and peripheral circuit energy are neglected. The energy

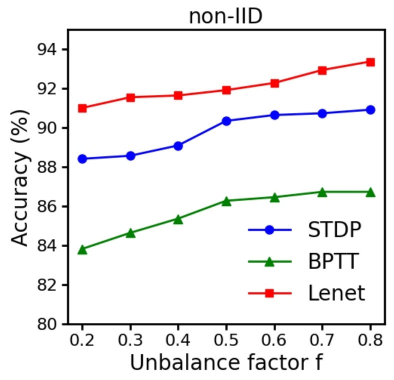

Fig. 9. Google Soli dataset: Effect of unbalance dataset among clients

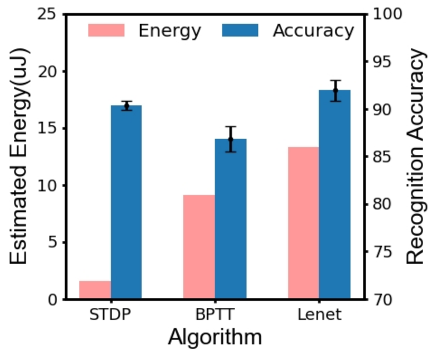

Fig. 11. Google Soli dataset: Comparison of energy consumption under different algorithms

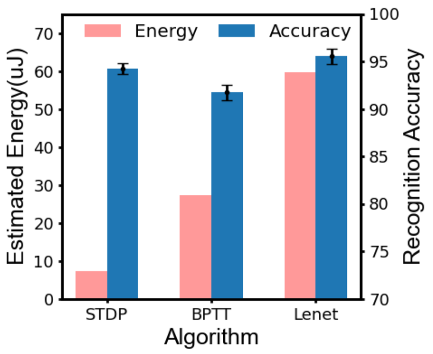

Fig. 10. 8GHz radar dataset: Comparison of energy consumption under different algorithms

consumption of Lenet can be calculated directly as

$$E_{Lenet}^{BP} = O_{BP} * (E_{mul} + E_{add}) \tag{31}$$

Different from Lenet, there is no multiplication operation in SNN, with only the weighted addition of spikes in the time dimension. Therefore, the energy consumption of SNN based on STDP is

$$E_{SNN}^{STDP} = O_{STDP} * E_{Add} * T * R \tag{32}$$

where T is total time steps, R is the spiking rate. Similarly, the energy consumption of BPTT is

$$E_{SNN}^{BPTT} = O_{BPTT} * E_{Add} * T * R \tag{33}$$

8GHz radar dataset: Fig. 10 shows the total estimated energy of three different algorithms at $p = 0.5$ with non-IID data distribution. With little difference in the radar gesture recognition accuracy, the energy consumption of the FL framework based on STDP is $7.208\mu J$, which is $3.78\times$ more efficient than the BPTT algorithm and $8.27\times$ more efficient than the Lenet.

Google Soli dataset: The comparison of FL's energy consumption based on STDP, BPTT and Lenet is shown in Fig. 11. The FL framework based on STDP is most energy-efficient with $1.554\mu J$ consumption, which is $5.86\times$ more than the BPTT and $8.53\times$ more than the Lenet.

Model pruning analysis. We study the weight pruning in FL based on STDP, which can reduce the computational cost in the model transmission process by setting the unimportant weights in the network to zero. The sparsity of the model is defined as s, which denotes the proportion of non-zero elements in the parameters. We analyze the pruning performance of the LTH in STDP-based FL and compare it with the magnitude-based iterative pruning and random pruning. We set the pruning rate as 50% and perform six rounds of iterative pruning, where the pruning rate of the output layer is halved because of its small number of nodes.

8GHz radar dataset: We observe the performance of model pruning when the data obeyed non-IID, which is shown in Fig. 12. The pruned model can match the original model's accuracy when the sparsity is 50.03% and 25.03%, and it converges even more quickly. After that, as the pruning portion grows, performance gradually declines. When the sparsity is 12.53% and 6.28%, the accuracy of the model after LTH pruning is nearly identical to the original model, and the performance of magnitude-based iterative pruning is slightly lower than the original model, while random pruning performs much worse. The performance of all three methods deteriorated significantly when pruning was continued. We compare the detailed pruning in Fig. 13 and find that the LTH has the best pruning result, with the accuracy loss of 1.12% and 1.90% when s is 12.53% and 6.28%, respectively. And when the sparsity s reaches 3.15%, the performance of the model decreases by 2.7%. The iterative pruning performance based on weight magnitude is slightly inferior to LTH, with accuracy loss is 1%$\sim$1.5% lower compared to the corresponding LTH when the sparsity is greater than 12.53%. In comparison, the random pruning method has the worst performance since its connections are randomly chosen without any good reasoning.

Google Soli dataset: We prune the model and observe the recognition performance of the Google Soli dataset, which is shown in Fig. 14. It is clear that LTH yields the best pruning outcomes, with classification performance remaining essentially stable when the fraction of non-zero elements is higher than 12.53% and progressively

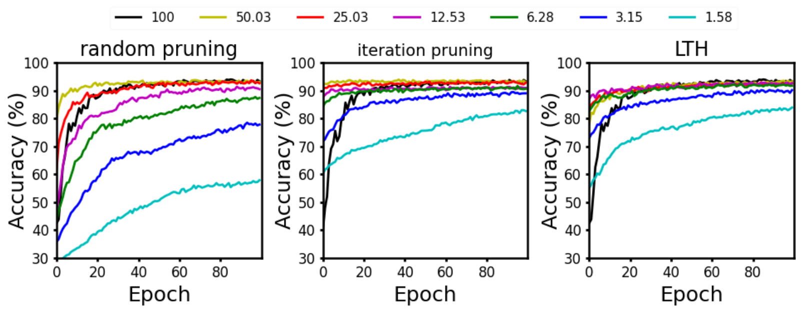

Fig. 12. 8GHz radar dataset: Test accuracy on STDP as training proceeds. The pruning results of the three approaches are illustrated in turn under various sparsity conditions. Labels are the sparsity(%) of the weights after pruning.

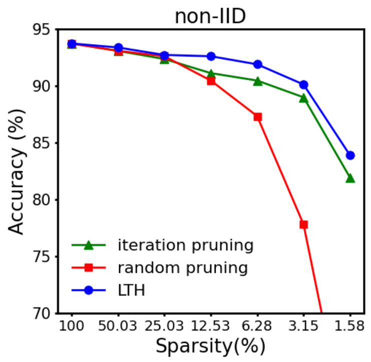

Fig. 13. 8GHz radar dataset: Pruning performance comparison of STDP-based FL for 12 clients with half participate in training

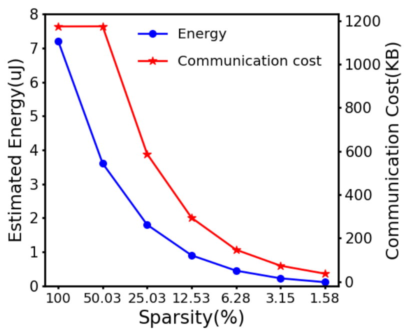

Fig. 15. 8GHz radar dataset: Variations of energy consumption and communication cost with sparsity

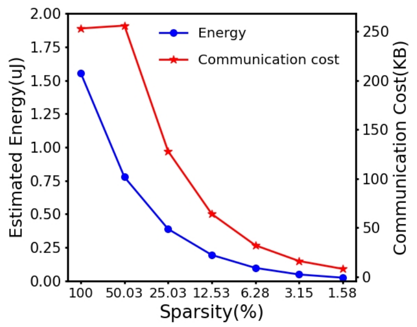

Fig. 16. Google Soli dataset: Variations of energy consumption and communication cost with sparsity

declining as s decreases. The performance of the iterative pruning is slightly inferior to LTH, and its performance progressively deteriorates as s decreases. The lowest performing model, on the other hand, is the random pruning model, whose performance significantly declines when its weights are compressed arbitrarily.

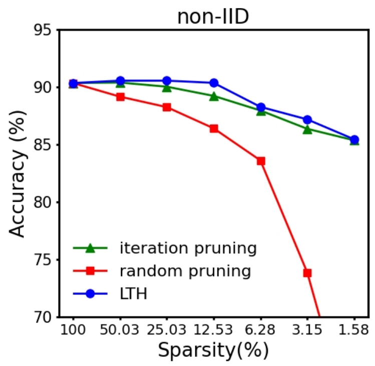

Fig. 14. Google Soli dataset: Pruning performance comparison of STDP-based FL for 12 clients with half participate in training

Communication cost comparison. We use a sparse matrix for model pruning, where a dense matrix is used for the full-sized model and a sparse matrix is used for the weights in the fully connected layer.

8GHz radar dataset: We plot the variation of energy consumption and communication cost with sparsity for each sample in Fig. 15 to further examine the performance of the pruned model. This figure demonstrates that as the model's non-zero element decreases, the energy consumption and communication cost of the model gradually drop. Combined with the results in Fig. 13, it can be seen that when LTH pruning method is used, the energy consumption of the whole model is only $0.45\mu J$, which is $15.8\times$ more efficient with little loss of accuracy compared to the original STDP-based FL. In this condition, the communication cost of the model is 146.79KB, which is reduced by $8\times$ compared to the original model. Combining with Fig. 10, it can be seen that the STDP-based FL is $132\times$ more efficient than the Lenet-based FL. To achieve the same energy consumption as the STDP-based FL, the Lenet-based FL must remove 99.24% of the parameters, which will result in a significant reduction in accuracy [38].

Google Soli dataset: Fig. 16 illustrates how the cost of communication and energy usage varied with changes in sparsity. Combined with the result in Fig. 14, it can be seen that the LTH pruning strategy performs well, losing 1.97% of the classification while keeping the model's sparsity at 6.28%. At this point, the model's energy consumption and communication cost are $0.098\mu J$ and 31.98KB, respectively, which is $15.86\times$ and $7.9\times$ more efficient than the STDP-based FL.

V. Conclusion

This paper investigates the SNN-based radar gesture recognition in the FL framework, which optimizes the model by the biologically inherent property and reduces the energy consumption of the entire network through weight pruning. Unlike traditional BP algorithms, our model optimizes the client network in FL through the STDP rule, which is cost-effective in the energy consumption of the whole network and desirable in solving the resource-limit problem at the edge nodes of FL. We also add model pruning to the process of network optimization. We creatively explore the weight pruning strategies, such as iterative pruning and LTH in STDP, which can greatly minimize the communication overhead during the model transmission in FL. We conduct comparison experiments with BP-based Lenet and BPTT-based SNN as benchmarks, which demonstrate that our STDP-based FL model can reduce energy consumption while maintaining similar performance. When weight pruning is considered, the energy consumption and communication cost of the model is further reduced with a negligible loss of accuracy.

Although this paper presents promising results with SNNs based on biological interpretability, there are still a lot of work to be explored in the future. Firstly, the method that integrating more biological characteristics needs to be explored, which can simulate the brain's operational process and be more efficient with improved learning performance. Secondly, the radar recognition circumstances in real life are more sophisticated, thus it is necessary to apply the federated learning framework to more intricate scenarios, such as sequence learning, few-shot learning, etc. Finally, this paper only considers the learning ability of the proposed model and ignores the evaluation of security and privacy protection, which is critical in a wide range of real-life scenarios.

REFERENCES

[1] V. C. Chen, F. Li, S.-S. Ho, and H. Wechsler, "Micro-doppler effect in radar: phenomenon, model, and simulation study," *IEEE Transactions on Aerospace and electronic systems*, vol. 42, no. 1, pp. 2–21, 2006.

[2] D. Tahmoush and J. Silvious, "Angle, elevation, prf, and illumination in radar microdoppler for security applications," in *2009 IEEE Antennas and Propagation Society International Symposium*, pp. 1–4, IEEE, 2009.

[3] V. M. Lubecke, O. Boric-Lubecke, A. Host-Madsen, and A. E. Fathy, "Through-the-wall radar life detection and monitoring," in *2007 IEEE/MTT-S International Microwave Symposium*, pp. 769–772, IEEE, 2007.

[4] Y. Li, Z. Peng, R. Pal, and C. Li, "Potential active shooter detection based on radar micro-doppler and range-doppler analysis using artificial neural network," *IEEE Sensors Journal*, vol. 19, no. 3, pp. 1052–1063, 2018.

[5] A.-K. Seifert, A. M. Zoubir, and M. G. Amin, "Radar classification of human gait abnormality based on sum-of-harmonics analysis," in *2018 IEEE Radar Conference (RadarConf18)*, pp. 0940–0945, IEEE, 2018.

[6] P. Pierleoni, A. Belli, L. Palma, M. Pellegrini, L. Pernini, and S. Valenti, "A high reliability wearable device for elderly fall detection," *IEEE Sensors Journal*, vol. 15, no. 8, pp. 4544–4553, 2015.

[7] P.-H. Chen, M. C. Shastry, C.-P. Lai, and R. M. Narayanan, "A portable real-time digital noise radar system for through-the-wall imaging," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 10, pp. 4123–4134, 2012.

[8] J. Li, Z. Zeng, J. Sun, and F. Liu, "Through-wall detection of human being's movement by uwb radar," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 6, pp. 1079–1083, 2012.

[9] Y. Kim and H. Ling, "Human activity classification based on micro-doppler signatures using a support vector machine," *IEEE transactions on geoscience and remote sensing*, vol. 47, no. 5, pp. 1328–1337, 2009.

[10] G. Li, R. Zhang, M. Ritchie, and H. Griffiths, "Sparsity-driven micro-doppler feature extraction for dynamic hand gesture recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 2, pp. 655–665, 2017.

[11] C. Ding, H. Hong, Y. Zou, H. Chu, X. Zhu, F. Fioranelli, J. Le Kernec, and C. Li, "Continuous human motion recognition with a dynamic range-doppler trajectory method based on fmcw radar," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 9, pp. 6821–6831, 2019.

[12] A. A. Pramudita *et al.*, "Time and frequency domain feature extraction method of doppler radar for hand gesture based human to machine interface," *Progress In Electromagnetics Research C*, vol. 98, pp. 83–96, 2020.

[13] M. S. Seyfioğlu, A. M. Özbayoğlu, and S. Z. Gürbüz, "Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1709–1723, 2018.

[14] Y. Kim and T. Moon, "Human detection and activity classification based on micro-doppler signatures using deep convolutional neural networks," *IEEE geoscience and remote sensing letters*, vol. 13, no. 1, pp. 8–12, 2015.

[15] S. Franceschini, M. Ambrosanio, S. Vitale, F. Baselice, A. Gifuni, G. Grassini, and V. Pascazio, "Hand gesture recognition via radar sensors and convolutional neural networks," in *2020 IEEE Radar Conference (RadarConf20)*, pp. 1–5, 2020.

[16] S. Skaria, A. Al-Hourani, M. Lech, and R. J. Evans, "Hand-gesture recognition using two-antenna doppler radar with deep convolutional neural networks," *IEEE Sensors Journal*, vol. 19, no. 8, pp. 3041–3048, 2019.

[17] S. Franceschini, M. Ambrosanio, V. Pascazio, and F. Baselice, "Hand gesture signatures acquisition and processing by means of a novel ultrasound system," *Bioengineering*, vol. 10, no. 1, p. 36, 2022.

[18] Y. Sang, L. Shi, and Y. Liu, "Micro hand gesture recognition system using ultrasonic active sensing," *IEEE Access*, vol. 6, pp. 49339–49347, 2018.

[19] Y. Kim and B. Toomajian, "Hand gesture recognition using micro-doppler signatures with convolutional neural network," *IEEE Access*, vol. 4, pp. 7125–7130, 2016.

[20] J. Bryan, J. Kwon, N. Lee, and Y. Kim, "Application of ultra-wide band radar for classification of human activities," *IET Radar, Sonar*

& Navigation, vol. 6, no. 3, pp. 172–179, 2012.

[21] N. Kern, T. Grebner, and C. Waldschmidt, "Pointnet+ lstm for target list-based gesture recognition with incoherent radar networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 6, pp. 5675–5686, 2022.

[22] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[23] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[24] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[25] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[26] S. Subbulakshmi Radhakrishnan, A. Sebastian, A. Oberoi, S. Das, and S. Das, "A biomimetic neural encoder for spiking neural network," *Nature communications*, vol. 12, no. 1, pp. 1–10, 2021.

[27] E. Hunsberger and C. Eliasmith, "Training spiking deep networks for neuromorphic hardware," *arXiv preprint arXiv:1611.05141*, 2016.

[28] Y. Zhang, P. Qu, Y. Ji, W. Zhang, G. Gao, G. Wang, S. Song, G. Li, W. Chen, W. Zheng, *et al.*, "A system hierarchy for brain-inspired computing," *Nature*, vol. 586, no. 7829, pp. 378–384, 2020.

[29] A. Safa, F. Corradi, L. Keuninckx, I. Ocket, A. Bourdoux, F. Catthoor, and G. G. Gielen, "Improving the accuracy of spiking neural networks for radar gesture recognition through preprocessing," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[30] P. Gerhards, F. Kreutz, K. Knobloch, and C. G. Mayr, "Radar-based gesture recognition with spiking neural networks," in *2022 7th International Conference on Frontiers of Signal Processing (ICFSP)*, pp. 40–44, IEEE, 2022.

[31] M. Arsalan, A. Santra, and V. Issakov, "Spiking neural network-based radar gesture recognition system using raw adc data," *IEEE Sensors Letters*, vol. 6, no. 6, pp. 1–4, 2022.

[32] I. J. Tsang, F. Corradi, M. Sifalakis, W. Van Leekwijck, and S. Latré, "Radar-based hand gesture recognition using spiking neural networks," *Electronics*, vol. 10, no. 12, p. 1405, 2021.

[33] A. Safa, J. Van Assche, M. D. Alea, F. Catthoor, and G. G. Gielen, "Neuromorphic near-sensor computing: from event-based sensing to edge learning," *Ieee Micro*, vol. 42, no. 6, pp. 88–95, 2022.

[34] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang, "Stronger generalization bounds for deep nets via a compression approach," in *International Conference on Machine Learning*, pp. 254–263, PMLR, 2018.

[35] D. C. Mocanu, E. Mocanu, P. Stone, P. H. Nguyen, M. Gibescu, and A. Liotta, "Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science," *Nature communications*, vol. 9, no. 1, p. 2383, 2018.

[36] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, 2015.

[37] Y. Guo, A. Yao, and Y. Chen, "Dynamic network surgery for efficient dnns," *Advances in neural information processing systems*, vol. 29, 2016.

[38] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.

[39] F. Fioranelli, H. Griffiths, M. Ritchie, and A. Balleri, *Micro-doppler radar and its applications*. The Institute of Engineering and Technology (IET), 2020.

[40] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[41] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[42] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in neuroscience*, vol. 12, p. 331, 2018.

[43] Y. Dan and M.-m. Poo, "Spike timing-dependent plasticity of neural circuits," *Neuron*, vol. 44, no. 1, pp. 23–30, 2004.

[44] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

[45] M. Zhu and S. Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," *arXiv preprint arXiv:1710.01878*, 2017.

[46] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

[47] D. Banerjee, S. Rani, A. M. George, A. Chowdhury, S. Dey, A. Mukherjee, T. Chakravarty, and A. Pal, "Application of spiking neural networks for action recognition from radar data," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10, IEEE, 2020.

[48] S. Benjamin and B. Yoshua, "Equilibrium propagation: Bridging the gap between energy-based models and backpropagation," *Frontiers in Computational Neuroence*, vol. 11, 2017.

[49] Z. Hu, T. Wang, and X. Hu, "An stdp-based supervised learning algorithm for spiking neural networks," in *International Conference on Neural Information Processing*, 2017.

[50] Y. Jiang, S. Wang, V. Valls, B. J. Ko, W.-H. Lee, K. K. Leung, and L. Tassiulas, "Model pruning enables efficient federated learning on edge devices," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2022.

[51] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[52] A. Wainakh, F. Ventola, T. Müßig, J. Keim, C. G. Cordero, E. Zimmer, T. Grube, K. Kersting, and M. Mühlhäuser, "User-level label leakage from gradients in federated learning," *Proceedings on Privacy Enhancing Technologies*, vol. 2022, no. 2, pp. 227–244, 2022.

[53] J. Stuijt, M. Sifalakis, A. Yousefzadeh, and F. Corradi, "μbrain: An event-driven and fully synthesizable architecture for spiking neural networks," *Frontiers in neuroscience*, vol. 15, p. 664208, 2021.

[54] Y.-H. Liu, S. Sheelavant, M. Mercuri, P. Mateman, J. Dijkhuis, W. Zomagboguelou, A. Breeschoten, S. Traferro, Y. Zhan, T. Torf, *et al.*, "9.3 a680 μw burst-chirp uwb radar transceiver for vital signs and occupancy sensing up to 15m distance," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, pp. 166–168, IEEE, 2019.

[55] J. Lien, N. Gillian, M. E. Karagozler, P. Amihood, C. Schwesig, E. Olson, H. Raja, and I. Poupyrev, "Soli: ubiquitous gesture sensing with millimeter wave radar," in *International Conference on Computer Graphics and Interactive Techniques*, 2016.

[56] S. Wang, J. Song, J. Lien, I. Poupyrev, and O. Hilliges, "Interacting with soli: Exploring fine-grained dynamic gesture recognition in the radio-frequency spectrum," in *Symposium on User Interface Software & Technology*, 2016.

[57] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," *arXiv preprint arXiv:2002.06440*, 2020.

[58] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *International conference on machine learning*, pp. 7252–7261, PMLR, 2019.

[59] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14, IEEE, 2014.