Supplementary Material for "Class Balanced Adaptive Pseudo Labeling for Federated Semi-Supervised Learning"

Ming Li Qingli Li Yan Wang*
Shanghai Key Laboratory of Multidimensional Information Processing, East China Normal University
lm1640362161@gmail.com, qlli@cs.ecnu.edu.cn, ywang@cee.ecnu.edu.cn

A. Outline

In this supplementary material, we provide more details for our paper titled "Class Balanced Adaptive Pseudo Labeling for Federated Semi-Supervised Learning", organized into the following sections:

- Appendix A gives the overall training procedure of our CBAFed in Algorithm 1.
- Appendix B gives detailed review of traditional pseudo labeling methods.
- Appendix C gives more discussions of proposed residual weight connection.
  -Appendix C.1 gives detailed comparison of proposed residual weight connection and mean teachers [10].
  -Appendix C.2 empirically shows that skip connection is important for the success of residual weight connection.
- Appendix D gives proof of theorem 3.1 in paper.
- Appendix E gives more discussions of class balanced adaptive pseudo labeling.
- Appendix F gives experiment details and more experimental results.
  -Appendix F.1 gives detailed dataset splitting and pre-processing.
  -Appendix F.2 gives the implementation details of experiments in our paper.
  -Appendix F.3 gives discussions of different training strategies on labeled clients when number of labeled clients is Two.
  -Appendix F.4 gives experimental results on AUC metric.
  -Appendix F.5 gives experimental results on partially labeled clients.

A. Algorithm 1

To further illustrate our method, the overall training procedure is summarized in Algorithm 1.

B. Review of Classic Batch-based Pseudo Labeling

In semi-supervised learning, for unlabeled data, traditional approaches [1,9,12], such as Fixmatch [9] and Flexmatch [12], use original data or their weak augmented version in one batch to generate pseudo labels. These labels are adopted to supervise model's training. Let $\mathcal{X}^l = \{(X_i^l, y_i^l)\}_{i=1}^B$ be a batch of B labeled data and $\mathcal{X}^u = \{X_i^u\}_{i=1}^{\epsilon B}$ be a batch of ϵB unlabeled data, where ϵ is the hyperparameter that determines the relative sizes of $\mathcal{X}^l$ and $\mathcal{X}^u$. Let $p_m(y|\theta(X))$ be the predicted class distribution produced by the model θ for input X. The pseudo label $\hat{q}_i^u$ is calculated by:

$$\hat{q}_i^u = \arg\max p_m(y|\theta(X_i^u)), i = 1, 2, \cdots, \epsilon B. \tag{1}$$

*Corresponding Author.

Algorithm 1: Class Balanced Adaptive Pseudo Labeling

Input: Initialized global model θ_1^G, dataset D_i in C_i, $i = 1, 2, \cdots, n+m$, threshold base τ, upper bound threshold τ_h, hyper-parameter α_1, α_2, β, skip epoch s, local supervised training epoch J, warm up stage communication rounds P and total communication rounds T .

Output: Global model θ_{T+1}^G

for $t \leftarrow$ 1 *to* P **do**
  In local clients
  for $\ell \leftarrow$ 1 *to* m *in parallel* **do**
    $\theta_t^\ell \leftarrow \theta_t^G$
    $\theta^{res} \leftarrow \theta_t^G$
    for $j \leftarrow$ 1 *to* J **do**
      $\theta_t^\ell \leftarrow$use Eq. 5 to update θ_t^ℓ
      if $j\%s = 0$ **then**
        $\theta_t^\ell = \alpha_1 \theta^{res} + (1-\alpha_1)\theta_t^\ell$
        $\theta^{res} = \theta_t^\ell$
    return θ_t^ℓ, $\sigma_t^\ell(1),\ \sigma_t^\ell(2),\ \cdots,\ \sigma_t^\ell(C)$, $|D_\ell|$
  In central server
  $\theta_{t+1}^G \leftarrow \sum_{\ell=1}^m \frac{|D_\ell|}{\sum_{i=1}^m |D_i|}\theta_t^\ell$
  if $t\%s = 0$ **then**
    $\theta_{t+1}^G = \alpha_1 \theta_{t+1-s}^G + (1-\alpha_1)\theta_{t+1}^G$
In central server
set $\sigma_t^u(c) = 0$ and use equations 10∼14 to compute class distribution $\widetilde{p}_t(c)$ and threshold $\mathcal{T}_{t+1}(c), c = 1, 2, \cdots C$.
for $t \leftarrow$ *P+1 to T* **do**
  In local clients
  for $\ell \leftarrow$ *1 to m in parallel* **do**
    $\theta_t^\ell \leftarrow \theta_t^G$
    $\theta^{res} \leftarrow \theta_t^G$
    for $j \leftarrow$ 1 *to* J **do**
      $\theta_t^\ell \leftarrow$use Eq. 5 to update θ_t^l
      if $j\%s = 0$ **then**
        $\theta_t^\ell = \alpha_1 \theta^{res} + (1-\alpha_1)\theta_t^\ell$
        $\theta^{res} = \theta_t^\ell$
    return θ_t^ℓ, $\sigma_t^\ell(1)$, $\sigma_t^\ell(2), \cdots, \sigma_t^\ell(C)$, $|D_\ell|$
  for $\mu \leftarrow m+1$ *to n+m in parallel* **do**
    $\theta_t^\mu \leftarrow \theta_t^G$
    $\mathcal{T}_{t+1}(c) \leftarrow \mathcal{T}_{t+1}(c), c = 1, 2, \cdots, C$
    $\widetilde{p}_t(c) \leftarrow \widetilde{p}_t(c), c = 1, 2, \cdots, C$
    use Eq. 17, 19 and 20 to compute new training dataset D_μ^{train}
    use Eq. 22 to compute $\sigma_t^\mu(c), c = 1, 2, \cdots, C$
    $\theta_t^\mu \leftarrow$ use Eq. 21 to update θ_t^μ
    return θ_t^μ, $\sigma_t^\mu(1),\ \sigma_t^\mu(2),\ \cdots,\ \sigma_t^\mu(C)$, $|D_\mu^{train}|$
  In central server
  use Eq. 23 to compute scaling factor w_t^i
  $\theta_{t+1}^G \leftarrow \sum_{i=1}^{n+m} w_t^i \theta_t^i$
  if $t\%s = 0$ **then**
    $\theta_{t+1}^G = \alpha_2 \theta_{t+1-s}^G + (1-\alpha_2)\theta_{t+1}^G$
  use equations 10 ∼ 14 to compute class distribution $\widetilde{p}_t(c)$ and threshold $\mathcal{T}_{t+1}(c), c = 1, 2, \cdots C$.
In central server
return θ_{T+1}^G

The training losses are: $\mathcal{L} = \mathcal{L}^l + \lambda\mathcal{L}^u$, where loss $\mathcal{L}^l$ for labeled data (supervised loss) and loss $\mathcal{L}^u$ for unlabeled data (unsupervised loss) are computed by:

$$\mathcal{L}^u = \frac{1}{\epsilon B}\sum_{B=1}^{\epsilon B} \mathbf{1}(\max(p_m(y|\theta(X_i^u)) > \tau) H(\hat{q}_i^u, p_m(y|\theta(X_i^u))), \tag{2}$$

$$\mathcal{L}^l = \frac{1}{B}\sum_{i=1} H(y_i^l, p_m(y|\theta(X_i^l))), \tag{3}$$

where $H(,)$ is an entropy function , τ is the threshold and λ is a trade-off parameter. Noted that in batch-based pseudo labeling, the unsupervised loss is applied on a selected subset of *current batch*, which supervised by the pseudo labels computed in current batch, as shown in Eq. 2.

C. More Discussions of Residual Weight Connection

C.1. Comparison of Residual Weight Connection and Mean Teachers

In this section, we give more comparisons of our residual weight connection and mean teachers [10]. The main differences lie in three folds: 1) The update strategy is different. In mean teachers [10], the update of teacher model is conducted after each training iteration with the student model optimized by the consistency loss. In our residual weight connection, there is only one model and the there is one skip epoch between every residual update. Residual update is done after every s epochs with a weighted average of the model's weight and weight of the same model in the s epoch before this epoch. 2) The usage scenarios are different. The mean teachers model [10] is designed for semi-supervised learning. It mainly uses consistency regularization to train the model. Our residual weight connection is designed for fully supervised learning in labeled clients or model aggregation in the central server. 3) The scale of parameter α is different. In mean teacher [10], α is set to 0.999, so the impact of the student model's weight is low for the model update. In our residual weight connection, α is set to 0.8 or 0.5. So the impact of the former model's weight is high in residual update.

C.2. Skip Connection is Important

In our residual weight connection, we define a skip epoch s between residual update, i.e., residual weight connection every s epochs. In this section, we will give experimental results to show this skip connection is important for the success of residual weight connection. We conduct experiments on SVHN dataset and set the same setting as that in Sec. 4.1. Fig. 1 shows the test accuracy curve of FedAVG [7] training on on one labeled client with $s = 1$ and $s = 5$. As can be shown in the figure, with the skip connection ($s = 5$), the model eventually achieves better performance.

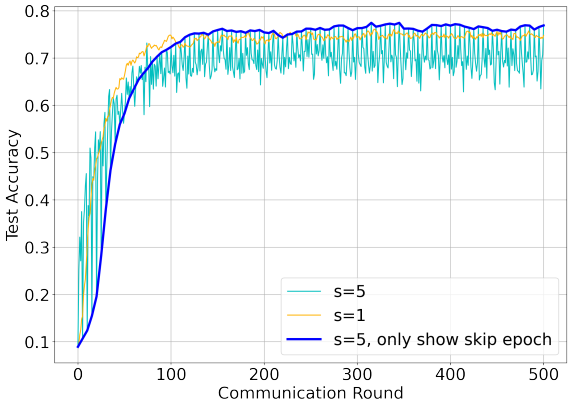

Figure 1. Test accuracy curve in local training of SVHN dataset with $s = 1$ and $s = 5$. For $s = 5$, we also show a curve on epochs with skip weight connection.

D. Proof of Theorem 3.1

Theorem.

$$\tau + \widetilde{p}_t(c) - \sqrt{\frac{1}{C}} \leq \mathcal{T}_t(c) \leq \tau + \widetilde{p}_t(c), \tag{4}$$

Proof. Note that

$$\overline{p}_t = \frac{1}{C}\sum_{c=1}^{C} \widetilde{p}_t(c) = \frac{1}{C}, \tag{5}$$

so we have

$$\begin{aligned} 0 \leq std(\widetilde{p}_t) &= \sqrt{\frac{1}{C-1}(\sum_{c=1}^{C} \widetilde{p}_t(c)^2 - 2\sum_{c=1}^{C} \widetilde{p}_t(c)\overline{p}_t + C\overline{p}_t^2)} \\ &= \sqrt{\frac{1}{C-1}(\sum_{c=1}^{C} \widetilde{p}_t(c)^2 - \frac{1}{C})} \\ &\leq \sqrt{\frac{1}{C-1}(1 - \frac{1}{C})} = \sqrt{\frac{1}{C}}. \end{aligned} \tag{6}$$

Thus,

$$\tau + \widetilde{p}_t(c) - \sqrt{\frac{1}{C}} \leq \mathcal{T}_t(c) \leq \tau + \widetilde{p}_t(c). \tag{7}$$

□

E. More Discussions of Class Balanced Adaptive Pseudo Labeling

In this section, we give more discussions of class balanced adaptive pseudo labeling. We argue that $std(\widetilde{p}_t)$ (Eq. 12 in the main paper) is important for balancing the empirical distribution of training data. Considering some extreme cases that if the amount of training data of some classes (e.g., c_1) are large and some classes (e.g., c_2) are low, this will lead to a large $std(\widetilde{p}_t)$. Thus, the threshold of c_2 will be low and more pseudo labels in c_2 can be selected. For c_1, because $\widetilde{p}_t(c_1)$ is large, $\widetilde{p}_t(c_1) + \tau$ will be much larger than τ_h, so relatively larger $std(\widetilde{p}_t)$ will not change the value of the threshold of c_1 ($\mathcal{T}_t(1)$ will still be τ_h). Finally, the training distribution will be more balanced in the next communication round. In conclusion, introducing $std(\widetilde{p}_t)$ when computing the threshold will encourage a more balanced training process.

F. Experiment Details and More Experimental Results

F.1. Dataset Splitting and Pre-processing

To evaluate the effectiveness of our proposed method, we conduct extensive experiments on four image classification datasets, i.e., SVHN, CIFAR-10, CIFAR-100, Fashion MNIST and one medical image classification dataset: ISIC 2018 (Skin Lesion Analysis Towards Melanoma Detection). For the former four natural datasets, we use the original training and test dataset for training and testing. For ISIC 2018 dataset, we randomly select 80% images for training and the remaining images for testing. For SVNH and CIFAR-10/100, we resize the original 32×32 images to 40×40 pixels and randomly crop a 32×32 region. For Fashion MNIST dataset, we resize the original 28×28 images of these datasets to 36×36 pixels and randomly crop a 32×32 region. Regarding ISIC 2018, we resize the spatial resolution of the original image from 600×450 to 240×240 and randomly crop a 224×224 region. After resizing and randomly croping, we normalize the cropped region for all 5 datasets as the network input. Noted that we strictly follow the settings used in [5] for CIFAR datasets, SVHN and ISIC 2018.

F.2. Detailed Implementation Details

In this section, we give detailed implementation details in our experiments.

Main Experiments in Sec. 4.1 We utilize the SGD optimizer, and implement our method with PyTorch. We adopt ResNet18 [4] from PyTorch for all datasets. For fair comparison, we use the same network architecture and training protocol, including the optimizer, data preprocessing, etc. across all FSSL methods. The learning rate in the labeled client and the unlabeled clients are empirically set to 0.03 and 0.02 for all datasets. The batch size is set to 64 for SVHN, CIFAR-10/100 and Fashion MNIST, and 12 for for ISIC 2018. We empirically set $\tau = 0.84$ for all datasets, $\tau_h = 0.95$ for CIFAR-10/100 and Fashion MNIST and $\tau_h = 0.9$ for SVHN and ISIC 2018. The hyper-parameter β is set as 0.7 for CIFAR-10 and Fashion MNIST, 0.3 for SVHN and CIFAR-100, and 0.03 for ISIC 2018. For residual weight connection, we set $\alpha_1 = 0.8$ in local labeled training and $\alpha_2 = 0.5$ in global model aggregation of all datasets. For ISIC 2018 dataset, we follow [5] to enlarge the the weight of labeled client to about 50%. The total communication round is set to 1000 (warm up stage is 500) for all 5 datasets except CIFAR-100. Since the task of CIFAR-100 is harder, we set the total communication round to 2000 (warm up stage is 1000).

ViT Backbone. The experiments are conducted on SVHN dataset. We utilize the SGD optimizer, and implement our method with PyTorch. We adopt ViT-Tiny pre-trained model on Imagenet [2] as backbone [3] from PyTorch. The learning rate in the labeled client and the unlabeled clients are empirically set to 0.005 and 0.001. The batch size is set to 64. We empirically set $\tau = 0.84$, $\tau_h = 0.95$, $\beta = 0.3$, $\alpha_1 = 0.8$ and $\alpha_2 = 0.5$. Since ViT [3] converges much faster than CNNs [8], the total communication round is set to 200 (warm up stage is 100).

F.3. Discussions of Training Strategies in Warm up Stage when Number of Labeled Clients is Two

When the number of labeled clients is larger than 1, different training strategies can be used for labeled clients w.r.t. number of local epochs and the usage of residual weight connection. [8] claims that final global model with one local training epoch performs better compared with more than one local training epochs. But, we find that more local training epochs with residual weight connection will perform better when the number of labeled clients are more than one. We first compare three representative training strategies: (**1**) model w/ one local training epoch w/o residual weight connection (**2**) model w/ one local training epoch w/ residual weight connection in global aggregation stage (**3**) model w/ eleven local training epoch w/ residual weight connection in both local training and global aggregation stage. We set $\alpha = 0.8$ for both local training and global aggregation and set the skip epoch as $s = 5$. Fig. 2 and 3 show the training curve of these three training strategies. We can see that strategy **3** performs much better and more stable than strategy **1** and **2**. Meanwhile, strategy **1** and **2** perform closely. So residual weight connection in local training has dominant effect for more robust training process compared with in global aggregation. Since more local training epochs w/ residual weight connection performs better, we then compare strategy **1**, **2** and a new training strategy **4**: model w/ eleven local training epochs w/o residual weight connection. Fig. 4 shows the training curves. As shown in Fig. 4, although strategy **4** can converge much faster, the final performances of strategy **4** and strategy **1** are close, which is much lower than strategy **3**. Lastly, we compare the effect of residual weight connection in local training and global aggregation. We set two new strategies, i.e., strategy **5**: model w/ eleven local training epochs w/ residual weight connection only in local training and strategy **6**: model w/ eleven local training epochs w/ residual weight connection only in the global aggregation stage. Fig. 5 shows the training curves. As shown in Fig. 5, strategy **3** and **5** have close performances, which is higher than strategy **6**. So residual weight connection in local training dominates the improvement of final performance. Moreover, the test curves of strategy **3** and **5** are much more stable than strategy **6**, so residual weight connection in local training can also make the global model more stable during training.

F.4. More Experimental Results.

In this section, we give more experimental results. Table 1 reports Area under the ROC Curve (AUC) results on five datasets.

F.5. Results on Partially Labeled Clients

To better study the ability of our CBAPL and residual weight connection in another FSSL setting, following [5], we further conduct experiments that all local clients are partially labeled with 10% data. Since FedIRM [6] requires extra supervision from fully labeled clients which cannot be generalized on this setting, we only compare our method with RSCFed [5] and Fed-Consist [11]. Besides, we report the results of FedAVG [7] trained on all data as the upper bound (local training epoch 1), FedAVG [7] trained on only labeled data as the lower bound (local training epoch 1) and FedAVG [7] trained on only labeled data with residual weight connection in both local training and global aggregation (local training epoch 6). As shown in Table 2, our method surpasses all compared baselines. Note that FedAVG [7] trained on only labeled data w/ residual weight connection can significantly surpass the one w/o residual weight connection. In this setting, although labeled training data

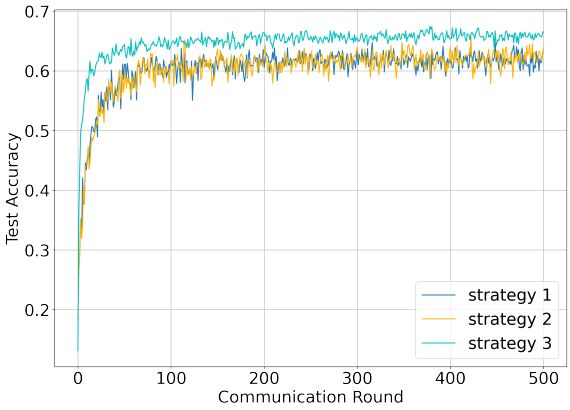

Figure 2. Test accuracy curve of models trained on two labeled clients with different training strategies.

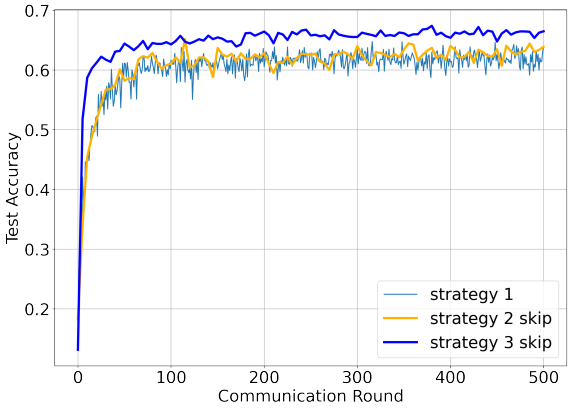

Figure 3. Test accuracy curve of models trained on two labeled clients with different training strategies. For training curve of strategies with residual weight connection, we only show epochs on skip weight connection.

Table 1. Area under the ROC Curve (AUC) results on SVHN, CIFAR-10/100, Fashion MNIST and ISIC 2018 datasets under heterogeneous data partition with ResNet18. FedAVG$^+$ means FedAvg [7] trained with all one labeled clients using our residual weight connection. Fed-consist$^+$ means Fed-Consist [11] using our proposed fixed pseudo labeling without enlarging the weight of labeled client.

Labeling Strategy	Method	Client Num.		Dataset				
		labeled	unlabeled	SVHN	CIFAR10	CIFAR100	Fashion-MNIST	ISIC 2018
Fully supervised	FedAvg [7](upper-bound)	10	0	99.39	98.16	96.70	99.41	93.69
	FedAvg [7](lower-bound)	1	0	94.21	89.90	81.18	95.48	83.62
	FedAvg$^+$ [7]	1	0	97.01	91.61	84.88	97.32	85.45
Semi supervised	FedIRM [6]	1	9	94.79	89.38	83.17	95.55	82.39
	Fed-Consist [11]	1	9	95.92	88.97	82.79	94.36	82.13
	Fed-Consist$^+$ [11]	1	9	97.14	89.75	81.27	95.62	83.37
	RSCFed [5]	1	9	96.32	90.76	85.02	95.96	83.99
	CBAFed(ours)	1	9	98.85	93.87	85.23	98.48	85.01

from all clients are balanced, the total training amount is small. It shows our residual weight connection enjoys the benefits of robustness and the ability of reaching better optimum even when training on balanced federated setting whose amount of training data is small. Fig. 6 shows the test curves w/ and w/o residual weight connection. Since the overall training data is balanced, the test curve w/o residual weight is slightly more stable compared with imbalanced case. But, model w/ residual

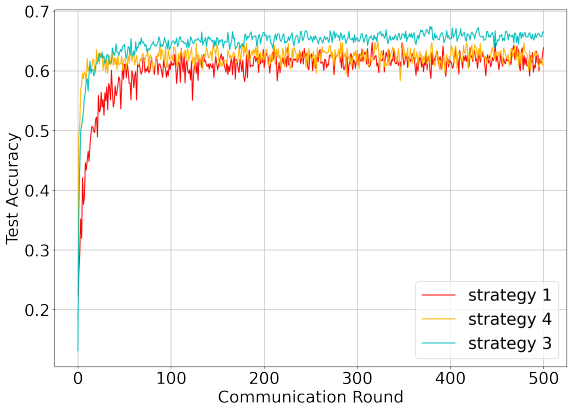

Figure 4. Test accuracy curve of models trained on two labeled clients with different training strategies.

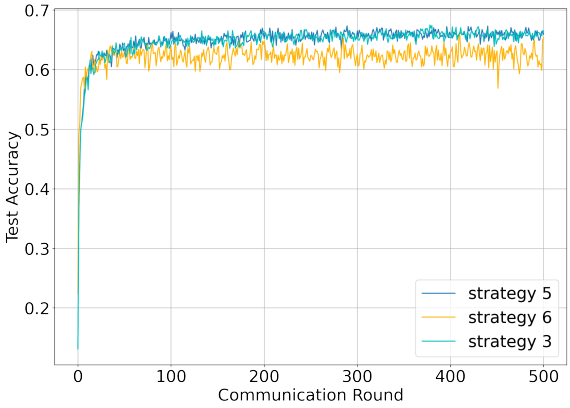

Figure 5. Test accuracy curve of models trained on two labeled clients with different training strategies.

Table 2. Comparison of our method against RSCFed [5], Fed-Consist [11] and FedAVG [7] in cifar-10 dataset, with the number of labeled data in every client is set to 10%.

Method	Client Num.		Accuracy
	labeled	unlabeled	
FedAVG [7](upper bound)	100%	0	80.89
FedAVG [7](lower bound)	10%	0	52.83
FedAVG$^+$ [7]	10%	0	60.55
Fed-Consist [11]	10%	90%	59.75
RSCFed [5]	10%	90%	63.14
CBAFed(ours)	10%	90%	64.82

weight connection performs much better than w/o residual weight connection.

References

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019. 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at

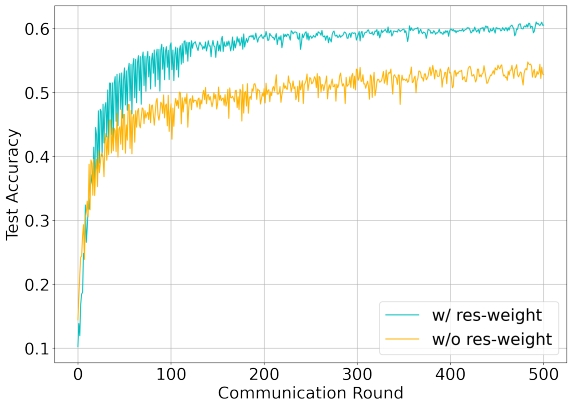

Figure 6. Test accuracy curve of model trained w/ and w/o residual weight connection under partially labeling setting.

scale. *arXiv preprint arXiv:2010.11929*, 2020. 5

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[5] Xiaoxiao Liang, Yiqun Lin, Huazhu Fu, Lei Zhu, and Xiaomeng Li. Rscfed: Random sampling consensus federated semi-supervised learning. In *Proc. CVPR*, 2022. 4, 5, 6, 7

[6] Quande Liu, Hongzheng Yang, Qi Dou, and Pheng-Ann Heng. Federated semi-supervised medical image classification via inter-client relation matching. In *Proc. MICCAI*, 2021. 5, 6

[7] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 3, 5, 6, 7

[8] Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Ehsan Adeli, Li Fei-Fei, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10061–10071, 2022. 5

[9] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020. 1

[10] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017. 1, 3

[11] Dong Yang, Ziyue Xu, Wenqi Li, Andriy Myronenko, Holger R Roth, Stephanie Harmon, Sheng Xu, Baris Turkbey, Evrim Turkbey, Xiaosong Wang, et al. Federated semi-supervised learning for covid region segmentation in chest ct using multi-national data from china, italy, japan. *Medical image analysis*, 70:101992, 2021. 5, 6, 7

[12] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34:18408–18419, 2021. 1