模式分析与机器智能
工业和信息化部重点实验室
MIIT Key Laboratory of
Pattern Analysis & Machine Intelligence

ParNeC 模式识别与神经计算研究组
PAttern Recognition and NEural Computing

Exploiting Label Skews in Federated Learning with Model Concatenation

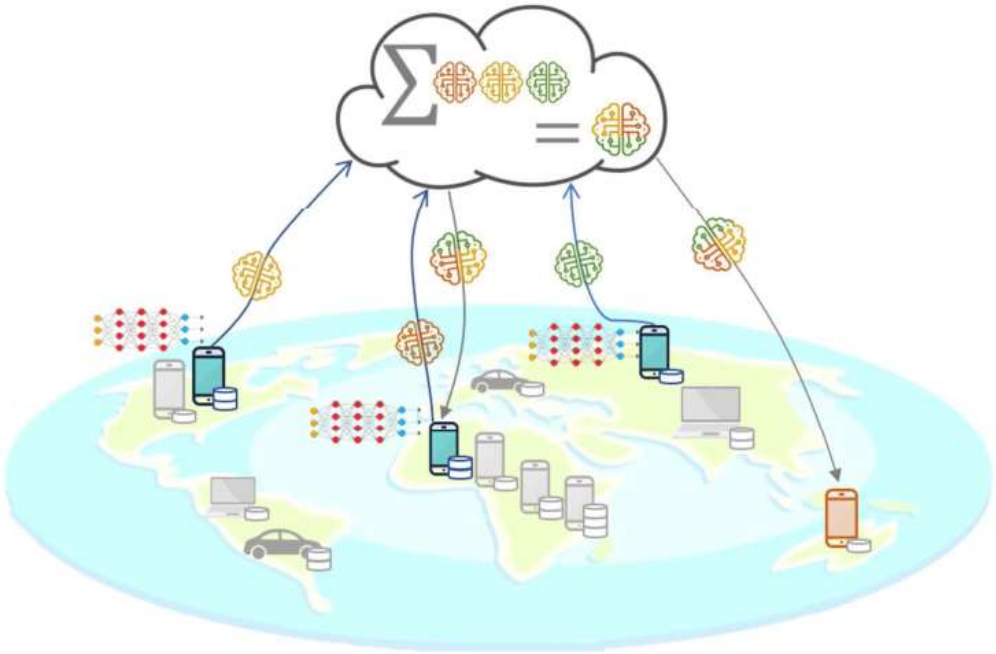
Yiqun Diao[1], Qinbin Li[2], Bingsheng He[1]

[1]National University of Singapore
[2]University of California, Berkeley
{yiqun, hebs}@comp.nus.edu.sg, qinbin@berkeley.edu

AAAI 2024

Federated Learning (FL)

FL can improve model's performance while protecting users' privacy.



Label skews

Different clients have different label distributions. Some may have few or no data of some classes.

	bird	deer	frog	ship
Client 1				
Client 2				

Related work

FedAvg
(McMahan et al.,2016)

under the non-IID data distribution cases, each client trains a good local model towards its local optimum.

FedProx
(Li et al.,2020)

adds a regularization term which measures the distance between the local model and the global model.

SCAFFOLD
(Karimireddy et al.,2020)

adjusts the local gradient by keeping a correction term for each client, therefore its communication cost doubles.

Related work

MOON
(Li, He, and Song 2021)

regularizing by a contrastive loss to measure the distance between representations of the local model and the global model.

FedRS
(Li and Zhan 2021)

proposes to restrict the updates of missing classes by down-scaling their logits, however it only deals with missing classes.

FedLC
(Zhang et al. 2022a)

further calibrates logits to reduce the updates of minority classes based on the label statistics of local training data.

Pitfalls of existing methods in label skews

Under label skews, the local models can be much different as they are trained on different classes. Therefore it hardly makes sense to average each parameter of these models with quite different tasks.

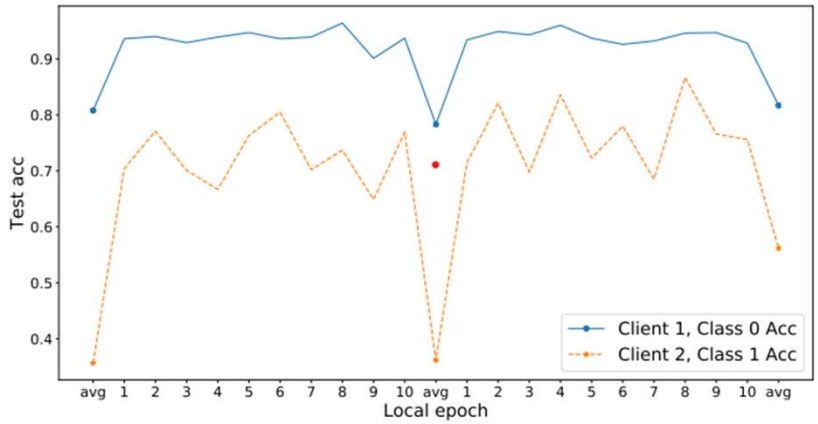


Figure 1: Accuracy of local models and averaged model on two clients under label skews.

Motivation

An alternative view of label skews

Since each client's model is well-fitted in its own dataset, we already have quite a few locally well-trained feature extractors. Intuitively, concatenating the features from different local extractors can provide a better feature representation for label skews. Thus the authors propose the idea of concatenating feature extractors and training a global classifier.

If we concatenate the models of all clients, our final model size can grow much large if there are many clients, and the overhead of training the global classifier is much more expensive. In practice, although label skews are prevalent, some parties may have similar label distributions. By clustering all clients into a few groups via their label distributions, we can control the size of global model. Inside each group, since grouped clients have similar label distributions, the trained model can capture this kind of data well.

FedConcat has three stages: clustering, averaging and post-training.

Stage 1-A: Clustering with label distributions

In order to alleviate the label imbalance problem, the authors perform clustering based on label distributions, so that each cluster hosts clients with similar label distributions. Formally, for client i, suppose there are $N_{i,j}$ samples of class j, and there are a total of $N_i = \sum_j N_{i,j}$ samples. Its label distribution is defined as vector

$$P_i(y) = (\frac{N_{i,1}}{N_i}, \frac{N_{i,2}}{N_i}, ..., \frac{N_{i,m}}{N_i}), \qquad (2)$$

where there are m classes globally. In this paper, they use K-means algorithm (Lloyd 1982) to perform clustering. For the hyper-parameter K, one can utilize elbow method to select the best value.

Stage 1-B: Clustering without label distributions

If clients are unable to upload label distributions due to privacy concerns, they propose to utilize the uploaded local models of the first round to infer the approximate label distribution of each client. In this way, we only upload trained models like FedAvg, which does not cause any extra privacy leakage.

Many works have observed that predictions of deep learning model are biased towards the majority classes of the training set. Intuitively, if we put a large batch of random inputs into the client model, the average prediction can indicate the label distribution of training data.

Formally, denote the model of client i as f_i. They randomly generate r inputs $X_1, \ldots, X_r$, the inferred distribution of client i

$$P_i^{ID}(y) = \frac{1}{r}\sum_{j=1}^{r}\sigma(f_i(X_j)), \qquad (3)$$

They refer to this variant as FedConcat with Inferred Distribution (FedConcat-ID).

Stage 2: Averaging

Within each cluster, we use FedAvg to train a model that fits well for such cluster. Inside a cluster, since the label distributions of the clients are similar, we expect the global model to have a good performance on the dominant classes of the cluster.

Stage 3: Post-training

Now that we have K models, we stack their encoders (all layers but the last layer) as the global feature extractor. Then we broadcast the global feature extractor to all clients for one time, and ask clients to jointly train a classifier using FedAvg, with the global feature extractor fixed.

The Overall Algorithm

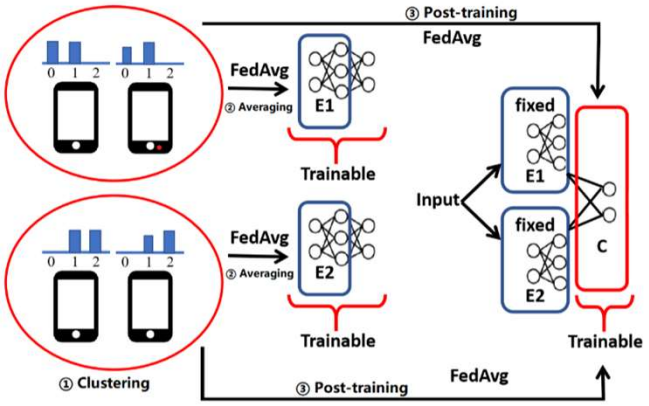


Figure 2: The workflow of FedConcat. (1) Clustering stage: clients are clustered based on label distributions; (2) Averaging stage: each cluster trains a model using FedAvg; (3) Post-training stage: all well-trained feature extractors (E1, E2) are concatenated. All clients train a global classifier (C) collectively with feature extractors fixed. For FedConcat-ID, label distributions are inferred in the clustering stage.

Algorithm 1: FedConcat and FedConcat-ID

Input: number of clients N, number of clusters K, number of training rounds of the encoder T_e, number of training rounds of the classifier T_c

Output: the final model w

if FedConcat **then**
  $S_1, S_2, ..., S_K \leftarrow Kmeans(P_i(y)_{i=1}^{N})$ // Perform K-means based on label distributions
if FedConcat-ID **then**
  Initialize global model f_g
  for $i = 1, 2, ..., N$ in parallel **do**
    $f_i \leftarrow TrainLocal(f_g)$ // Send model to each client for local training
  $S_1, S_2, ..., S_K \leftarrow Kmeans(P_i^{ID}(y)_{i=1}^{N})$ // Infer label distributions by Eq. (3) and perform K-means
Initialize encoder E_i and classifier C_i for each cluster
for $t = 1, 2, ..., T_e$ **do**
  for $i = 1, 2, ..., K$ **do**
    $E_i, C_i \leftarrow FedAvg(\{E_i, C_i\}, S_i)$ // Run FedAvg to train encoder and classifier for each cluster
$E = \{E_1, E_2, ..., E_K\}$
Initialize global classifier C
for $t = 1, 2, ..., T_c$ **do**
  $C \leftarrow FedAvg(C, \bigcup_{i=1}^{K} S_i)$ // Fix E and run FedAvg on all clients to train C
return final model $w = \{E, C\}$

Experiment

Effectiveness

Table 1: Experimental results of our methods compared with baselines with same communication cost. The model of baseline algorithms is the model of one cluster in FedConcat. We repeat experiments with three different random seeds.

Dataset	Partition	FedAvg	FedProx	MOON	FedRS	FedLC	FedConcat	FedConcat-ID	Centralized
CIFAR-10	$\#C=2$	53.6%±0.8%	53.1%±0.6%	53.4%±1.3%	53.8%±0.8%	49.8%±0.6%	**56.9%±0.2%**	**56.5%±2.6%**	70.2%±0.9%
	$\#C=3$	57.6%±0.6%	57.4%±0.8%	58.6%±1.8%	59.1%±1.4%	58.1%±1.4%	**62.0%±0.6%**	**61.8%±0.8%**	
	$p_k \sim Dir(0.1)$	53.0%±1.0%	52.8%±1.0%	53.1%±3.5%	54.8%±0.3%	53.7%±0.8%	**57.7%±0.4%**	**56.9%±1.4%**	
	$p_k \sim Dir(0.5)$	59.9%±0.5%	59.9%±0.6%	61.2%±1.8%	61.5%±0.8%	61.5%±1.1%	**64.2%±0.7%**	**63.7%±0.8%**	
SVHN	$\#C=2$	82.8%±0.5%	82.6%±0.6%	83.0%±0.1%	79.5%±1.2%	75.7%±2.3%	**83.4%±1.4%**	**83.2%±1.9%**	86.2%±0.9%
	$\#C=3$	85.2%±0.4%	85.2%±0.6%	84.7%±0.4%	85.7%±0.4%	84.8%±0.4%	**86.0%±0.9%**	**86.1%±0.5%**	
	$p_k \sim Dir(0.1)$	**84.0%±1.3%**	83.9%±1.2%	83.7%±1.1%	80.9%±0.9%	78.8%±0.8%	83.2%±0.9%	82.9%±0.3%	
	$p_k \sim Dir(0.5)$	87.2%±0.2%	87.2%±0.2%	87.2%±0.2%	87.1%±0.1%	87.1%±0.5%	**87.5%±0.1%**	**87.9%±0.3%**	
FMNIST	$\#C=2$	79.0%±4.7%	81.8%±2.8%	81.4%±1.4%	78.3%±1.8%	77.7%±2.9%	**84.4%±0.6%**	**83.0%±2.0%**	88.4%±0.2%
	$\#C=3$	84.7%±1.4%	85.7%±0.3%	84.6%±1.9%	85.8%±0.8%	86.0%±0.3%	**87.1%±0.2%**	**86.6%±0.1%**	
	$p_k \sim Dir(0.1)$	85.1%±0.5%	**85.2%±0.8%**	85.0%±0.1%	82.5%±0.5%	82.1%±1.6%	84.5%±0.1%	85.0%±0.4%	
	$p_k \sim Dir(0.5)$	87.5%±0.3%	87.4%±0.4%	87.4%±0.5%	87.5%±0.5%	87.5%±0.4%	**87.7%±0.1%**	**87.5%±0.2%**	

Experiment

Scalability

Table 2: Scalability of FedConcat and FedConcat-ID compared with baselines on CIFAR-10 dataset.

#Clients	Partition	FedAvg	FedProx	MOON	FedRS	FedLC	FedConcat	FedConcat-ID
100	$\#C = 2$	44.4%	43.7%	44.4%	**54.4%**	52.6%	51.0%	53.2%
	$\#C = 3$	55.4%	54.6%	55.2%	56.5%	54.8%	**59.0%**	**59.3%**
	$p_k \sim Dir(0.1)$	45.0%	44.8%	45.3%	50.5%	50.0%	49.1%	**50.7%**
	$p_k \sim Dir(0.5)$	58.1%	58.3%	57.7%	59.0%	58.1%	**63.3%**	**62.2%**
200	$\#C = 2$	39.7%	40.9%	40.3%	46.2%	48.6%	**48.9%**	44.6%
	$\#C = 3$	47.7%	48.2%	47.8%	51.5%	51.0%	**53.1%**	**51.8%**
	$p_k \sim Dir(0.1)$	40.5%	41.2%	40.9%	43.6%	43.7%	**47.6%**	**46.7%**
	$p_k \sim Dir(0.5)$	53.9%	53.5%	53.9%	54.4%	53.7%	**56.7%**	**56.8%**

Experiments on Larger Model, More Complicated Tasks

Table 3: Experimental results on CIFAR-100 and Tiny-ImageNet with ResNet-50. We tune the weight decay among $\{0.00001, 0.001, 0.002, 0.005\}$ for all algorithms. We present the average and standard deviation of the last 10 rounds.

Dataset	Partition	FedAvg	FedProx	MOON	FedRS	FedLC	FedConcat	FedConcat-ID	Centralized
CIFAR-100	$\#C = 2$	8.4%±1.2%	8.1%±1.3%	8.0%±1.0%	7.0%±1.5%	3.8%±0.4%	**18.5%±0.7%**	**16.4%±0.8%**	70.5%±0.1%
	$\#C = 3$	21.6%±2.8%	18.7%±6.0%	19.0%±2.4%	19.2%±1.7%	12.6%±0.7%	**34.4%±1.1%**	**32.9%±0.6%**	
	$p_k \sim Dir(0.1)$	52.4%±4.9%	51.5%±8.1%	55.2%±3.3%	51.8%±2.8%	50.5%±2.9%	**61.2%±0.4%**	**62.1%±0.5%**	
	$p_k \sim Dir(0.5)$	62.0%±1.2%	61.2%±1.3%	61.9%±0.9%	61.9%±1.2%	61.4%±1.3%	**66.3%±0.1%**	**65.6%±0.1%**	
Tiny-ImageNet	$\#C = 2$	3.1%±0.4%	2.7%±0.3%	3.0%±0.4%	3.1%±0.1%	2.0%±0.1%	**4.3%±0.1%**	**4.3%±0.2%**	49.9%±0.2%
	$\#C = 3$	4.9%±2.4%	5.1%±1.0%	6.3%±1.7%	3.3%±0.4%	1.7%±0.1%	**11.7%±0.3%**	**9.6%±0.2%**	
	$p_k \sim Dir(0.1)$	40.8%±0.5%	40.8%±0.6%	40.6%±0.8%	39.7%±0.7%	39.9%±0.8%	**43.1%±0.2%**	**42.6%±0.2%**	
	$p_k \sim Dir(0.5)$	44.0%±0.6%	44.2%±0.7%	44.1%±0.5%	43.6%±0.9%	43.9%±0.5%	**44.3%±0.1%**	43.8%±0.2%	

Effect of Clustering

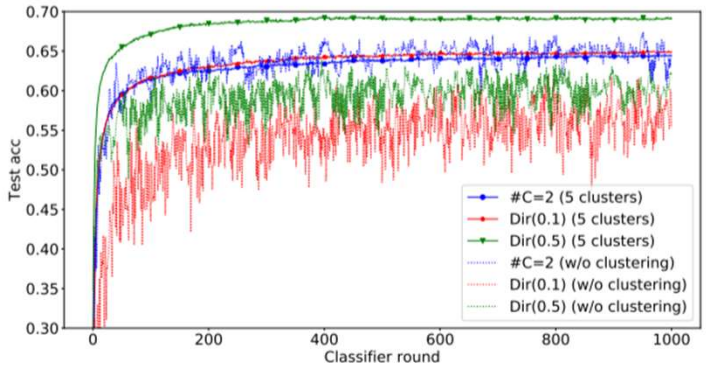


Figure 3: Training curves with clustering versus without clustering on CIFAR-10 (40 clients).

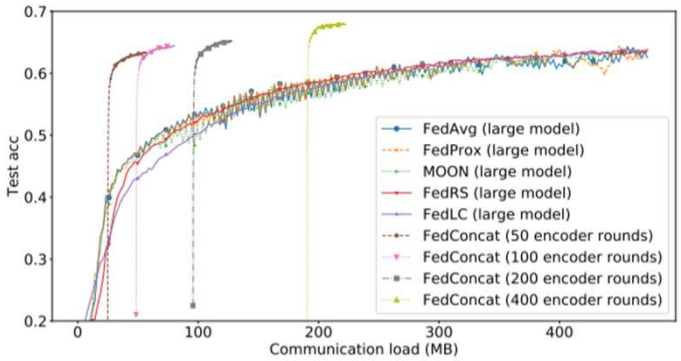


Figure 4: Comparing with baselines on the final global model of FedConcat on CIFAR-10 (40 clients, $\#C = 2$).

Thanks