IEEE INFOCOM 2022 - IEEE Conference on Computer Communications | 978-1-6654-5822-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/INFOCOM48880.2022.9796733

Joint Superposition Coding and Training for Federated Learning over Multi-Width Neural Networks

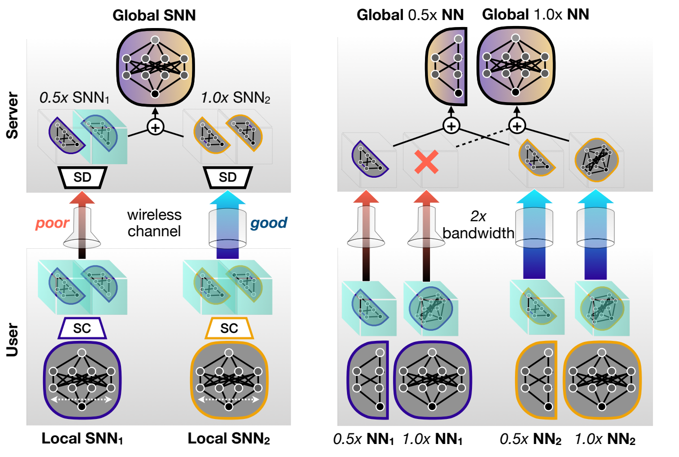
[†]Hankyul Baek, [†]Won Joon Yun, [†]Yunseok Kwak, [‡]Soyi Jung, [°]Mingyue Ji, [*]Mehdi Bennis,
[◇]Jihong Park, and [†]Joongheon Kim
[†]Department of Electrical and Computer Engineering, Korea University, Seoul, Republic of Korea
[‡]School of Software, Hallym University, Chuncheon, Republic of Korea
[°]Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, UT, USA
[*]Centre for Wireless Communications, University of Oulu, Oulu, Finland
[◇]School of Information Technology, Deakin University, Geelong, Australia
E-mails: {67back,ywjoon95,rhkrdbstjr0}@korea.ac.kr, sjung@hallym.ac.kr, mingyue.ji@utah.edu,
mehdi.bennis@oulu.fi, jihong.park@deakin.edu.au, joongheon@korea.ac.kr

***Abstract*—This paper aims to integrate two synergetic technologies, federated learning (FL) and width-adjustable slimmable neural network (SNN) architectures. FL preserves data privacy by exchanging the locally trained models of mobile devices. By adopting SNNs as local models, FL can flexibly cope with the time-varying energy capacities of mobile devices. Combining FL and SNNs is however non-trivial, particularly under wireless connections with time-varying channel conditions. Furthermore, existing multi-width SNN training algorithms are sensitive to the data distributions across devices, so are ill-suited to FL. Motivated by this, we propose a communication and energy efficient SNN-based FL (named *SlimFL*) that jointly utilizes *superposition coding (SC)* for global model aggregation and *superposition training (ST)* for updating local models. By applying SC, SlimFL exchanges the superposition of multiple width configurations that are decoded as many as possible for a given communication throughput. Leveraging ST, SlimFL aligns the forward propagation of different width configurations, while avoiding the inter-width interference during back propagation. We formally prove the convergence of SlimFL. The result reveals that SlimFL is not only communication-efficient but also can counteract non-IID data distributions and poor channel conditions, which is also corroborated by simulations.**

I. Introduction

Federated learning (FL) is a promising solution to enable high-quality on-device learning at mobile devices such as phones, cars, and drones [1], [2]. Each of these devices has only a limited amount of local data, and FL can overcome the lack of local model training samples by exchanging and aggregating the local models of different devices. To reach its full potential, it is essential to scale up the range of federating devices that are often wirelessly connected while having heterogeneous levels of available energy [3], [4]. This mandates addressing the following interrelated energy and wireless communication problems.

On the one hand, different devices have heterogeneous levels of available energy. Low-energy devices are likely to run small models, whereas high-energy devices prefer to operate large models. Unfortunately, FL can only aggregate the local models under the same architecture [1], so is able to train either small or large models at a time. To cope with



(a) SlimFL. (b) Vanilla FL-1.5x.

Fig. 1. A schematic illustration of (a) *slimmable federated learning (SlimFL)* using the slimmable neural networks (SNNs) with the superposition coding (SC) and successive decoding (SD), compared to (b) vanilla federated learning (Vanilla FL-1.5x) consuming 2x bandwidth.

heterogeneous energy capacity, one should therefore perform FL two times with the increased overall training time, or run FL simultaneously for two separate groups of devices with reduced training samples while compromising accuracy. Performing FL using a width-controllable *slimmable neural network (SNN)* architecture enables to train these two-level models at once while federating across all devices, after which each trained local SNN model can adjust its width [5].

On the other hand, wireless communication channel conditions vary over time and across different devices. When the channel information is known before transmission, poor-channel devices can only exchange small models, while good-channel devices can participate in FL using large models. SNN allows them to collaborate together, in a way that poor-channel devices send their local SNNs after reducing the widths, and contribute only to a fraction of the entire global model construction. This however entails extra communication and energy costs for probing channel conditions [6] that change over time and locations due to random fading and mobility.

Spurred by the aforementioned problems, we propose the first *SNN-based FL algorithm* that leverages *superposition coding (SC)* and *successive decoding (SD)*, coined *slimmable FL (SlimFL)*. By applying SNNs to FL, SlimFL can address the heterogeneous energy capacities. Besides, by exploiting SC and SD, SlimFL can proactively cope with the heterogeneous channel conditions for unknown channel state information.

To illustrate, consider an SNN with two width levels as shown in Fig. 1. In the uplink from each device to the server, the device uploads its local updates after jointly encoding the left-half (LH) and the right-half (RH) of its local SNN model while allocating different transmission power levels to them, i.e., SC [7]. Then, the server first attempts to decode the LH. If decoding the LH is successful, the server successively tries to decode the RH, i.e., SD or also known as successive interference cancellation (SIC). Accordingly, when the device-server channel throughput is low, the server can decode only the LH of the uploaded model, obtaining the *half-width (0.5x)* model. When the channel throughput is high, the server can decode both LH and RH, and combine them to yield the *full-width model (1.0x)*. Consequently, the sever constructs a global model superpositioning the decoded 0.5x and 1.0x local models, which is downloaded by each device. The device replaces its local model with the downloaded global model, and iterates the aforementioned operation until convergence.

In essence, the effectiveness of SlimFL hinges on creating a synergy between multiple width configurations, i.e., 0.5x and 1.0x models, which is however non-trivial. The global model is a mixture of different width configurations, so the standard FL convergence becomes questionable. Furthermore, the local model consists of multiple width configurations, so training them may interfere with one another. Existing SNN architectures and training algorithms are intended for standalone learning, so are ill-suited for SlimFL particularly under non-independent and identically distributed (non-IID) data distributions. To address these challenges in SlimFL, in this paper we develop novel SNN architecture and and training algorithm, named *superposition training (ST)*, and study the convergence and effectiveness of SlimFL. The major contributions of this paper are summarized as below.

1) We first propose an FL framework for SNNs, SlimFL (see Fig. 1(a) and **Algorithm 2**), which exploits SC for improving communication efficiency under time-varying wireless channels with limited bandwidth.
2) We develop a local SNN training method for SlimFL, ST (see **Algorithm 1**), which avoids unnecessary inter-width interference, and thus achieves fast convergence with high accuracy regardless of data distributions.
3) We propose an energy-efficient SNN architecture, Ultra Light MobileNet (see **Table I**), achieving $>$ 36x less FLOPS than the state-of-the-art SNN architecture [5].
4) We prove the convergence of SlimFL (see **Theorem 1**). The result shows the favorable conditions of SlimFL in terms of the channel quality and data distributions, and provides the optimal transmit power allocation guideline on SC (see **Proposition 1**) as well as the optimal weight guideline on ST (see **Proposition 2**).
5) We corroborate our analysis by simulation, showing that compared to vanilla FL (see Fig. 1(b)), SlimFL achieves higher accuracy and lower communication costs under poor channel conditions and non-IID data distributions.

The notations in this paper are listed in Tab. VIII.

II. Related Work

A. Multi-Width/Depth Neural Networks

To meet different on-device energy and memory requirements, it is common to prune model weights [8] or transfer a large trained model's knowledge into a small empty model via knowledge distillation (KD) [9], which however incurs additional training operations. Alternatively, one can adjust a trained model's width and/or depth in accordance with the resource requirements. Following this principle, depth-controlled neural networks [10] and adaptive neural networks [11] can adjust their depths after training, whereas SNNs tune their widths [5]. In this paper, we leverage width-controllable SNNs, and develop its FL version, SlimFL. Such an extension is non-trivial, and entails several design issues, such as local SNN training algorithms, aggregating segment prioritization.

B. Superposition Coding & Successive Decoding

In a nutshell, SC encodes two different data signals into one while allocating two different power levels before transmissions [7]. After receptions, SD decodes the SC-encoded signal by first decoding the stronger signal, followed by subtracting it and decoding the remainder as the weaker signal [12]. SC has been widely utilized in communication systems, particularly for simultaneously supporting different devices in the context of non-orthogonal multiple access (NOMA) [13]. We apply the same principle for supporting a single device simultaneously requesting two types of data with different priorities, such that the higher priority signal should almost surely be decoded while the lower priority signal can be successively decoded only under good channel conditions. Precisely, SlimFL makes an SNN's LH a higher priority so as to receive the 0.5x model even under poor channels. It can decode the SNN's RH only when the channel conditions are good, obtaining the 1.0x model by combining both LH and RH. Consequently, SlimFL ensures stable convergence under poor channels.

C. FL Convergence Analysis

FL convergence has recently been studied extensively [14], [15], among which we fundamentally rely on the following convergence results. Under IID data distributions, vanilla FL, also known as FedAvg, is equivalent to the local SGD algorithm whose convergence is known [16]. Under non-IID data distributions, the convergence of FedAvg is provided in [17] where the non-IIDness is determined by the bound of a dissimilarity between global and local average risks. Alternatively, the convergence under non-IID data distributions is proved by [15] where the non-IIDness is measured by the average of the local stochastic gradient variance, where the

TABLE I
MODEL ARCHITECTURE OF UL–MOBILENET.

UL–MobileNet Layers	Weight connection of layers 1.0x (0.5x)
Convolution layer	$C_{in} \times C_{out} \times K \times K$
Conv2D + ReLU6	$1 \times 32 \times 3 \times 3$ $(1 \times 16 \times 3 \times 3)$
Conv2D + ReLU6	$1 \times 32 \times 3 \times 3$ $(1 \times 16 \times 3 \times 3)$
Conv2D + ReLU6	$32 \times 32 \times 1 \times 1$ $(16 \times 16 \times 1 \times 1)$
Conv2D + ReLU6	$1 \times 32 \times 3 \times 3$ $(1 \times 16 \times 3 \times 3)$
Conv2D + ReLU6	$32 \times 64 \times 1 \times 1$ $(16 \times 32 \times 1 \times 1)$
Fully connected layer	$C_{in} \times C_{out}$
Linear	64×10 (32×10)

C_{in}, C_{out}, and K stand for dimension of input channel, dimension of output channel, kernel size, respectively, and $\text{ReLU6}(x) = \min(\max(0, x), 6)$.

average is taken across the devices. Taking into account non-IID data distributions, our SlimFL convergence analysis relies primarily on the method in [17], but our non-IIDness definition is similar to [15]. Note that the convergence of standalone SNN training has recently been studied in [18], yet without FL.

III. LOCAL MODEL ARCHITECTURE AND TRAINING

Existing SNN architectures and training algorithms are intended for standalone learning [5]. This section proposes a novel SNN architecture and its local training for SlimFL.

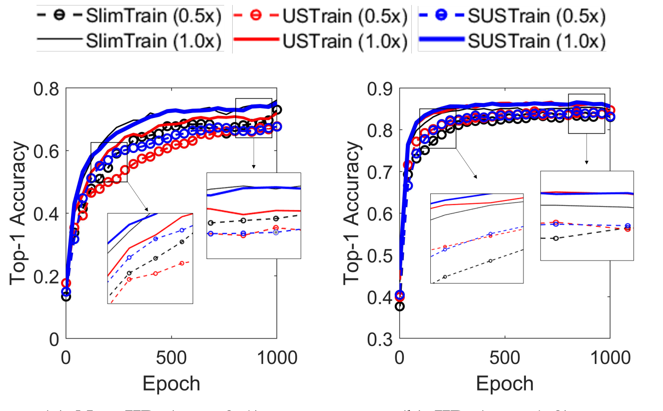
A. *Ultra Light SNN Architecture*

The state-of-the-art SNN architecture is the US-MobileNet proposed in [5]. As opposed to a *de facto* standard neural network architecture with a universal batch normalization (BN) layer, US-MobileNet is equipped with multiple separate BN layers to cope with all slimmable model configurations. While effective in standalone learning, in SlimFL with wireless connectivity, not all multi-width configurations are exchanged due to insufficient communication throughput, while the exchanged width configurations are aggregated across devices, diluting the effectiveness of BN. In our experiments we even observed training convergence failures due to BN. Furthermore, managing multiple BN layers not only consumes additional memory costs, but also entails high computing computing overhead. For these reasons, we remove BN layers, and consider a lighter version of US-MobileNet, named *Ultra Light MobileNet (UL-MobileNet)*, with the specifics provided in Tab. II. Compared to US-MobileNet with more than 100M FLOPS, UL-MobileNet costs only 2.76M FLOPS.

Hereafter, we consider that each device k has an SNN following the UL-MobileNet architecture. At the t-th iteration, the SNN model has the weight vector θ_t^k with two width configurations: 0.5x width configuration $\theta_t^k \odot \Xi_1$ and 1.0x width configuration $\theta_t^k \odot \Xi_2$ $(= \theta_t^k)$, where $\odot$ is the element-wise product and Ξ_i represents a binary mask for extracting the parameters of i-th width configuration.

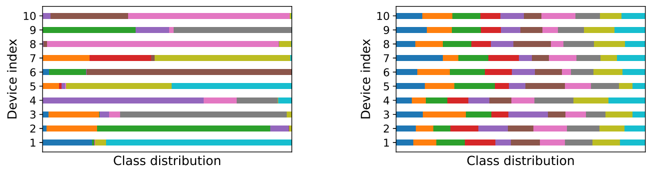
B. *Superposition SNN Training*

Training a multi-width SNN is challenging, in that the weights of earlier trained width configurations can be distorted by the latter backpropagation (BP) for other overlapping widths. This inter-width interference not only deteriorates the inference accuracy, while hindering the training convergence.



(a) Non-IID ($\alpha = 0.1$). (b) IID ($\alpha = 1.0$).

Fig. 2. Comparison of SNN training algorithms: SlimTrain [19], USTrain [5], and our proposed SUSTrain ($K = 10$, $\sigma^2 = -30\text{dB}$).



(a) Non-IID ($\alpha = 0.1$). (b) IID ($\alpha = 10$).

Fig. 3. An illustration of the data distributions across 10 devices for the different values of the Dirichlet concentration ratio α.

The first SNN training algorithm proposed in [19], referred to as SlimTrain (see Appendix 3), partly mitigates such inter-width interference by training different width configurations in descending order of size. While following the sample principle, the state-of-the-art SNN training algorithm proposed in [5], referred to as universal SNN (USTrain), additionally applies the inplace knowledge distillation (IPKD) from the full-width to all the sub-widths. The IPKD encourages each sub-width (i.e., student) to yield a similar softmax output (i.e., logit) to that of the full-width (i.e., teacher) such that their overlapping BP gradients become less different from each other, thereby reducing the inter-width interference.

However, as shown by experiments in Fig. 2, USTrain is unfit for SlimFL particularly under non-IID data distributions (i.e., $\alpha = 0.1$, see Fig. 3), where SlimTrain even outperforms USTrain. We conjecture that the problem comes from the use of outdated teacher's knowledge in USTrain. In USTrain, the teacher's logit is set as the value before updating the teacher's model, and is compared with a student after updating the teacher's model. Non-IID data distributions exacerbate this mismatch, where the full-width teacher model is significantly updated in the first epoch after downloading the global model due to the huge gap between local and global models.

To resolve this problem, we propose an ST algorithm, coined *superpositioned USTrain (SUSTrain)*, which first holds all the forward propagation (FP) losses, and then concurrently updates all the width configurations with the superpositioned gradients. In doing so, a student is trained using IPKD without the logit mismatch with its full-width teacher's logit, while the teacher is simultaneously trained using the ground truth. For the device k, the aforementioned local SNN update rule is,

Algorithm 1: Superposition Training (SUSTrain)

```
1  Initialize train parameter $\Theta = \{\theta^1, \cdots, \theta^k, \cdots, \theta^K, \theta^G\}$,
2  Initialize local dataset $\boldsymbol{Z} = \{Z_1, \cdots, Z_k, \cdots, Z_K\}$ with Dirichlet
     distribution
3  Initialize learning rate $\eta_t \leftarrow \eta_0$
4  Further Constraints: $\hat{F}^1(\cdot) = F^1(\cdot)$                ▷ Discuss in Sec. V
5  for $t = 1, \cdots, T$ do
6      for $k = 1, \cdots, K$ do
7          Initialize gradients of model optimizer as 0.
8          Sample batch $\zeta_t^k$ from $Z_k$.
9          Compute loss, $loss \leftarrow F^k(\theta_t^k \odot \Xi_i, \zeta_t^k)$.
10         Execute full-network $M(\theta_t^k, \zeta_t^k)$.
11         Accumulate gradients, $loss.backward()$.
12         Execute full-network $M(\theta_t^k, \zeta_t^k)$.
13         Stop gradients of $M(\theta_t^k, \zeta_t^k)$ as label.
14         for $i = 1, \cdots, S-1$ do
15             Execute and calculate loss $\hat{F}^k(\theta_t^k \odot \Xi_i, \zeta_t^k)$
16             $loss \leftarrow loss + w_i \hat{F}^k(\theta_t^k \odot \Xi_i, \zeta_t^k)$.
17         end
18         Calculate gradient of $loss$.
19         Update model parameters.                        ▷ Eq. (1)
20     end
21 end
```

$$\theta_{t+1}^k = \theta_t^k - \eta_t \big[w_1 \nabla \hat{F}^k(\theta_t^k \odot \Xi_1, \zeta_t^k) + w_2 \nabla F^k(\theta_t^k \odot \Xi_2, \zeta_t^k)\big], \quad (1)$$

where $w_1 + w_2 = 1$ for constants $w_1, w_2 > 0$. The term $\eta_t > 0$ is a learning rate, and ζ_t^k implies a stochastic input realization. The function $F^k(\theta_t^k \odot \Xi_i, \zeta_t^k)$ is the cross-entropy between the ground truth $y(\zeta_t^k)$ and the logit $M(\theta_t^k \odot \Xi_i, \zeta_t^k)$ of the i-th width configuration, whereas the IPKD function $\hat{F}^k(\theta_t^k \odot \Xi_i, \zeta_t^k)$ is the cross-entropy between the logit $M(\theta_t^k, \zeta_t^k)$ of the full-width configuration and the logit $M(\theta_t^k \odot \Xi_i, \zeta_t^k)$. Fig. 2 corroborates that regardless of the data distributions, SUSTrain achieves high accuracy with fast convergence, as opposed to USTrain that is effective only under IID data distributions (i.e., $\alpha = 1.0$). The details of SUSTrain are in **Algorithm 1**.

IV. Global Model Aggregation with Superposition Coding & Successive Decoding

A. Superposition Coding & Successive Decoding

At a receiver, the signal-to-interference-plus-noise ratio (SINR) is given as $\gamma = gd^{-\beta}P/(\sigma^2 + P^I)$, where P, P^I, and σ^2 stand for the transmission, received interference, and noise powers. In addition, d is a transmitter-receiver distance, $\beta \geq 2$ is a path loss exponent, and g is random small-scale fading. Following the Shannon's capacity formula with a Gaussian codebook, the received throughput R with the bandwidth W is $R = W\log_2(1+\gamma)$ (bits/sec). When the transmitter encodes raw with a code rate u, its receiver successfully decodes the encoded data if $R > u$. The decoding success probability is,

$$\Pr(R \geq u) = \Pr\left(\frac{gd^{-\beta}P}{\sigma^2 + P^I} \geq u'\right). \quad (2)$$

where $u' = 2^{\frac{u}{W}} - 1$. The decoding success probability with SC and SD is given by balancing P and P as elaborated next.

We consider simultaneously conveying S messages from a transmitter to its receiver. These messages are SC-encoded before transmission [7], while the total transmission power budget P is allocated to the i-th message with the amount of $P = \sum_{i=1}^{S} P_i$ transmission power for $i \in [1, S]$. When conveying only a single message, i.e., $S = 1$, there exists no interference at reception, i.e., $P^I = 0$. For $S > 1$, SD determines the interference.

Algorithm 2: SlimFL with SC & SD

```
1  Initialize train parameters $\Theta = \{\theta^1, \cdots, \theta^k, \cdots, \theta^K, \theta^G\}$.
2  Split dataset $\boldsymbol{Z}$ into $K$ datasets $\boldsymbol{Z} = \{Z_1, \cdots, Z_k, \cdots, Z_K\}$.
3  while Training do
4      //Local Model Training (Algorithm 1)
5      for $k = 1, \cdots, K$ do
6          for $\zeta_k$ in $Z_k$ do
7              Update local model parameter $\theta^k$               ▷ Eq. (1)
8          end
9      end
10     //SC&SD-based Server Aggregation (Uplink)
11     if Aggregation Period then
12         $n_{\mathsf{L}} = |\mathsf{H} \cup \mathsf{F}| \leftarrow 0, n_{\mathsf{R}} = |\mathsf{F}| \leftarrow 0, \mathsf{H} \leftarrow \varnothing, \mathsf{F} \leftarrow \varnothing$
13         for $k = 1, \cdots, K$ do
14             $g \leftarrow rand(1)$
15             if $p_2 \leq g < p_1$ then
16                 $\mathsf{H} \leftarrow \mathsf{H} \cup k,\ n_{\mathsf{L}} \leftarrow n_{\mathsf{L}} + 1$
17             end
18             if $g \geq p_2$ then
19                 $\mathsf{F} \leftarrow \mathsf{F} \cup k,\ n_{\mathsf{L}} \leftarrow n_{\mathsf{L}} + 1,\ n_{\mathsf{R}} \leftarrow n_{\mathsf{R}} + 1$
20             end
21         end
22         ▷ Case1. $n_{\mathsf{L}} > 0,\ n_{\mathsf{R}} > 0$
23             $\theta^G \leftarrow \frac{1}{|\mathsf{H} \cup \mathsf{F}|}\sum_{k \in \mathsf{H} \cup \mathsf{F}} \theta^k \odot \Xi + \frac{1}{|\mathsf{F}|}\sum_{k \in \mathsf{F}} \theta^k \odot \Xi^{-1}$
24         ▷ Case2. $n_{\mathsf{L}} > 0,\ n_{\mathsf{R}} = 0$
25             $\theta^G \leftarrow \frac{1}{n_{\mathsf{L}}}\sum_{k \in \mathsf{H}}(\theta^k \odot \Xi)$
26         ▷ Case3. $n_{\mathsf{L}} = n_{\mathsf{R}} = 0$
27             Skip aggregation
28     end
29     //Local Update (Downlink)
30     for $n = 1, \cdots, K$ do
31         $\theta^k \leftarrow \theta^G$
32     end
33 end
```

At its receiver, the SC-encoded message is supposed to be successively reconstructed by first decoding the stronger signal, followed by cancelling out the reconstructed (stronger) signal and then decoding the next stronger signal, i.e., SD, also known as successive interference cancellation [20]. Under Rayleigh fading, the small-scale fading power gain g follows an exponential distribution, i.e., $g \sim \mathsf{Exp}(1)$. Assuming $P_i > P_{i'}$ and $i' > i$, the receiver can successively decode the i-th message while experiencing the rest of the messages as its interference P_i^I, i.e., $P_i^I = gd^{-\beta}\hat{P}_i^I$, where $\hat{P}_i^I \triangleq \sum_{i'=i+1}^{I} P_{i'}$ for $i \leq S-1$, and $\hat{P}_S^I = P_S^I = 0$ as there is no interference for the last message. Let R_i denote the throughput for the i-th message. By substituting P_i^I into (2), the distribution of R_i is cast as, $\Pr(R_i \geq u) = \Pr\left(g \geq \frac{c}{P_i/u' - \hat{P}_i^I}\right)$, where $c = \sigma^2 d^\beta$. Applying this result, the decoding success probability p_i of the i-th message is:

$$p_i = \Pr(R_1 \geq u, R_2 \geq u, \cdots, R_i \geq u) \quad (3)$$

$$= \Pr\left(g \geq \frac{c}{P_1/u' - \hat{P}_1^I}, \cdots, g \geq \frac{c}{P_i/u' - \hat{P}_i^I}\right) \quad (4)$$

$$= \Pr\left(g \geq \max\left\{\frac{c}{P_1/u' - \hat{P}_1^I}, \cdots, \frac{c}{P_i/u' - \hat{P}_i^I}\right\}\right). \quad (5)$$

B. SlimFL Operations

We elaborate the SlimFL and global model aggregation. The notation for SlimFL is as summarized in Tab. VIII. The overall SlimFL operations are described in **Algorithm 2**. The network consists of K devices connected to a parameter server over wireless links. In the uplink from each device to the server, the device applies SC, and the server utilizes SD. To be precise, for every $k \in [1, K]$, the k-th local device has its local dataset $Z^k \in \boldsymbol{Z}$ and its SNN parameter θ^k with 2 width configurations. The global data $\boldsymbol{Z}$ can be either IID or non-IID across devices. Each SNN θ^k is divided into the LH segment $\theta^k \odot \Xi$ and the RH segment $\theta^k \odot \Xi^{-1}$, where $\Xi = \Xi_1$ and $\Xi^{-1} = \Xi_2 - \Xi_1$. The k-th local device is trained with superposition training (lines 4–9), which is written as (1).

The local device uploads the SC-encoded local model θ^k to the server. All local devices transmit two messages (i.e., LH and RH segments) with different transmission power P_1 and P_2 where $P_1 \gg P_2$. After reception, according to (5), the server can successively decode using SD, and obtain: (i) 0.5x model if $g \geq c/(P_1/u' - P_2)$ (lines 15–17); (ii) 1.0x model if the channel fading gain satisfies $g \geq \max\{c/(P_1/u' - P_2), c/(P_2/u')\}$ (lines 18–20); and (iii) otherwise it obtains no model. Accordingly, the server aggregates the RH segments from F of devices, and the LH segments from $\mathsf{H}\cup\mathsf{F}$ of devices.

Hereafter, for the convergence analysis in the next section, we assume that K is sufficiently large such that $|\mathsf{H}\cup\mathsf{F}| \approx Kp_1$ and $|\mathsf{F}| \approx Kp_2$, where p_1 and p_2 are the decoding success probabilities of the LH and RH segments, respectively, given in (5). Consequently, at the t-th communication round, the server constructs a global model θ_t^G as follows:

$$\theta_t^G \leftarrow \frac{1}{Kp_1} \sum\nolimits_{k \in \mathsf{H}\cup\mathsf{F}} \theta_t^k \odot \Xi + \frac{1}{Kp_2} \sum\nolimits_{k \in \mathsf{F}} \theta_t^k \odot \Xi^{-1}. \quad (6)$$

Although SlimFL is flexible enough to incorporate various training and communication techniques, henceforth we limit our scope by considering the following assumptions.

- The downlink decoding is always successful (lines 29–32), ignoring SC and SD. This is partly advocated by the fact that the server (e.g., a base station) has much larger large transmit power than the uplink power.
- The number of local iterations per communication round is 1, therefore omitting the superscript G, i.e., $\theta_t = \theta_t^G$.

These assumptions make the analysis of SlimFL mathematically amenable, as we shall elaborate in the next section.

V. SlimFL Convergence Analysis

To show the convergence of SlimFL, we follow the key derivation techniques utilized in [15], [17] for FedAvg. Nonetheless, SlimFL convergence analysis is non-trivial. One major reason is that the local model updates in (1) and the global model aggregation in (6) include complicated binary masks due to the SNN architecture as well as SC and SD. Therefore, as opposed to FedAvg whose global objective function to be minimized is the weighted average of local loss functions $\{F^k(\theta_t^k)\}$, i.e., empirical risk, the objective function $F(\theta_t)$ of SlimFL is unclear. Alternatively, we define $F(\theta_t)$ based on its gradient $f_t = \nabla F(\theta_t)$ that can be derived through the local and global operations of SlimFL as detailed next.

After the downlink, the device k replaces its local model with the downloaded global model, i.e., $\theta_t^k \leftarrow \theta_t$. Then, the device updates the local model, yielding:

$$\theta_{t+1}^k \leftarrow \theta_t - \eta_t g_t^k, \quad (7)$$

where $g_t^k = \sum_{i=1}^2 \omega_i \nabla F^k(\theta_t \odot \Xi_i, \zeta_t^k)$ follows from (1). For mathematical tractability, here we assume that the soft target of the student can be approximated as the hard target, i.e., $\hat{F}^k(\theta_t \odot \Xi_i, \zeta_t^k) \approx F^k(\theta_t \odot \Xi_i, \zeta_t^k)$.

Next, after the uplink, the server aggregates the updated local models, constructing the global model θ_{t+1}. Applying (7) to (6), the constructed global model is cast as:

$$\theta_{t+1} = \frac{1}{Kp_1} \sum_{k \in \mathsf{H}\cup\mathsf{F}} (\theta_t - \eta_t g_t^k) \odot \Xi + \frac{1}{Kp_2} \sum_{k \in \mathsf{F}} (\theta_t - \eta_t g_t^k) \odot \Xi^{-1} \quad (8)$$

$$= \theta_t - \eta_t \underbrace{\Big(\frac{1}{Kp_1} \sum_{k \in \mathsf{H}\cup\mathsf{F}} g_t^k \odot \Xi + \frac{1}{Kp_2} \sum_{k \in \mathsf{F}} g_t^k \odot \Xi^{-1} \Big)}_{:= f_t}, \quad (9)$$

resulting in f_t in (9), which characterizes $F(\theta_t)$. In (9), the last step can be obtained from $|\mathsf{H} \cup \mathsf{F}| = Kp_1$, $|\mathsf{F}| = Kp_2$, and $\theta_t = \theta_t \odot (\Xi + \Xi^{-1})$.

Hereafter we use the bar notation $\bar{\cdot}$ for the value averaged over $\{\zeta_t^k\}$, and * for indicating the optimum. For the functions F and $\{F^k\}$, we consider the following assumptions that are widely used in the literature [17], [21].

Assumption 1. ***(L-smoothness)*** *F and $\{F^k\}$ are L-smooth, i.e., $F^k(\theta_v) \leq F^k(\theta_w) + (\theta_v - \theta_w)^T \nabla F^k(\theta_w) + \frac{L}{2}\|\theta_v - \theta_w\|^2$ for all $v, w > 0$.*

Assumption 2. ***(μ-strong convexity)*** *F and $\{F^k\}$ are μ-strong convex: i.e., $F^k(\theta_v) \geq F^k(\theta_w) + (\theta_v - \theta_w)^T \nabla F^k(\theta_w) + \frac{\mu}{2}\|\theta_v - \theta_w\|^2$ for all $v, w > 0$.*

Assumption 3. ***(Bounded local gradient variance)*** *The variance of the local gradient $\nabla F^k(\theta^k, \zeta_t^k)$ is bounded within Z_k, which is given as $\mathbb{E}[\|\nabla F^k(\theta^k, \zeta_t^k) - \nabla \bar{F}^k(\theta)\|^2] \leq \sigma_k^2$.*

Inspired by [15], we define $\delta = \frac{1}{K}\sum_{k=1}^K \sigma_k^2$ as a factor that measures the non-IIDness of $\boldsymbol{Z}$. Indeed, $\frac{1}{K}\sum_{k=1}^K (\sigma_k - \frac{1}{K}\sum_{k=1}^K \sigma_k)^2$ is the variance (over k) of the local gradient variance (over Z_k). This characterizes the data distributions over devices, and so does δ without loss of generality.

To prove the convergence of SlimFL, we derive the following two lemmas.

Lemma 1. ***(Bounded global gradient variance)*** *Under Assumption 3, the variance of the global gradient f_t is bounded within $\boldsymbol{Z}$, which is given as $\mathbb{E}\|f_t - \bar{f}_t\|^2 \leq B$ where $B = 4\delta(\frac{1}{p_1} + \frac{1}{p_2})\sum_{i=1}^2 w_i^2$.*

Proof. According to f_t in (9) and Assumption 3, $\|f_t - \bar{f}_t\|^2 = \|\frac{1}{Kp_1}\sum_{k\in\mathsf{H}\cup\mathsf{F}}(g_t^k - \bar{g}_t^k)\odot\Xi + \frac{1}{Kp_2}\sum_{k\in\mathsf{F}}(g_t^k - \bar{g}_t^k)\odot\Xi^{-1}\|^2 \leq \frac{2}{Kp_1}\sum_{k\in\mathsf{H}}\|(g_t^k - \bar{g}_t^k)\odot\Xi\|^2 + \frac{2}{Kp_2}\sum_{k\in\mathsf{F}}\|(g_t^k - \bar{g}_t^k)\odot\Xi^{-1}\|^2 \leq \frac{2}{Kp_1}\sum_{k\in\mathsf{H}}\|g_t^k - \bar{g}_t^k\|^2 + \frac{2}{Kp_2}\sum_{k\in\mathsf{F}}\|g_t^k - \bar{g}_t^k\|^2$, where the first inequality follows from the Cauchy–Schwarz (C-S) inequality, and the last step is because $\|X \odot \Xi\|^2 \leq \|X\|^2$. Similarly, $\|g_t^k - \bar{g}_t^k\|^2 = \|\sum_{i=1}^2 w_i(\nabla F^k(\theta_t, \zeta_t^k) - \nabla F^k(\theta_t)) \odot \Xi_i\|^2 \leq$

$2\sum_{i=1}^2 w_i^2\|\nabla F^k(\theta_t,\zeta_t^k)-\nabla F^k(\theta_t)\|^2$ and taking expectation of both sides gives $\mathbb{E}\|g_t^k-\bar{g}_t^k\|^2 \le 2\sigma_k^2\sum_{i=1}^2 w_i^2$. Combining these results finalizes the proof. □

Lemma 2. ***(Per-round global model progress)*** *Under Assumptions 1 and 2 with a learning rate $\eta_t \le \frac{1}{L}$, the error between the updated global model and its optimum progresses as $\mathbb{E}\|\theta_{t+1}-\theta^*\|^2 \le (1-\mu\eta_t)\mathbb{E}\|\theta_t-\theta^*\|^2+\eta_t^2 B$.*

Proof. According to (9), we have $\|\theta_{t+1}-\theta^*\|^2 = \|\theta_t-\eta_t f_t-\theta^*-\eta_t\bar{f}_t+\eta_t\bar{f}_t\|^2 = \underbrace{\|\theta_t-\theta^*-\eta_t\bar{f}_t\|^2}_{A_1}+\underbrace{2\eta_t\langle\theta_t-\theta^*-\eta_t f_t,\bar{f}_t-f_t\rangle}_{A_2}+\underbrace{\eta_t^2\|f_t-\bar{f}_t\|^2}_{A_3} = \|\theta_t-\theta_t^*\|^2-\underbrace{2\eta_t\langle\theta_t-\theta^*,\bar{f}_t\rangle}_{B_1}+\underbrace{\eta_t^2\|\bar{f}_t\|^2}_{B_2}+A_2+A_3$. Here, $\mathbb{E}[A_2]=0$ due to $\mathbb{E}(f_t)=\bar{f}_t$, and A_3 is bounded according to Lemma 1. Note that $\bar{f}_t=\mathbb{E}[f_t]=\mathbb{E}[\nabla F(\theta_t)]=\nabla\mathbb{E}[F(\theta_t)]$, and $\mathbb{E}[F]$ inherits the μ-strong convexity and L-smoothness from F. By the L-smoothness of $\mathbb{E}[F]$, we have $\|\bar{f}_t\|^2 \le 2L(\mathbb{E}[F(\theta_t)-F(\theta^*)])$, showing the boundness of B_2. Next, by the μ-strong convexity of $\mathbb{E}[F]$, we have $\langle\theta^*-\theta_t,\bar{f}_t\rangle \le \mathbb{E}[F(\theta^*)-F(\theta_t)]-\frac{\mu}{2}\|\theta_t-\theta^*\|^2$, proving the boundness of B_1. Applying the bounds of B_1 and B_2, we obtain $A_1 \le (1-\mu\eta_t)\|\theta_t-\theta^*\|^2-2\eta_t(1-L\eta_t)\mathbb{E}[F(\theta_t)-F(\theta^*)]$, where the last term on the RHS vanishes for $\eta_t<\frac{1}{L}$. Taking the expectation at both sides completes the proof. □

Now we are ready to prove our main theorem.

Theorem 1. ***(SlimFL Convergence)*** *Under Assumptions 1–3 with the learning rate $\eta_t=\frac{2}{\mu t+2L-\mu}$, one has*

$$\mathbb{E}[F(\theta_t)]-F^* \le \frac{L}{\mu}\cdot\frac{\mu L\Delta_1+2B}{\mu t+2L-\mu}, \tag{10}$$

where $B=4\delta(\frac{1}{p_1}+\frac{1}{p_2})\sum_{i=1}^2 w_i^2$ and $\Delta_t \triangleq \mathbb{E}\|\theta_t-\theta^*\|^2$. Therefore, $\mathbb{E}[F(\theta_t)]$ converges to F^* as $t\to\infty$.

Proof. Since $\eta_t=\frac{2}{\mu t+2L-\mu}\le\frac{1}{L}$, applying Lemma 2, we have $\Delta_{t+1}\le(1-\mu\eta_t)\Delta_t+\eta_t^2 B$. By induction, we aim to show that $\Delta_t\le\frac{v}{t+2\kappa-1}$ where $\kappa=\frac{L}{\mu}$ and $v=\max\{2\kappa\Delta_1,4B/\mu^2\}$ as elaborated next. By the definition of v, it is trivial that $\Delta_1\le\frac{v}{2\kappa}$. Assuming that $\Delta_{t'}\le\frac{v}{t'+2\kappa-1}$ holds, we have $\Delta_{t'+1}\le(1-\mu\eta_{t'})\Delta_{t'}+\eta_{t'}^2 B\le\left(1-\frac{2}{t'+2\kappa-1}\right)\frac{v}{t'+2\kappa-1}+\frac{4B/\mu^2}{(t'+2\kappa-1)^2}=\frac{(t'+2\kappa-2)v-(v-4B/\mu^2)}{(t'+2\kappa-1)^2}\le\frac{t'+2\kappa-2}{(t'+2\kappa-1)^2}v\le\frac{v}{t'+2\kappa}$, which proves that $\Delta_t\le\frac{v}{t+2\kappa-1}$. For $t=1$, we obtain $v=\max\{2\kappa\Delta_1,\frac{4B}{\mu^2}\}\le2\kappa\Delta_1+\frac{4B}{\mu^2}$. Finally, by the L-Smoothness of F, one has $\mathbb{E}[F(\theta_t)]-F^*=\mathbb{E}[F(\theta_t)-F(\theta^*)]\le\frac{L}{2}\mathbb{E}\|\theta_t-\theta^*\|^2$. Applying Lemma 2 with the aforementioned results, we have $\mathbb{E}\|\theta_t-\theta^*\|^2\le\frac{v}{t+2\kappa-1}\le\frac{2}{\mu}\cdot\frac{\mu L\Delta_1+2B}{\mu t+2L-\mu}$, which completes the proof of the theorem. □

The result of Theorem 1 exhibits several insightful characteristics of SlimFL as follows.

Robustness to Poor Channels. In (10), we observe that aggregating more 0.5x and 1.0x models (i.e., increasing p_1 and p_2) equally contributes to reducing the global optimality gap. Therefore, aggregating 0.5x models can complement the frequent decoding failures of 1.0x models under poor channels.

Failure under Extremely Poor Channels. Consider an extremely poor channels where the server is unable to decode 1.0x models while aggregating only 0.5x models (i.e., $p_2\approx0$ and $p_1>0$). In this case, the optimality gap diverges although it aggregates 0.5x models. Under such channel conditions, SC becomes useless, and vanilla FL with only 0.5x models is preferable to SlimFL.

Robuestness to non-IID Data. The optimality gap increases with δ (i.e., more non-IID). The increased gap can be counteracted by aggregating not only 1.0x models but also 0.5x models, as opposed to vanilla FL that benefits only from aggregating either 0.5x models or 1.0x models.

Judging from the aforementioned observations, we conclude that SlimFL is preferable for non-IID data distributions and moderately poor channel conditions where $0\ll p_1,p_2<1$. For an extremely good (i.e., $p_2\approx1$) or an extremely poor (i.e., $p_1\approx0$) channel conditions, vanilla FL with only 1.0x models or 0.5x models is preferable, respectively. These favorable conditions and effectiveness of SlimFL will be corroborated by simulation in Sec. VI.

Furthermore, Theorem 1 provides the design guidelines on SC and ST as elaborated in the following two propositions.

Proposition 1 (**Optimal SC power allocations**). *Consider the SC power allocation ratio $\lambda\in(0.5,1]$ such that $P_1=\lambda P$ and $P_2=(1-\lambda)P$. If $\lambda\gg\max\{0.5,cu'(1+u')/P\}$, the optimal SC power allocation ratio that minimizes the RHS of (10) is given as $\lambda^*=\frac{u'+\sqrt{1+u'}-1}{u'}$.*

Proof. Define $D\triangleq\frac{1}{p_1}+\frac{1}{p_2}$. According to the RHS of (10), λ^* minimize D. Since $P_1>P_2$, we have $D=\exp\left(\frac{c}{\lambda P/u'-(1-\lambda)P}\right)+\exp\left(\frac{c}{(1-\lambda)P/u'}\right)$. If $\lambda\gg cu'(1+u')/P$, we can approximate the both terms in D using the first-order Taylor expansion, yielding $D\approx2+\frac{c}{\lambda P/u'-(1-\lambda)P}+\frac{c}{(1-\lambda)P/u'}$. The approximated D is convex, and the optimum is given by the first order necessary condition. □

Note the condition $\lambda\gg\max\{0.5,cu'(1+u')/P\}$ above can be satisfied under small model sizes (e.g., $t'\to0$), large bandwidth (e.g., $W\to\infty$), good channel conditions (e.g., $\sigma^2\to0$), and/or large total transmit power budget (e.g., $P\to\infty$). For practical scenarios, by simulation we confirm that the analytic optimum is indistinguishable from the numerical optimum as shown in Fig 4.

Proposition 2 (**Optimal ST weights**). *The optimal ST weights that minimize the RHS of (10) are given as $w_1^*=w_2^*=1/2$.*

Proof. The RHS of (10) is minimized at the minimum of $\sum_{i=1}^2 w_i^2$. By the C-S inequality, we have $\sum_{i=1}^2 w_i^2\ge\frac{1}{2}(\sum_{i=1}^2 w_i)^2$. By the condition $\sum_{i=1}^2 w_i=1$ in (1) and the equality condition of the AM-GM inequality, one has the desired result. □

The impact of λ^* and w_i^* will be shown by simulation in Fig. 5 in the next section.

TABLE II
SIMULATION PARAMETERS.

Description	Value
Initial learning rate (η_0)	10^{-3}
Optimizer	Adam
Distance (d)	100 [m]
Path loss exponent (β)	2.5
Bandwidth per device (W)	75×10^6 [Hz]
Central frequency (f_c)	5.9 [GHz]
Uplink transmission power (P)	23 [dBm]
Noise power spectrum (N_0)	-169 [dB/Hz]

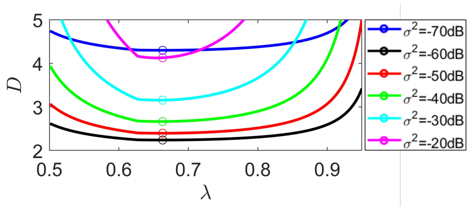
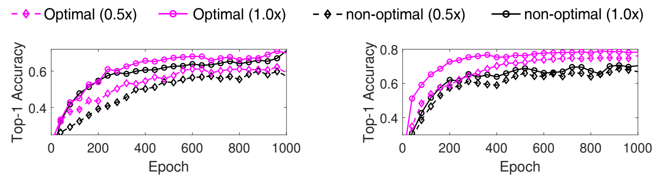
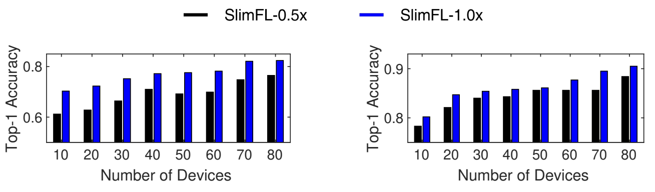


Fig. 4. SC power allocation ratio λ versus D $(:= 1/p_1 + 1/p_2)$.



(a) SC power allocation ratio (λ). (b) ST weight (w_i)

Fig. 5. Top-1 accuracy under optimal and non-optimal design parameters: (a) $\lambda^* = 0.663$ and $\lambda = 0.8$, and (b) $w_1^* = w_2^* = 0.5$ and $w_1 = 0.3$, and $w_2 = 0.7$) with $\alpha = 0.1$.



(a) Top-1 accuracy ($\alpha = 0.1$) (b) Top-1 accuracy ($\alpha = 1.0$)

Fig. 6. Top-1 accuracy with different number of devices.

VI. EXPERIMENTS

To show the effectiveness and feasibility of SlimFL, we present the performance of SlimFL exploiting SC and SD compared to its Vanilla FL counterpart without SC nor SD, in terms of accuracy, communication efficiency, and energy efficiency, as well as their robustness to various channel conditions and non-IID data distributions.

A. *Experimental Setup*

Baselines. Our goal is enabling each device to obtain both large and small models so as to cope with its large and small energy levels in future. To this end, by leveraging SNNs with SC and SD, SlimFL simultaneously exchanges and trains 0.5x and 1.0x models by consuming the per-device bandwidth W, uplink transmission power P. This is compared with a Vanilla FL baseline, *Vanilla FL*-1.5*x*. Due to the lack of width-adjustable SNNs, each device in Vanilla FL-1.5x separately runs fixed-width 0.5x and 1.0x models, referred to as *Vanilla FL*-0.5*x* and *Vanilla FL*-1.0*x*, respectively. Without SC nor SD, the device exchanges both 0.5x and 1.0x models separately. In brief, Vanilla FL-1.5x is tantamount to simultaneously running the two federated averaging operations separately for 0.5x and 1.0x models by doubling the bandwidth, transmission power, and computing resources. For clarity, we report the performance of Vanilla FL-0.5x and Vanilla FL-1.0x individually if available (i.e., accuracy, received bits), and otherwise we report only Vanilla FL-1.5x (i.e., energy cost).

Simulation Settings. We consider a classification task by default with the Fashion MNIST dataset. Following the method proposed in [22], the non-IIDness of the dataset distribution across devices is controlled by the Dirichlet distribution with its concentration parameter $\alpha \in \{0.1, 1.0, 10\}$, where a lower α is more non-IID distributed (i.e., more imbalanced numbers of samples over labels across devices), as visualized in Fig. 3. A single round of uplink and downlink communications is followed by every single local training epoch. The communication channels over different devices are orthogonal in both uplink and downlink. The small-scale fading gain g for each channel realization follows an exponential distribution $g \sim \mathsf{Exp}(1)$, i.e., Rayleigh fading [12]. Communication hyperparameters are summarized in Tab. II.

B. *Guidelines for SlimFL*

By convergence analysis, we have guidelines in respect to optimal power allocation (Proposition 1) and determination of weight parameters in superposition learning (Proposition 2).

Optimal SC Power Allocations. We propose a method of adjusting the parameter ω_l of ST to optimize our proposed model. Fig. 4 represents the numerical results of optimal λ^*. In Proposition 1, λ^* is calculated via the derivative of Tayler expansion. From analytical solution, we derive the optimal power allocation factor as $\lambda^* = 0.662$, which is exactly same value of numerical optimum. In order to verify the guideline from Proposition 1, the simulation is conducted with the baseline ($\lambda = 0.8$) in non-IID setting ($\alpha = 0.1$). Fig. 5(a) shows the result of experiment. The top-1 accuracy with λ^* shows 6.4%, 8.8% higher accuracy than the top-1 accuracy in 0.5x and 1.0x, respectively. In other words, Proposition 1 provides influential guideline in SlimFL.

Optimal ST Weights. In Proposition 2, SlimFL has tight bound when $w_1 = \cdots = w_S = \frac{1}{S}$. Since we consider $S = 2$ in our proposed scheme, all hyperparameters constituting ST should be 0.5, i.e., $(w_1^*, w_2^*) = (0.5, 0.5)$. To verify Proposition 2, we design baseline as $(w_1, w_2) = (0.3, 0.7)$. Fig. 5(b) shows performance difference according to different ω_i. In optimal ST settings, top-1 accuracy achieves 78% whereas, baseline achieves 69%. Thus, the guideline for ST positively effects the performance of SlimFL.

Scalability. As Fig. VI-B represents, the accuracy of SLimFL improves as the number of federating devices increases. The SlimFL-0.5x accomplished the accuracy up to 79%, and the SlimFL-1.0x accomplished the accuracy up to 85%. In addition, with the non-iidness ($\alpha = 0.1$), and with over the number of 70 federating local devices, SlimFL-0.5x shows

TABLE III
ACCURACY UNDER DIFFERENT CHANNEL CONDITIONS AND α.

Method	Top-1 Accuracy (%)					
	Good			Poor		
	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$
SlimFL-0.5x	54 ± 2.2	83 ± 1.0	85 ± 1.0	56 ± 2.4	82 ± 1.7	85 ± 1.1
SlimFL-1.0x	59 ± 2.3	85 ± 1.1	87 ± 1.1	65 ± 2.9	84 ± 1.4	87 ± 0.9
Vanilla FL-0.5x	45 ± 5.9	84 ± 1.1	85 ± 1.0	39 ± 8.3	83 ± 1.2	85 ± 0.9
Vanilla FL-1.0x	69 ± 5.8	85 ± 4.0	86 ± 4.3	55 ± 9.2	80 ± 6.0	82 ± 4.7

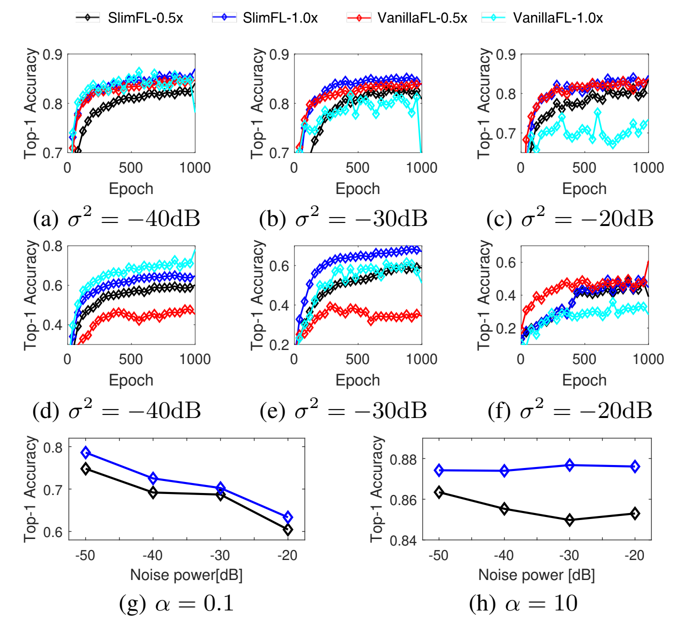


Fig. 7. Test accuracy in various channel noise conditions (on average). (a–c) are with $\alpha = 1$, (d–f) and (h) are with $\alpha = 0.1$, (f) is with $\alpha = 10$.

higher accuracy than the SlimFL-1.0x with 20 federating local devices. Based on the experimental result, it is expected that optimality can be achieved by adjusting the number of local devices and the width through SlimFL adaptation when configuring an FL system based on non-IID datasets.

C. Performance of SlimFL

We constructed an experiment to verify the performance of SlimfL compared to Vanilla FL in the environments with various communication conditions and non-iid settings.

Robustness to Non-IID Data. As illustrated in Fig. 7(d–f) and Tab. III, SlimFL-0.5x shows a stable convergence under the conditions of α. Vanilla FL-0.5x and Vanilla FL-1.0x exhibit the std of 8.3 and 9.2, in poor channel condition and with the non-IID dataset ($\alpha = 0.1$). On the contrary, both SlimFL-0.5x and SlimFL-1.0x exhibit the std of 2.4 and 2.9 at top-1 accuracy. This tendency holds even when $\alpha = 1$, $\alpha = 10$. SlimFL-1.0x and SlimFL-0.5x exhibit lower variation than Vanilla FL-1.0x and Vanilla FL-0.5x. This underscores the robustness of SlimFL to non-IID data in poor channels.

Robustness to Poor Channels. Fig. 7 and Tab. III show that both SlimFL and Vanilla FL achieve high accuracy in good channel conditions. However as the channel condition deteriorates from good to poor channels, Fig. 7(c,f) and Tab. III illustrate that the maximum accuracy of Vanilla FL-1.0x at $\alpha = 10$ drops from 86% to 82%. Meanwhile, the accuracy

TABLE IV
COMPUTING COSTS AND TRANSMISSION POWER OF UL-MOBILENET.

Description		1.0x	0.5x
Computation	MFLOPS / round	2.76	0.79
	# of parameters	$4,586$	$2,293$
	Bits / round	$172,688$	$86,344$
Transmission Power (P) [mW]		132.1	67.4

TABLE V
TRANSMISSION AND COMPUTING COSTS PER COMMUNICATION ROUND.

Metric	SlimFL	Vanilla FL-1.5x
Communication Cost [mW/Round]	199.5	399.1
Computation Cost [MFLOPS/Epoch]	3.56	3.56

TABLE VI
SUCCESSFULLY DECODED BITS OF SLIMFL, AND VANILLA FL.

Decoding Success Bits [MBytes]	SlimFL			Vanilla FL-0.5x		Vanilla FL-1.0x	
	0.5x	1.0x	drop	0.5x	drop	1.0x	drop
$\sigma^2 = -30$dB	1.96	198.45	5.46	102.21	0.72	200.30	5.56
$\sigma^2 = -40$dB	18.32	130.10	57.44	93.87	9.06	144.93	60.96

of Slim-FL-1.0x keeps the same maximum accuracy 87% at $\alpha = 10$ under both good and poor channels. What is more, at $\alpha = 0.1$, SlimFL-1.0x even achieves 18% higher top-1 accuracy than Vanilla FL-1.0x that consumes more communication and computing costs. Furthermore, the std of Vanilla FL-1.0x's top-1 accuracy increases by up to 59% as channel condition deteriorates, whereas that of SlimFL increases by only up to 31%. These results advocate the robustness of SlimFL against poor channels, as well as its robustness to non-IID data distributions (low α) and communication efficiency.

D. Communication and Energy Efficiency

In order to figure out the efficiency of communication and computation, we first calculate computation cost for feed-forwarding in UL-MobileNet [23] and communication per one communication round.

Communication Efficiency. The total amounts of transmitted bits between 10 devices and server in ideal channel conditions (i.e., always successful decoding) are 205.8MBytes for SlimFL and Vanilla FL-1.0x, and 102.9MBytes for Vanilla FL-0.5x. Tab. VI shows that SlimFL achieves up to 3.52% less dropped bits than Vanilla FL-1.0x, thanks to the use of SC and SD. The reduced dropped bits of SlimFL can be found by the successfully decoded bits of 0.5x models that cannot be simultaneously received under Vanilla FL-1.0x. Note that SlimFL decodes less 1.0x model bits than Vanilla FL-1.0x, as a part of transmission power of SlimFL is allocated to 0.5x models. In return, SlimFL not only receives 1.0x models but also 0.5x models simultaneously. The additionally received 0.5x models correspond to the LH parts of the 1.0x models, which therefore improve the accuracy and convergence speed of both 0.5x and 1.0x models. SlimFL enjoys the aforementioned benefits while consuming only the half of the transmission power and bandwidth compared to Vanilla FL-1.5x, as illustrated in Tab. VI, corroborating its communication efficiency.

TABLE VII
TOTAL COMPUTATION COST AND TRANSMISSION POWER OF SLIMFL AND VANILLA FL-1.5X IN VARIOUS NON-IIDNESS ($\alpha = 0.1, 1.0, 10$).

Metric	non-IIDness	SlimFL Good	SlimFL Poor	Vanilla FL-1.5x Good	Vanilla FL-1.5x Poor
Communication Cost [W]	$\alpha = 0.1$	71.0	57.3	158.8	196.8
	$\alpha = 1.0$	8.5	10.4	15.8	36.7
	$\alpha = 10$	3.03	3.51	10.2	25.4
Computation Cost [GFLOPS]	$\alpha = 0.1$	1.27	1.02	1.88	2.41
	$\alpha = 1.0$	0.15	0.18	0.22	0.51
	$\alpha = 10$	0.05	0.06	0.14	0.35

Energy Efficiency. Thus far we have measured the performance of SlimFL after training with a fixed 1000 epochs. Here, we measure the energy expenditure until convergence, where the training convergence is defined by the moment when the standard deviation (std) of test accuracy is below a target threshold and the minimum test accuracy becomes higher than the average test accuracy in 100 consecutive rounds. To measure the convergence of models, we define the reference values of the mean μ_{Ref} as 80% and σ_{Ref} as 7.2%, respectively. We define the convergence when average of Top-1 accuracy for 100 consecutive epochs is higher than μ_{Ref}, and the average std is lower than σ_{Ref}. Given the communication and computing energy costs per round in Tab. V, Tab. VII compares the total energy costs of SlimFL and Vanilla FL-1.5x until convergence. The results show that on average, SlimFL achieves 3.6x less total computing cost with 2.9x lower total communication cost until convergence. Such higher energy efficiency comes from the faster convergence of SlimFL even under non-IID and/or poor channel conditions due to SC and SD.

VII. CONCLUSION

Existing FL solutions cannot cope flexibly with different devices having heterogeneous levels of available energy and channel throughput. To tackle this problem, we propose a novel framework of FL over SNNs, SlimFL, by developing ST for local SNN training and exploiting SC for the trained model aggregation. Extensive experiments verify that SlimFL is a communication and energy efficient solution under various communication environments and data distributions. Particularly under poor channel conditions and non-IID data distributions, SlimFL even achieves higher accuracy and faster convergence with lower energy expenditure than its vanilla FL counterpart consuming 2x more communication resources. Additionally incorporating more width configurations and local iterations could be interesting topics for future research.

ACKNOWLEDGMENT

This research was funded by IITP 2021-0-00467. The first three authors are equally contributed (first authors). S. Jung, J. Park, and J. Kim are corresponding authors.

APPENDIX A
LOCAL SNN TRAINING ALGORITHM

SlimTrain in [19] and USTrain in [5] are described by **Algorithm 3** and **Algorithm 4**, respectively. In essence, SlimTrain and USTrain both utilize alternating methods, and USTrain utilizes IPKD and the sandwich rule that are effective when each SNN has more than two width configurations. We only consider 2 width configurations, making the sandwich rule unfit for our case. Therefore, ignoring the sandwich rule, our proposed SUSTrain only utilizes IPKD while additionally exploiting ST. Fig. 2 shows that SUSTrain outperforms SlimTrain and USTrain under both IID and non-IID data.

Algorithm 3: SlimTrain [19]

```
Define switchable width list for slimmable network $M$, for example, $[0.25, 0.5, 0.75, 1.0]\times$.
Initialize shared convolutions and fully-connected layers for slimmable network $M$.
Initialize independent batch normalization parameters for each width in switchable width list.
for $i = 1, ..., n_{iters}$ do
    Get next mini-batch of data $x$ and label $y$.
    Clear gradients of weights, $optimizer.zero\_grad()$.
    for width in switchable width list do
        Switch the batch normalization parameters of current width on network $M$.
        Execute sub-network at current width, $\hat{y} = M'(x)$.
        Compute loss, $loss = criterion(\hat{y}, y)$.
        Compute gradients, $loss.backward()$.
    end
    Update weights, $optimizer.step()$.
end
```

Algorithm 4: USTrain [5]

```
Define width range, for example, $[0.25, 0.5, 0.75, 1.0]$x.
Define n as number of sampled widths per training iteration, for example, $n = 4$.
Initialize training settings of shared network $M$.
for $(t = 1, ..., T_{iters})$ do
    Get next mini-batch of data $x$ and label $y$).
    Clear gradients, $optimizer.zero\_grad()$.
    Execute full-network, $y' = M(x)$.
    Compute loss, $loss = criterion(y', y)$.
    Accumulate gradients, $loss.backward()$.
    Stop gradients of $y'$ as label, $y' = y'.detach()$.
    Add smallest width to width samples.
    for width in width samples do
        Execute sub-network at width, $\hat{y} = M'(x)$.
        Compute loss, $loss = criterion(\hat{y}, y')$.
        Accumulate gradients, $loss.backward()$.
    end
    Update weights, $optimizer.step()$.
end
```

TABLE VIII
LIST OF NOTATIONS

Notation	Description
T	Total iteration steps.
S	The number of SNN width configurations.
θ^G	Global model parameter vector.
θ^k	Local model parameter vector of the k-th device.
$\mathcal{K}$	A set of devices ($\mathcal{K} = \{1, \cdots, k, \cdots, K\}$).
Ξ	A binary mask to extract weight parameters of an LH segment.
Ξ^{-1}	A binary mask to extract weight parameters of an RH segment.
H	A set of successfully decoded LH segments.
F	A set of successfully decoded RH segments.
n_{L}	The number of successfully decoded LH segments.
n_{R}	The number of successfully decoded RH segments.
p_1	The decoding success probability of an LH segment.
p_2	The decoding success probability of an RH segment.
$\boldsymbol{Z}$	Entire dataset.
Ξ_i	A binary mask to extract weight parameters of the i-th smallest model.
ω_i	Positive constant for updating full model via the i-th smallest model.
η_t	Learning rate at the iteration t.
ζ_t^k	The local data sampled from k-th user at the iteration t.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, FL, USA, April 2017, pp. 1273–1282.

[2] N. H. Tran, W. Bao, A. Y. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, April 2019, pp. 1387–1395.

[3] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proceedings of the IEEE*, vol. 107, no. 11, pp. 2204–2239, 2019.

[4] J. Park, S. Samarakoon, A. Elgabli, J. Kim, M. Bennis, S. Kim, and M. Debbah, "Communication-efficient and distributed learning over wireless networks: Principles and applications," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 796–819, May 2021.

[5] J. Yu and T. S. Huang, "Universally slimmable networks and improved training techniques," in *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, South Korea, October 2019, pp. 1803–1811.

[6] M. K. Ozdemir and H. Arslan, "Channel estimation for wireless OFDM systems," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 2, pp. 18–48, Second Quarter 2007.

[7] T. Cover, "Broadcast channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 2–14, January 1972.

[8] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. of International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, May 2016.

[9] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. of the Conference on Neural Information Processing Systems (NIPS) Deep Learning and Representation Learning Workshop*, December 2015, pp. 1–9.

[10] D. Kim, J. Kim, J. Kwon, and T.-H. Kim, "Depth-controllable very deep super-resolution network," in *Proc. of IEEE International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary, July 2019.

[11] H. Hu, D. Dey, M. Hebert, and J. A. Bagnell, "Learning anytime predictions in neural networks via adaptive loss balancing," in *Proc. of AAAI Conference on Artificial Intelligence (AAAI)*, Honolulu, Hawaii, USA, January 2019, pp. 3812–3821.

[12] D. N. C. Tse and P. Viswanath, *Fundamentals of Wireless Communications*. Cambridge University Press, 2005.

[13] Z. Ding, Y. Liu, J. Choi, Q. Sun, M. Elkashlan, I. Chih-Lin, and H. V. Poor, "Application of non-orthogonal multiple access in LTE and 5G networks," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 185–191, February 2017.

[14] J. Wang, Z. Charles, Z. Xu, G. Joshi, H. B. McMahan, M. Al-Shedivat, G. Andrew, S. Avestimehr, K. Daly, D. Data *et al.*, "A field guide to federated optimization," *arXiv preprint, abs/2107.06917*, July 2021.

[15] A. Khaled, K. Mishchenko, and P. Richtárik, "Tighter theory for local SGD on identical and heterogeneous data," in *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, Palermo, Sicily, Italy, August 2020, pp. 4519–4529.

[16] L. Mangasarian, "Parallel gradient distribution in unconstrained optimization," *SIAM Journal on Control and Optimization*, vol. 33, no. 6, pp. 1916–1925, November 1995.

[17] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-iid data," in *Proc. of International Conference on Learning Representation (ICLR)*, Addis Ababa, Ethiopia, April 2020.

[18] A. Mohtashami, M. Jaggi, and S. U. Stich, "Simultaneous training of partially masked neural networks," *arxiv preprint, abs/2106.08895*, June 2021.

[19] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, "Slimmable neural networks," in *Proc. of International Conference on Learning Representation (ICLR)*, New Orleans, LA, USA, May 2019.

[20] J. Choi, "Joint rate and power allocation for NOMA with statistical CSI," *IEEE Trans. Communi.*, vol. 65, no. 10, pp. 4519–4528, October 2017.

[21] S. U. Stich, "Local SGD converges fast and communicates little," in *Proc. of International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, April/May 2018.

[22] T. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *Conference on Neural Information Processing Systems (NeurIPS) Workshop on Federated Learning for Data Privacy and Confidentiality (available on arXiv preprint, abs/1909.06335)*, September 2019.

[23] D. Hernandez and T. B. Brown, "Measuring the algorithmic efficiency of neural networks," *arXiv preprint, abs/2005.04305*, May 2020.