Appendix of No Fear of Classifier Biases: Neural Collapse Inspired Federated Learning with Synthetic and Fixed Classifier

Zexi Li[1] Xinyi Shang[2†] Rui He[1] Tao Lin[3*] Chao Wu[1*]
[1]Zhejiang University [2]Xiamen University [3]Westlake University
{zexi.li,ruihe,chao.wu}@zju.edu.cn shangxinyi@stu.xmu.edu.cn lintao@westlake.edu.cn

1. Pseudo Codes of the Proposed Method

To better present our approach and demonstrate the workflow, we give the pseudo codes of FEDETF. The pseudo code of the federated learning (FL) training is shown in Algorithm 1, and the pseudo code of the personalized local finetuning is shown in Algorithm 2.

Algorithm 1: FEDETF FL Training

Input: Clients $\{1, \dots, K\}$, communication round T, local epoch E, initial model $\mathbf{w}^1 = \{\mathbf{u}, \mathbf{p}, \beta\}$, feature dimension d, balanced loss hyperparameter γ.
Output: Gloabl model $\mathbf{w}^g$.
Synthesize a simplex ETF $\mathbf{V}_{ETF} \in \mathbb{R}^{d\times C}$ by Eq. (3) as the fixed classifier for all clients;
for $t = 1, \dots, T$ **do**
  for client $k = 1, \dots, K$ ***in parallel*** **do**
    $\mathbf{w}_k^t \leftarrow \mathbf{w}^t$;
    for local epoch $e = 1, \dots, E$ **do**
      Obtain $\mathcal{L}_k^g$ by Eq. (6, 7, 8);
      $\mathbf{w}_k^t \leftarrow \mathbf{w}_k^t - \eta\nabla\mathcal{L}_k^g(\mathbf{w}_k^t)$;
    end
  end
  The server updates $\mathbf{w}^{t+1}$ by Eq. (2).
end
The final global model $\mathbf{w}^g = \mathbf{w}^T$.

2. Implementation Details

Models and Data. Our model implementations of the ResNet series and the DenseNet are referred from the codes of [5]. The model implementation of the EfficientNet is referred from the official code in [9], and the implementation of the MobileNetv2 is referred from [8, 4]. For the data, we use the Dirichlet-sampling-based data partition adopted in [6, 1, 2, 7]. It considers a class-imbalanced data heterogeneity, controlled by hyperparameter α, and smaller α refers to more Non-IID data of clients. When $\alpha < 1$, the data are considered to be rather Non-IID, which means that

Algorithm 2: FEDETF Personalized Finetuning

Input: Clients $\{1, \dots, K\}$, iteration round T_p, epoch for each stage E, final global model $\mathbf{w}^g = \{\mathbf{u}, \mathbf{p}, \beta, \mathbf{V}_{ETF}\}$.
Output: Personalized local models $\{\mathbf{w}_k^p\}_{k=1}^K$.
Assign the final global model $\mathbf{w}^g$ as clients' initial local models.
Finetune the feature extractor.
for client $k = 1, \dots, K$ ***in parallel*** **do**
  $\hat{\mathbf{w}}_k = \{\mathbf{u}, \beta\}$, $\overline{\mathbf{w}}_k = \{\mathbf{p}, \mathbf{V}_{ETF}\}$;
  for local epoch $e = 1, \dots, E$ **do**
    Obtain $\mathcal{L}_k^p$ by Eq. (9, 10, 11);
    $\hat{\mathbf{w}}_k \leftarrow \hat{\mathbf{w}}_k - \eta\nabla\mathcal{L}_k^p(\hat{\mathbf{w}}_k)$;
  end
end
for $t = 1, \dots, T_p$ **do**
  for client $k = 1, \dots, K$ ***in parallel*** **do**
    Finetune the ETF classifier.
    $\hat{\mathbf{w}}_k = \{\mathbf{V}_{ETF}, \beta\}$, $\overline{\mathbf{w}}_k = \{\mathbf{p}, \mathbf{u}\}$;
    for local epoch $e = 1, \dots, E$ **do**
      Obtain $\mathcal{L}_k^p$ by Eq. (9, 10, 11);
      $\hat{\mathbf{w}}_k \leftarrow \hat{\mathbf{w}}_k - \eta\nabla\mathcal{L}_k^p(\hat{\mathbf{w}}_k)$;
    end
    Finetune the projection layer.
    $\hat{\mathbf{w}}_k = \{\mathbf{p}, \beta\}$, $\overline{\mathbf{w}}_k = \{\mathbf{V}_{ETF}, \mathbf{u}\}$;
    for local epoch $e = 1, \dots, E$ **do**
      Obtain $\mathcal{L}_k^p$ by Eq. (9, 10, 11);
      $\hat{\mathbf{w}}_k \leftarrow \hat{\mathbf{w}}_k - \eta\nabla\mathcal{L}_k^p(\hat{\mathbf{w}}_k)$;
    end
  end
end
The personalized local models are $\{\mathbf{w}_k^p = \hat{\mathbf{w}}_k \cup \overline{\mathbf{w}}_k\}_{k=1}^K$.

*Corresponding authors. †Work was done during Xinyi's visit to Westlake University.

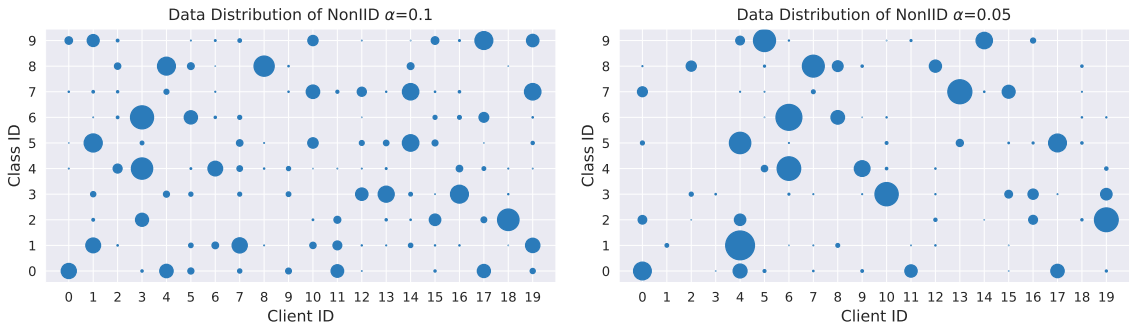

Figure 1. **Visualization of clients' data distributions.** Random seed is 8. Left: data distributions of Non-IID $\alpha = 0.1$ with 20 clients. Right: data distributions of Non-IID $\alpha = 0.05$ with 20 clients.

most of the training samples of one class are likely assigned to a small portion of clients [1]. In our Dirichlet implementation, when α goes smaller, the number of samples in each client along with the class distribution of each client both become more heterogeneous, which is realistic in practical scenarios. We use the same Tiny-ImageNet dataset as in [2].

Local learning rate and optimizer. For CIFAR-10 the local learning rate (LR) $\eta = 0.04$, and for CIFAR-10 and Tiny-ImageNet, $\eta = 0.01$. For clients, we use SGD optimizer with momentum 0.9 and weight decay 5×10^{-4}. Following [3], we adopt a learning rate decaying scheduler, which decays the local LR by 0.99 in each round.

Hyperparameters. For FEDETF, we set the feature dimension to the number of classes, i.e. $d = C$; the initial temperature $\beta = 1$; $\gamma = 1$. We set $\mu_{FedProx} = 0.001$ in FEDPROX and $\alpha_{FedDyn} = 0.01$ in FEDDYN as suggested in their official implementations or papers. For DITTO, the learning setting of the personalized model is the same as the one of the global model. For FEDREP, the epoch number of the classifier training and the epoch number of the feature extractor training are the same and are set as E. For FEDROD, we set $\gamma = 1$. For CCVR, the number of virtual features is 10 per class, and the number of classifier calibration training epochs is 100. For FEDNH, the smoothing hyperparameter $\rho = 0.9$ as suggested in the paper [2].

Randomness. We set the same random seeds for all methods in the same setting. The random seed list is $\{7, 8, 9, 10\}$. For the extremely Non-IID settings when $\alpha = 0.05$, we use the random seeds that can ensure all clients can be assigned a proportion of training data (on the contrary, some random seeds will generate a data partition where particular clients have zero data samples).

Environments. All experiments are conducted in PyTorch with Quadro RTX 8000 GPUs.

3. Visualization

3.1. Visualization of clients' data distributions.

Here, we additionally visualize the clients' data distributions mainly adopted in the main paper. In the main paper, we adopt $\alpha \in \{0.1, 0.05\}$ with 20 clients in Tables 1, 3, and 4. We visualize the data distributions in Figure 1. It shows that in both settings, the clients have extremely heterogeneous data distributions. Especially when $\alpha = 0.05$, all clients have some classes missing, and some clients have extremely rare data (e.g. client 1). We note that these settings are very realistic in practical FL scenarios.

References

[1] Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification. In *International Conference on Learning Representations*, 2021.

[2] Yutong Dai, Zeyuan Chen, Junnan Li, Shelby Heinecke, Lichao Sun, and Ran Xu. Tackling data heterogeneity in federated learning with class prototypes. 2023.

[3] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[4] Andrew Howard, Andrey Zhmoginov, Liang-Chieh Chen, Mark Sandler, and Menglong Zhu. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. 2018.

[5] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

[6] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[7] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *Advances in Neural Information Processing Systems*, 34:5972–5984, 2021.

[8] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[9] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.