

CMoS: Customizing Model Structures for Personalized Federated Learning

1st Bingyan Liu

Computer Science

Beijing University of Posts and Telecommunications

Beijing, China

bingyanliu@bupt.edu.cn

2nd Jing Yang

Department of Computer System and Technology

Universiti Malaya

Kuala Lumpur, Malaysia

s2147529@siswa.um.edu.my

Abstract—Personalized Federated Learning (PFL) has recently received significant interest due to its capability of generating customized models for data-heterogeneous clients. Previous work focuses on weight adaptation to fit the data distribution while ignoring the correlation between various data distributions and their corresponding model structures. In this paper, we introduce a new paradigm called data-aware structure personalization, aiming to personalize the model structure according to the data distribution of each client during FL. Specifically, we develop CMoS, a framework to Customize Model Structures for heterogeneous client data in a learning-based manner. The key idea is to gradually learn a data-aware sub-structure from the original model at each federated round, which is achieved by introducing a newly designed binary mask and a sparsity loss during local training. Extensive experiments on multiple scenarios demonstrate the effectiveness of CMoS in generating personalized models with superior performance.

Index Terms—Federated Learning, Heterogeneous Data, Personalization, Customization, Sparsity

I. INTRODUCTION

Personalized FL (PFL) [1]–[5] is a prevalent research direction in vision-based FL, which is motivated by the data heterogeneity situation in real-world scenarios. Specifically, data are typically non-independently and identically distributed (non-iid) in an FL system. For example, some users may prefer dogs and collect more dog images than others. The same object located in different environments may also lead to heterogeneous distribution because of various environment backgrounds. Therefore, it is inappropriate to directly deploy a shared global model to all users and researchers have proposed many algorithms to further optimize the model weights to fit different data distributions for a more personalized model [6].

Despite being proved effective, we observe that current PFL methods pay more attention to adapting model weights while keeping the model structure unchanged. Our intuition is that, besides the weights, the model structure also plays an important role since the distinctive data property in different clients may also require different structures to obtain good performance. Traditional model pruning techniques [7] seem natural to achieve this structure personalization by removing unrelated weights and several papers have tried to apply pruning techniques to FL [8]–[10]. However, all of them are **resource-aware**, which requires us to manually pre-set a pruning ratio before training according to the concrete device

constraints. In PFL, it is infeasible to apply such methods because the key objective of PFL is to generate a customized model to fit the data distribution of each client, regardless of the model size. Therefore, instead of meeting the resource constraints, structure personalization in PFL should be **data-aware** and have no limitation on the final model.

Unfortunately, to the best of our knowledge, there is no work that can automatically customize model structure in terms of the data distribution of clients. To fill the gap, we propose a new paradigm, called **Data-aware Structure Personalization (DSP)**, which aims at generating a data-aware model structure for improved PFL performance. Compared to *resource-aware structure pruning*, DSP imposes no limitation on the model size and focuses more on the specific data distribution, resulting various personalized models for different client. Following the paradigm, we propose **CMoS**, a framework that achieves DSP by **Customizing Model Structure** in a learning-based manner. The key idea is to automatically learn a suitable model sparsity degree in terms of the data in each client during the local training process, with the help of the introduced binary masks to each weight and a sparsity-based regularization term. In this way, we are able to generate a series of sparse structures for each client and the personalized structures can be obtained by removing the weights of zero value in the sparse models.

Specifically, CMoS introduces *sparsity-based structure adaptation*, where the specially designed binary masks and a sparsity regularization term are introduced into local training. Instead of adding a binary vector, the mask in our framework is decomposed into a threshold vector and a unit step function to fit our paradigm. CMoS achieves federation by conducting *index-based sub-structure aggregation*. Because each structure comes from a shared model, CMoS records the index of each weight location and only aggregates weights with the same index. Consequently, the knowledge in different locations can be effectively fused.

We have conducted a rich set of experiments to evaluate the performance of CMoS. To begin with, we evaluate CMoS on four different data distribution scenarios. Experiments on these scenarios demonstrate that CMoS can generate a personalized model structure for a specific data distribution while achieving comparable or even better performance with less overall communication cost. In addition, we also conduct

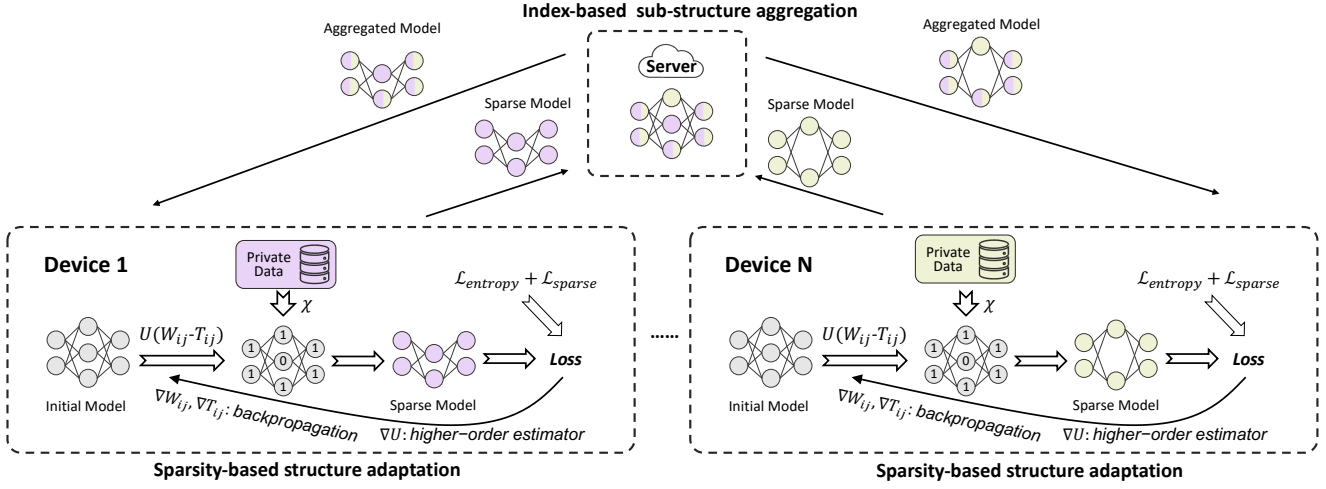


Fig. 1. The pipeline of our framework. Here different colors denote different features.

experiments on a non-iid benchmark [11] and discover that CMoS can be seamlessly combined with different aggregation techniques to further obtain an improved personalized model.

This paper makes the following contributions: (1) We propose a new paradigm, called Data-aware Structure Personalization (DSP), where each client owns a distinctive structure in terms of its data distribution. (2) We design and implement CMoS, a framework to achieve automated and effective DSP. Through *sparsity-based structure adaptation* and *index-based sub-structure aggregation*, CMoS can achieve improved performance with a customized model structure. (3) Extensive experiments on multiple scenarios demonstrate the superiority of CMoS over other baselines. Besides, CMoS can be effectively combined with existing aggregation schemes, confirming the generality of our framework.

II. RELATED WORK

Federated Learning. Federated learning (FL) has been proposed as a privacy-preserving distributed machine learning approach that enables training on a large corpus of decentralized client data [12], [13]. In our context, we focus on Personalized FL (PFL), which aims at addressing the data heterogeneity problem on FL [3]. However, these methods fail to change the final model structure in FL, which is sub-optimal as different distributions of client data may call for not only the personalized weights but also the personalized structure. We would like to highlight that CMoS pays more attention to the client-side model structure, which is orthogonal to existing personalization works.

Model Compression. Model compression targets at reducing the model size without incurring much performance degradation. Typically model compression includes four types: model pruning, model quantization, knowledge distillation and compact model design. Among them, model pruning has become a prevalent technique due to its flexibility and simplicity [7]. Recent works have explored leveraging model pruning techniques to change the model structure [8], [14], [15] for

efficient FL. However, they either target the computational and communication limitations or generating a global model, which is different from our CMoS that focuses on the data-aware personalized model structure.

III. CMoS DESIGN

A. Framework Overview

Figure 1 provides the pipeline of our *sparsity-based structure adaptation* and *index-based sub-structure aggregation*. In the remainder of the section, we describe in detail our approach for implementing each technique.

B. Sparsity-based Structure Adaptation

Unlike existing pruning work pursuing a high sparsity ratio, we pay more attention to the matching degree of the data distribution no matter how sparse the model is. Next, we describe the main steps to implement our technique.

Applying binary masks to the original model. Let $W = \{W_1, W_2, \dots, W_l\}$ be the parameter weights of the original model and $BM = \{BM_1, BM_2, \dots, BM_l\}$ is the corresponding binary masks attached in each layer. Here l represents the number of layers in the model. For the j_{th} neuron in the i_{th} layer with the weight W_{ij} , the output after the mask can be represented as

$$Output = BM_{ij} \odot (W_{ij}) = U(W_{ij} - T_{ij}) \quad (1)$$

where T_{ij} denotes the trainable threshold for normalization and U is a unit step function to enforce the final mask value to be 0 or 1.

Although the feed-forward process is simple, it is difficult to achieve back-propagation with the binary masks since the step function $U()$ is non-differentiable. In our work, we adopt the long-tailed higher-order estimator [16] to estimate the gradient, which has a wide active range between $[-1, 1]$ with a non-zero

TABLE I

RESULTS ON THE CATEGORY-HETEROGENEITY SCENARIO. HERE CLIENTS WITH THE SAME DISTRIBUTION TYPE ARE CONSIDERED TOGETHER AND WE AVERAGE THEIR ACCURACY.

Method	Category-heterogeneity					
	Type1	Type2	Type3	Type4	Type5	Avg
FedAvg	44.35	50.15	33.25	42.45	48.70	43.78
FedDST	86.70	84.50	72.23	87.36	82.97	82.75
PrunedFL	81.23	79.52	62.80	80.84	79.10	76.70
CMoS	88.14	93.69	80.36	89.56	82.79	86.91

gradient to avoid gradient vanishing during training. Formally, it can be denoted as

$$\frac{d}{dx}U(x) \approx \begin{cases} 2 - 4|x|, & -0.4 \leq x \leq 0.4 \\ 0.4, & 0.4 < |x| \leq 1 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

In this way, the mask can be trained via back-propagation. Besides, we would like to highlight that the model parameter W can receive two types of gradient: the performance gradient for better model performance and the structure gradient for customized sparse structure.

Introducing the sparse regularization loss. To further encourage the sparse structure adaptation, we design and add a regularization term called \mathcal{L}_{sparse} to the original training loss. Specifically, the term can be defined as

$$L_{sparse} = \sum_{i=1}^l \sum_{j=1}^{f_i} \exp(-T_{ij}) \quad (3)$$

where f_i denotes the number of neuron in the i_{th} layer. With this term, the threshold with a low value will be penalized, and thus the sparsity degree will increase, which means that we can utilize it to balance the weight and structure training.

Dynamic sparse structure training. After introducing the masks and the regularization term, we define our final optimization objective as follows

$$\arg \min_{W, T} \frac{1}{N} \left(\sum_{i=1}^N \mathcal{L}_{entropy}((\mathbf{x}_i, \mathbf{y}_i); W, T) \right) + \alpha \mathcal{L}_{sparse} \quad (4)$$

where $\mathcal{L}_{entropy}$ is the traditional classification loss and $(\mathbf{x}_i, \mathbf{y}_i)$ denotes the i_{th} data and label in a client. α is the hyper-parameter that controls the degree of the structure adaptation. As a result, we can finally obtain the optimal weights W and structure M_{sparse} with the trained T in each local training phase.

C. Index-based Sub-structure Aggregation

The index-based sub-structure aggregation aims to aggregate the knowledge of different sparse models. Because the structure of sparse models is originally from a shared model, we can record the location of each neuron's weight and aggregate the weights of different models in the same location.

Taking the generated sparse models after K rounds as an example. The models $M_{sparse}^K = \{M_{sparse1}^K, M_{sparse2}^K, \dots,$

TABLE II

RESULTS ON THE ENVIRONMENT-HETEROGENEITY SCENARIO.

Method	Environment-heterogeneity				
	Artistic	Clipart	Product	Real	Avg
FedAvg	62.47	61.05	83.88	78.30	71.43
FedDST	62.97	65.77	85.57	76.07	72.60
PrunedFL	60.62	66.36	84.19	78.16	72.33
CMoS	64.42	68.00	86.15	78.42	74.25

TABLE III

RESULTS ON THE SCALE-HETEROGENEITY SCENARIO.

Method	Scale-heterogeneity			
	Large	Middle	Small	Avg
FedAvg	70.31	70.31	70.31	70.31
FedDST	70.29	66.83	67.55	68.22
PrunedFL	70.29	70.29	70.29	70.29
CMoS	70.55	70.60	71.03	70.73

$M_{sparseN}^K\}$ and their corresponding location index $I_{sparse}^K = \{I_{sparse1}^K, I_{sparse2}^K, \dots, I_{sparseN}^K\}$ are uploaded to the server. For each index pair (i, j) , if $(i, j) \in I_{sparse}^K$, we extract the weights on the location W_{ij} . In this way, we are able to extract Z weights and then average them as follows

$$W_{ij}^* = \frac{1}{Z} \sum_{m \in Q_{ij}} W_{ij}^m \quad (5)$$

where W_{ij}^* is the final aggregated weight on this location, and Q_{ij} represents the index list of the extracted weights. With this equation, we can aggregate the sparse models and distribute the aggregated weights to each sparse model in terms of its location.

IV. EXPERIMENTS

A. Experimental Setup

Scenarios. We evaluate CMoS on three typical PFL scenarios: The *category-heterogeneity* (Cat) scenario suggests that different clients may collect different categories due to user preference. We simulate this situation by CIFAR-10 [17]. Specifically, we simulate 100 clients and every 20 clients belong to a type of distribution. Here we use TypeX (X=1,2,...,5) to denote each type of distribution. The *environment-heterogeneity* (Env) scenario indicates that the main recognition object is identical while the background is different due to diverse environments. We use Office-Home [18] for simulation, which contains four distinctive domains: Artistic images (Ar), Clipart images (Cl), Product images (Pr) and Real-World images (Rw). As a recent work did [19], we partition each domain into a training set (80%) and a testing set (20%). In total, we have 20 clients with 4 types of data distribution. The *scale-heterogeneity* (Sca) scenario means that the amount of data is different from each client. Based on CIFAR-10, we design three types of scale: *small-scale*, *middle-scale* and *large scale*. For each scale, we respectively create 10

TABLE IV
CUSTOMIZED FINAL MODEL SIZE (%) TO THE ORIGINAL MODEL IN EACH SCENARIO.

Scenario		Customized Size (%)
Category	Type1	37.41
	Type2	37.28
	Type3	37.36
	Type4	37.36
	Type5	37.37
Environment	Artistic	45.31
	Clipart	38.13
	Product	37.58
	RealWorld	37.77
Scale	Small	30.15
	Middle	33.58
	Large	36.74

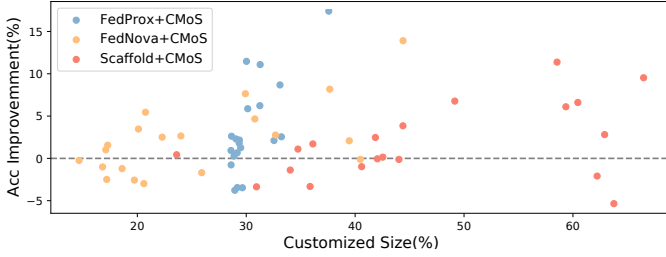


Fig. 2. Performance of applying CMoS to weight-based personalization methods.

clients and each of them holds 10 categories with 500, 1000 and 2000 samples.

Baselines. We compare the following baselines: *FedAvg* [12], *FedDST* [15], and *PrunedFL* [10]. Besides, we combine CMoS with three typical weight aggregation approaches (i.e., *FedProx* [6], *FedNova* [13] and *Scaffold* [20]) to test the performance. Besides, we would like to highlight that for each pruning-based baseline, we control the model size to be same as the averaged size among clients personalized by CMoS, in order to make a fair comparison.

Implementation details. We implement CMoS in Python with PyTorch and all the experiments are conducted on the ResNet-18 architecture [21]. The concrete parameter settings are as follows: The learning rate is set to 0.001 with a momentum of 0.5. For CIFAR-10, the input images are randomly cropped to 32*32 and normalized to zero mean for each channel. For Office-Home, the images are processed to the dimension of 224*224. Besides, the hyper-parameter α is set to 0.005 for the category-heterogeneity scenario and 0.0001 for other scenarios. All the experiments are conducted three times and we average them as the reported results.

B. Overall Results

Accuracy performance on different scenarios. Table I-III summarize the results. From these tables, we can see that on average, CMoS can achieve the best accuracy performance for all of the scenarios with a personalized model. Notably, on the category-heterogeneity scenario, CMoS can outperform other FL methods by 4.16%, which validates that by personalizing

the model structure, the resulting model is more suitable to the users with different data distributions. Besides the accuracy improvement, another main advantage of CMoS is its efficiency since the customized model is a sparse and small model. For instance, in our category-heterogeneity scenario, CMoS outperforms other baselines by 4.16% on average accuracy, with only 67.99% overall communication cost and 37.36% model size.

Performance of applying CMoS to other weight aggregation methods. Besides *FedAvg*, there are a variety of weight aggregation schemes. In this part, we explore whether CMoS can benefit them. Here we conduct experiments on a non-iid benchmark [11], which includes three aggregation methods (i.e., *FedProx* [6], *FedNova* [13] and *Scaffold* [20]). We apply CMoS to these methods on CIFAR-10 and simulate 20 clients for each method to observe the performance. Figure 2 shows the results. For each client, we record the accuracy improvement and the corresponding customized model size (i.e., percentage of the original model). From the figure, we can observe that: (1) The customized model varies significantly among different clients, which means that other weight aggregation methods also require adapting the model structure. (2) With CMoS, all of the aggregation methods can achieve comparable or better (over 70%) performance with a smaller customized model. Therefore, it is desirable to combine them together for improved personalization.

C. Analysis on Generated Structures

In this subsection, we analyze the detailed structure sparsity, which is shown in Table IV. We summarize the key observations as follows. (1) For the *category-heterogeneity scenario*, the final model size/sparsity among different distribution types is roughly identical, which indicates that the different categories will not significantly affect the overall model sparsity. Besides, we find that the learned model for all of the types is extremely small ($\approx 37\%$) compared to traditional FL. (2) For the *environment-heterogeneity scenario*, we can observe that the environmental factor does affect the final size of learned models. For example, clients of the *Artistic* domain require a bigger model than others. We believe this is because the artistic image is more sophisticated and hence needs more parameters to learn. (3) For the *scale-heterogeneity scenario*, it is clear to see that the scale of data plays an important role in the size of the final models. Notably, when the data scale is small, we are able to shrink the model size to roughly 30%, which proves the necessity of structure customization.

V. CONCLUSION

In this paper, we propose CMoS, a novel framework to achieve data-aware structure adaptation for personalized federated learning. Specifically, we design two key techniques: sparsity-based structure adaptation and index-based sub-structure aggregation to achieve our goal. Extensive experiments on our simulated scenarios demonstrate the effectiveness of CMoS, outperforming other methods with less communication cost and a smaller customized model.

REFERENCES

- [1] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan *et al.*, “Towards federated learning at scale: System design,” *arXiv preprint arXiv:1902.01046*, 2019.
- [2] C. T. Dinh, N. H. Tran, and T. D. Nguyen, “Personalized federated learning with moreau envelopes,” *arXiv preprint arXiv:2006.08848*, 2020.
- [3] P. P. Liang, T. Liu, L. Ziyin, N. B. Allen, R. P. Auerbach, D. Brent, R. Salakhutdinov, and L.-P. Morency, “Think locally, act globally: Federated learning with local and global representations,” *Neural Information Processing Systems*, 2019.
- [4] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen, “Billion-scale federated learning on mobile clients: a submodel design with tunable privacy,” in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020, pp. 1–14.
- [5] H. Wang, Z. Kaplan, D. Niu, and B. Li, “Optimizing federated learning on non-iid data with reinforcement learning,” in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1698–1707.
- [6] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.
- [7] S. Han, J. Pool, J. Tran, and W. J. Dally, “Learning both weights and connections for efficient neural networks,” *arXiv preprint arXiv:1506.02626*, 2015.
- [8] A. Li, J. Sun, P. Li, Y. Pu, H. Li, and Y. Chen, “Hermes: an efficient federated learning framework for heterogeneous mobile clients,” in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, 2021, pp. 420–437.
- [9] V. Mugunthan, E. Lin, V. Gokul, C. Lau, L. Kagal, and S. Pieper, “Fedltn: Federated learning for sparse and personalized lottery ticket networks,” in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII*. Springer, 2022, pp. 69–85.
- [10] Y. Jiang, S. Wang, V. Valls, B. J. Ko, W.-H. Lee, K. K. Leung, and L. Tassiulas, “Model pruning enables efficient federated learning on edge devices,” *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
- [11] Q. Li, Y. Diao, Q. Chen, and B. He, “Federated learning on non-iid data silos: An experimental study,” in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 965–978.
- [12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.
- [13] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.
- [14] A. Li, J. Sun, B. Wang, L. Duan, S. Li, Y. Chen, and H. Li, “Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets,” *arXiv preprint arXiv:2008.03371*, 2020.
- [15] S. Bibikar, H. Vikalo, Z. Wang, and X. Chen, “Federated dynamic sparse training: Computing less, communicating less, yet learning better,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 6, 2022, pp. 6080–6088.
- [16] Z. Xu and R. C. Cheung, “Accurate and compact convolutional neural networks with trained binarization,” *arXiv preprint arXiv:1909.11366*, 2019.
- [17] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images,” 2009.
- [18] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, “Deep hashing network for unsupervised domain adaptation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5018–5027.
- [19] B. Liu, Y. Guo, and X. Chen, “Pfa: Privacy-preserving federated adaptation for effective model personalization,” in *Proceedings of the Web Conference 2021*, 2021, pp. 923–934.
- [20] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, “Scaffold: Stochastic controlled averaging for on-device federated learning,” in *Proceedings of the 37th International Conference on Machine Learning*, 2020.
- [21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.