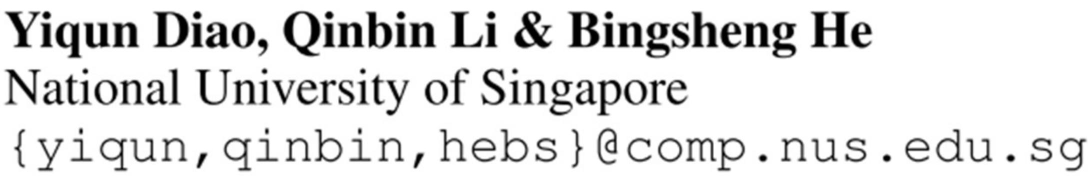
Towards Addressing Label Skews In One-shot Federated Learning



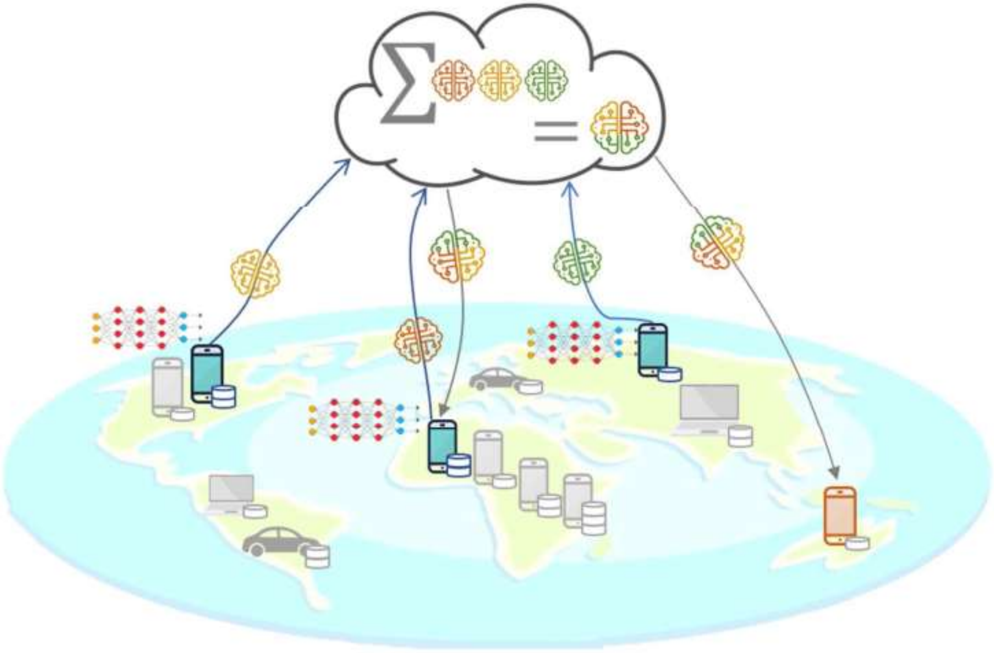
Yiqun Diao, Qinbin Li & Bingsheng He
National University of Singapore
`{yiqun,qinbin,hebs}@comp.nus.edu.sg`

ICLR 2023

Background

One-shot Federated Learning(One-shot FL)

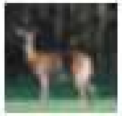
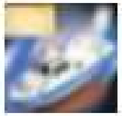
FL with only one communication.



Background

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

Model-averaging FL

FedAvg
(McMahan et al.,2016)

classic FL algorithm requires many communication rounds to train an effective global model.

FedProx
(Li et al.,2020)

adjusts the local training procedure to pull back local models from global model.

FedNova
(Wang et al.,2020)

normalizes local steps of each client during aggregation.

SCAFFOLD
(Karimireddy et al.,2020)

uses control variates (variance reduction) to correct for the client-drift in its local updates.

Related work

One-shot FL Algorithms

Close-set voting
(Guha et al.,2019)

collects local models as an ensemble for the final prediction and further proposes to use knowledge distillation on such ensemble with public data.

FedKT
(Li et al.,2021)

proposes consistent voting to improve the ensemble.

FedDF
(Lin et al.,2021)

uses public unlabeled/generated dataset to compute the KL divergence between teacher models and student model for distillation.

Related work

Open-set Recognition (OSR)

Generative-Optim
(Neal et al.,2019)

applies GANs to generate outliers that (1) is close to real samples, and (2) with high probability of outlier (low probability of any known class) from latent space.

PROSER
(Zhou et al.,2021)

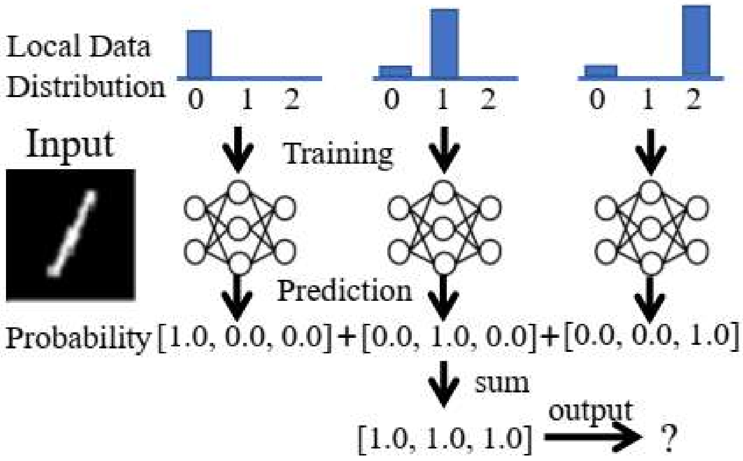
generates outliers by linear interpolation of embedding space among different classes and introduces an additional loss to increase the possibility of predicting a sample as “unknown” when discarding its true class.

$$L_{PROSER} = \sum_{(x,y)\in D} l(f(x), y) + \beta l(f(x)\backslash y, c) + \gamma \sum_{(x_i,x_j)\in D} l(\phi_{post}(\tilde{x}_{pre}), c)$$

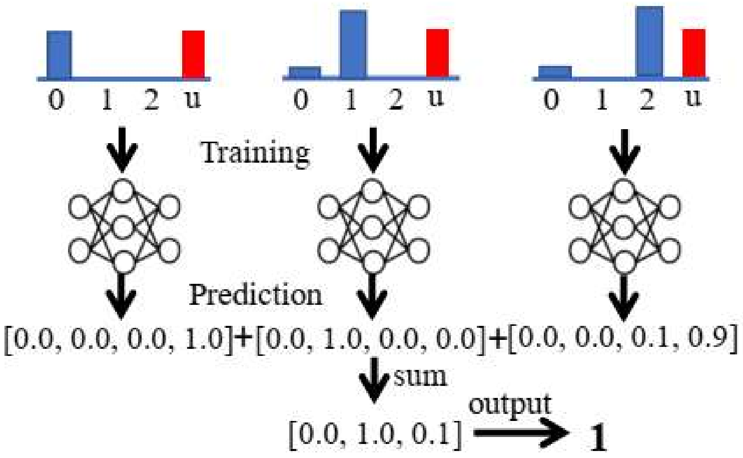
Motivation

Observation 1

The problem is that the predictions of close-set classification models are biased towards their seen classes as shown in Figure a. For voting, it would be better if models can be modest and admit unknown for its unseen classes as shown in Figure b.



(a) Traditional close-set voting

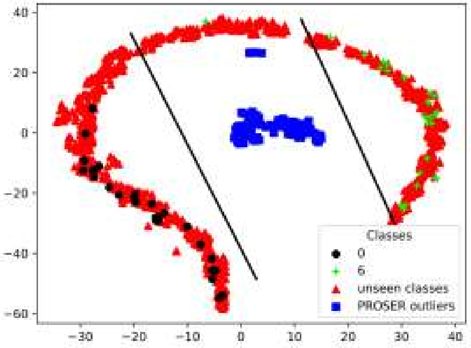


(b) Open-set voting framework ('u' denotes unknown class)

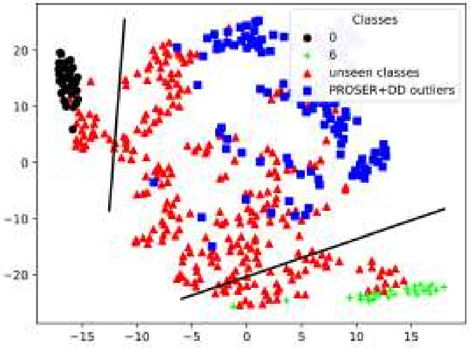
Motivation

Observation 2

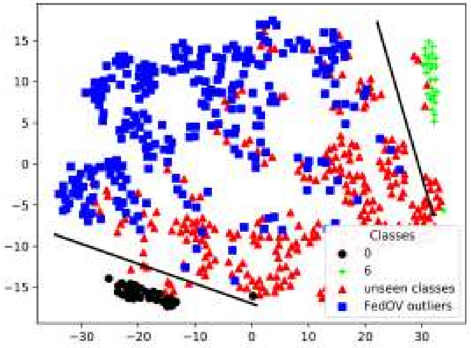
Directly applying PROSER in the local training of FL cannot achieve good local open-set classifiers. The generated outliers are quite limited and far from the training data when simply applying PROSER. The representations from the data of the seen and unseen classes are mixed and cannot be distinguished. To better suit OSR algorithms for label skews in FL, we need new techniques to generate outliers which should 1) be diverse and 2) be close to the seen classes.



(a) PROSER



(b) PROSER+DD



(c) FedOV

Data Destruction (DD)

First, in order to generate diverse outliers, they propose data destruction (DD) to directly generate outliers from true samples.

As opposed to applying to enhance the features, their DD applies intense data operations to corrupt the original key features, which is effective and efficient. Specifically, DD has two components: candidate data destruction operations and boosting outliers with a set of such operations.

a. Candidate Data Destruction Operations

(1) RandomCopyPaste;
(2) RandomSwap;
(3) RandomRotation;
(4) RandomErasing;
(5) GaussianBlur;
(6) RandomResizedCrop.

b. Boosting Outliers with Data Destruction Set

In each time, considering the above candidate operations as a set, they randomly sample one operation to generate an outlier each time. Then, in each batch of data during training, there exists diverse types of outliers generated by different operations.

Method

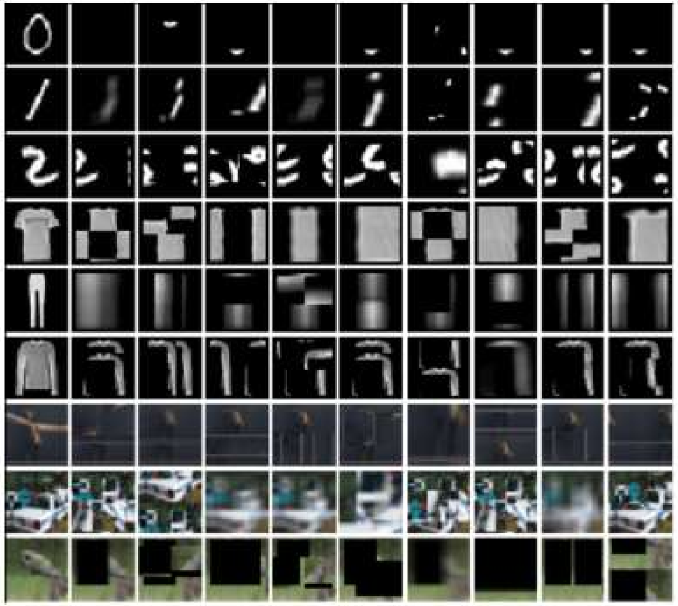
Adversarial Outlier Enhancement (AOE)

Second, in order to generate outliers that are even closer to true samples, they propose adversarial outlier enhancement (AOE) to learn a tighter boundary to surround the inliers.

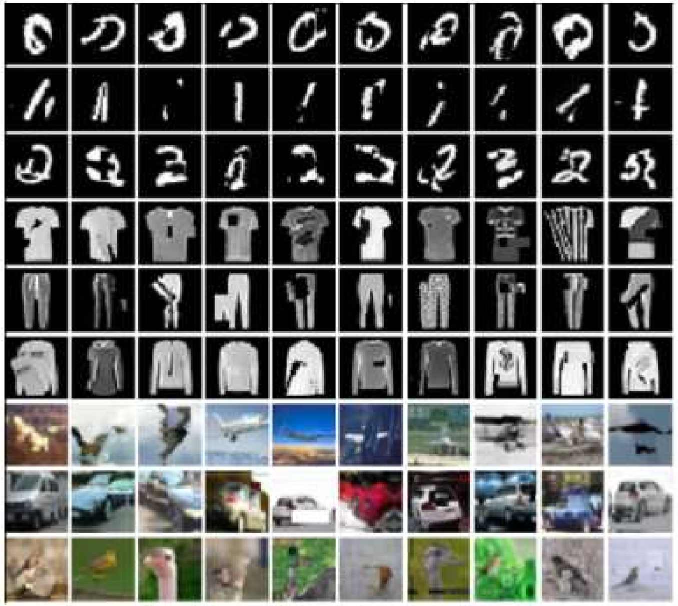
Specifically, suppose the client is training the model f with the generated outliers x′ by their data destruction method. They utilize FGSM to generate x″ such that the model wrongly outputs x″ as a seen sample with a high confidence. Then, the enhanced outliers x″ are used together with the generated outliers x′ as the unknown class to update the model. They call this method Adversarial Outlier Enhancement (AOE).

$$
\begin{aligned}
x' &= DataDestruction(x) \\
x'' &= FGSM(f_i, x', c) \\
L &= L_{PROSER} + L_{ce}(f_i(\{x', x''\}), c)
\end{aligned}
$$

Generated Outliers



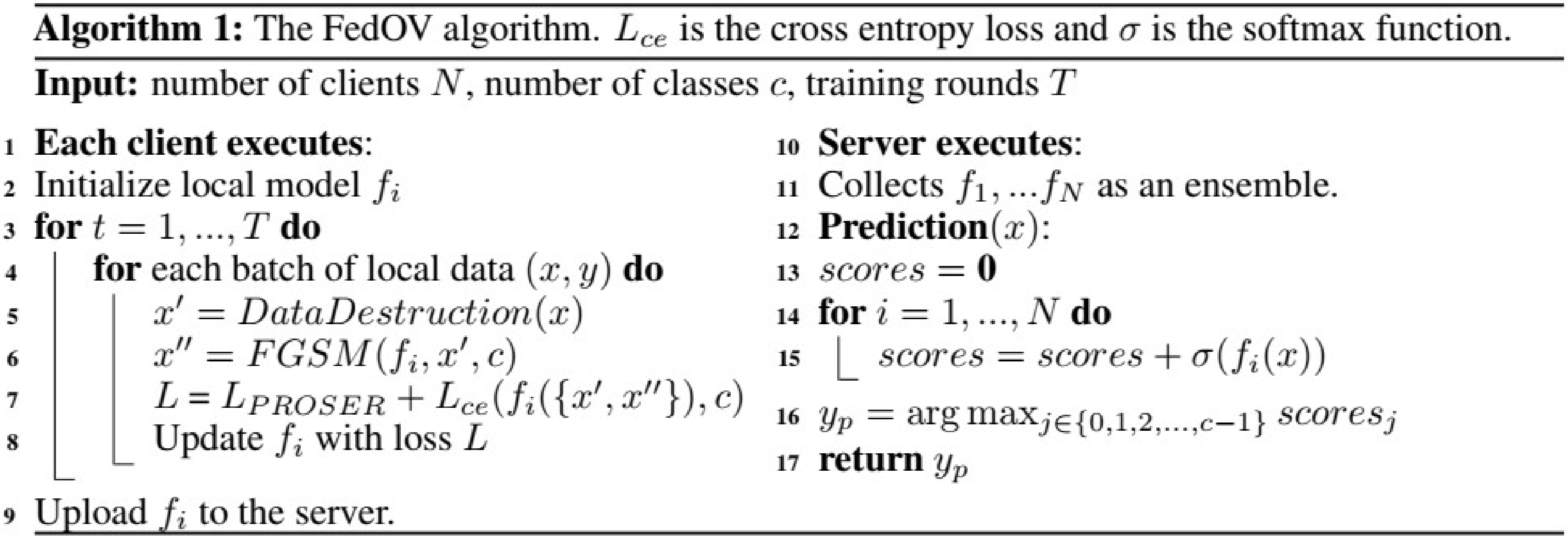
(a) Outliers generated by DD.



(b) Outliers after AOE.

The Overall Algorithm

The overall framework of open-set voting is described as follows. For the training stage, each client trains an open-set classifier locally and submits it to the server. For the prediction stage, the server sums up the prediction probability of all submitted models on the input sample while discarding their “unknown” channel. The class with maximum prediction probability is outputted as the prediction label.



Algorithm 1: The FedOV algorithm. L_{ce} is the cross entropy loss and σ is the softmax function.

```
Input: number of clients $N$, number of classes $c$, training rounds $T$
Each client executes:
Initialize local model $f_i$
for $t = 1, ..., T$ do
    for each batch of local data $(x, y)$ do
        $x' = DataDestruction(x)$
        $x'' = FGSM(f_i, x', c)$
        $L = L_{PROSER} + L_{ce}(f_i(\{x', x''\}), c)$
        Update $f_i$ with loss $L$
Upload $f_i$ to the server.
Server executes:
Collects $f_1, ...f_N$ as an ensemble.
Prediction$(x)$:
$scores = \mathbf{0}$
for $i = 1, ..., N$ do
    $scores = scores + \sigma(f_i(x))$
$y_p = \arg\max_{j \in \{0,1,2,...,c-1\}} scores_j$
return $y_p$
```

FedOV VS. Close-set Voting and Various FL Algorithms in One Round

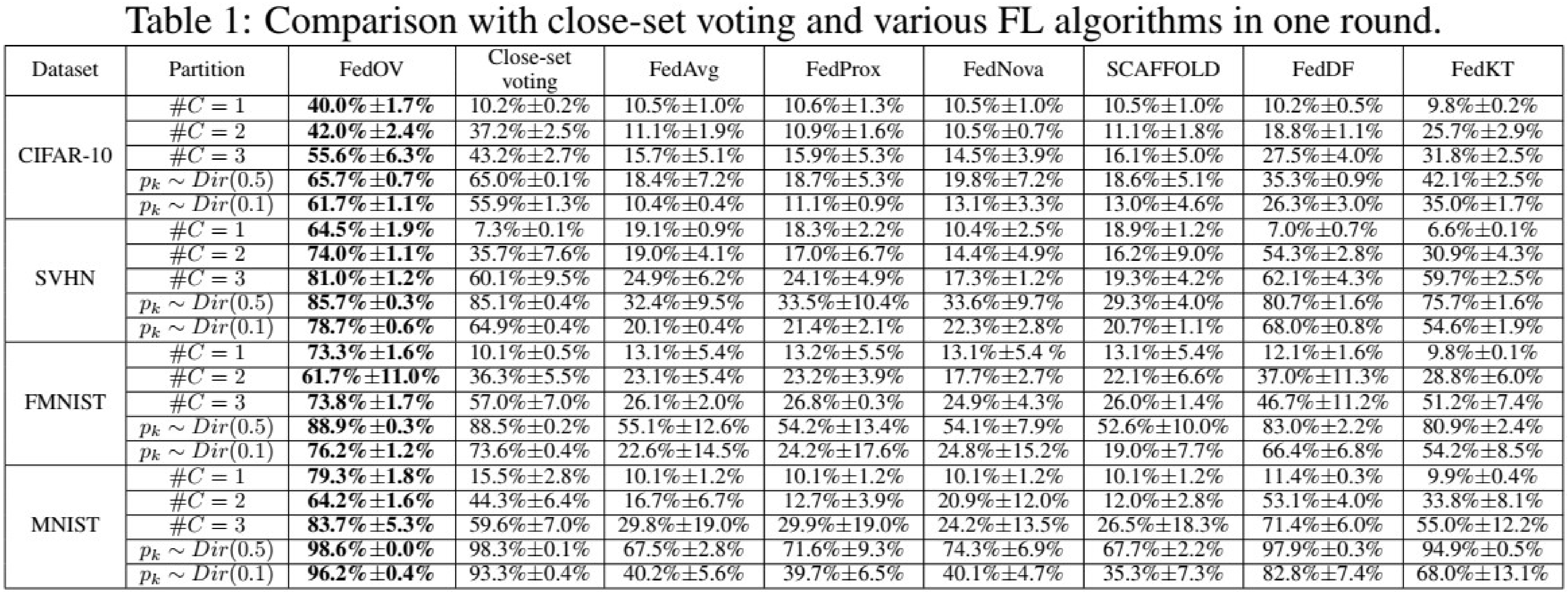


Table 1: Comparison with close-set voting and various FL algorithms in one round.

Dataset	Partition	FedOV	Close-set voting	FedAvg	FedProx	FedNova	SCAFFOLD	FedDF	FedKT
CIFAR-10	$\#C=1$	**40.0%±1.7%**	10.2%±0.2%	10.5%±1.0%	10.6%±1.3%	10.5%±1.0%	10.5%±1.0%	10.2%±0.5%	9.8%±0.2%
	$\#C=2$	**42.0%±2.4%**	37.2%±2.5%	11.1%±1.9%	10.9%±1.6%	10.5%±0.7%	11.1%±1.8%	18.8%±1.1%	25.7%±2.9%
	$\#C=3$	**55.6%±6.3%**	43.2%±2.7%	15.7%±5.1%	15.9%±5.3%	14.5%±3.9%	16.1%±5.0%	27.5%±4.0%	31.8%±2.5%
	$p_k \sim Dir(0.5)$	**65.7%±0.7%**	65.0%±0.1%	18.4%±7.2%	18.7%±5.3%	19.8%±7.2%	18.6%±5.1%	35.3%±0.9%	42.1%±2.5%
	$p_k \sim Dir(0.1)$	**61.7%±1.1%**	55.9%±1.3%	10.4%±0.4%	11.1%±0.9%	13.1%±3.3%	13.0%±4.6%	26.3%±3.0%	35.0%±1.7%
SVHN	$\#C=1$	**64.5%±1.9%**	7.3%±0.1%	19.1%±0.9%	18.3%±2.2%	10.4%±2.5%	18.9%±1.2%	7.0%±0.7%	6.6%±0.1%
	$\#C=2$	**74.0%±1.1%**	35.7%±7.6%	19.0%±4.1%	17.0%±6.7%	14.4%±4.9%	16.2%±9.0%	54.3%±2.8%	30.9%±4.3%
	$\#C=3$	**81.0%±1.2%**	60.1%±9.5%	24.9%±6.2%	24.1%±4.9%	17.3%±1.2%	19.3%±4.2%	62.1%±4.3%	59.7%±2.5%
	$p_k \sim Dir(0.5)$	**85.7%±0.3%**	85.1%±0.4%	32.4%±9.5%	33.5%±10.4%	33.6%±9.7%	29.3%±4.0%	80.7%±1.6%	75.7%±1.6%
	$p_k \sim Dir(0.1)$	**78.7%±0.6%**	64.9%±0.4%	20.1%±0.4%	21.4%±2.1%	22.3%±2.8%	20.7%±1.1%	68.0%±0.8%	54.6%±1.9%
FMNIST	$\#C=1$	**73.3%±1.6%**	10.1%±0.5%	13.1%±5.4%	13.2%±5.5%	13.1%±5.4 %	13.1%±5.4%	12.1%±1.6%	9.8%±0.1%
	$\#C=2$	**61.7%±11.0%**	36.3%±5.5%	23.1%±5.4%	23.2%±3.9%	17.7%±2.7%	22.1%±6.6%	37.0%±11.3%	28.8%±6.0%
	$\#C=3$	**73.8%±1.7%**	57.0%±7.0%	26.1%±2.0%	26.8%±0.3%	24.9%±4.3%	26.0%±1.4%	46.7%±11.2%	51.2%±7.4%
	$p_k \sim Dir(0.5)$	**88.9%±0.3%**	88.5%±0.2%	55.1%±12.6%	54.2%±13.4%	54.1%±7.9%	52.6%±10.0%	83.0%±2.2%	80.9%±2.4%
	$p_k \sim Dir(0.1)$	**76.2%±1.2%**	73.6%±0.4%	22.6%±14.5%	24.2%±17.6%	24.8%±15.2%	19.0%±7.7%	66.4%±6.8%	54.2%±8.5%
MNIST	$\#C=1$	**79.3%±1.8%**	15.5%±2.8%	10.1%±1.2%	10.1%±1.2%	10.1%±1.2%	10.1%±1.2%	11.4%±0.3%	9.9%±0.4%
	$\#C=2$	**64.2%±1.6%**	44.3%±6.4%	16.7%±6.7%	12.7%±3.9%	20.9%±12.0%	12.0%±2.8%	53.1%±4.0%	33.8%±8.1%
	$\#C=3$	**83.7%±5.3%**	59.6%±7.0%	29.8%±19.0%	29.9%±19.0%	24.2%±13.5%	26.5%±18.3%	71.4%±6.0%	55.0%±12.2%
	$p_k \sim Dir(0.5)$	**98.6%±0.0%**	98.3%±0.1%	67.5%±2.8%	71.6%±9.3%	74.3%±6.9%	67.7%±2.2%	97.9%±0.3%	94.9%±0.5%
	$p_k \sim Dir(0.1)$	**96.2%±0.4%**	93.3%±0.4%	40.2%±5.6%	39.7%±6.5%	40.1%±4.7%	35.3%±7.3%	82.8%±7.4%	68.0%±13.1%

Experiment

Ablation Study

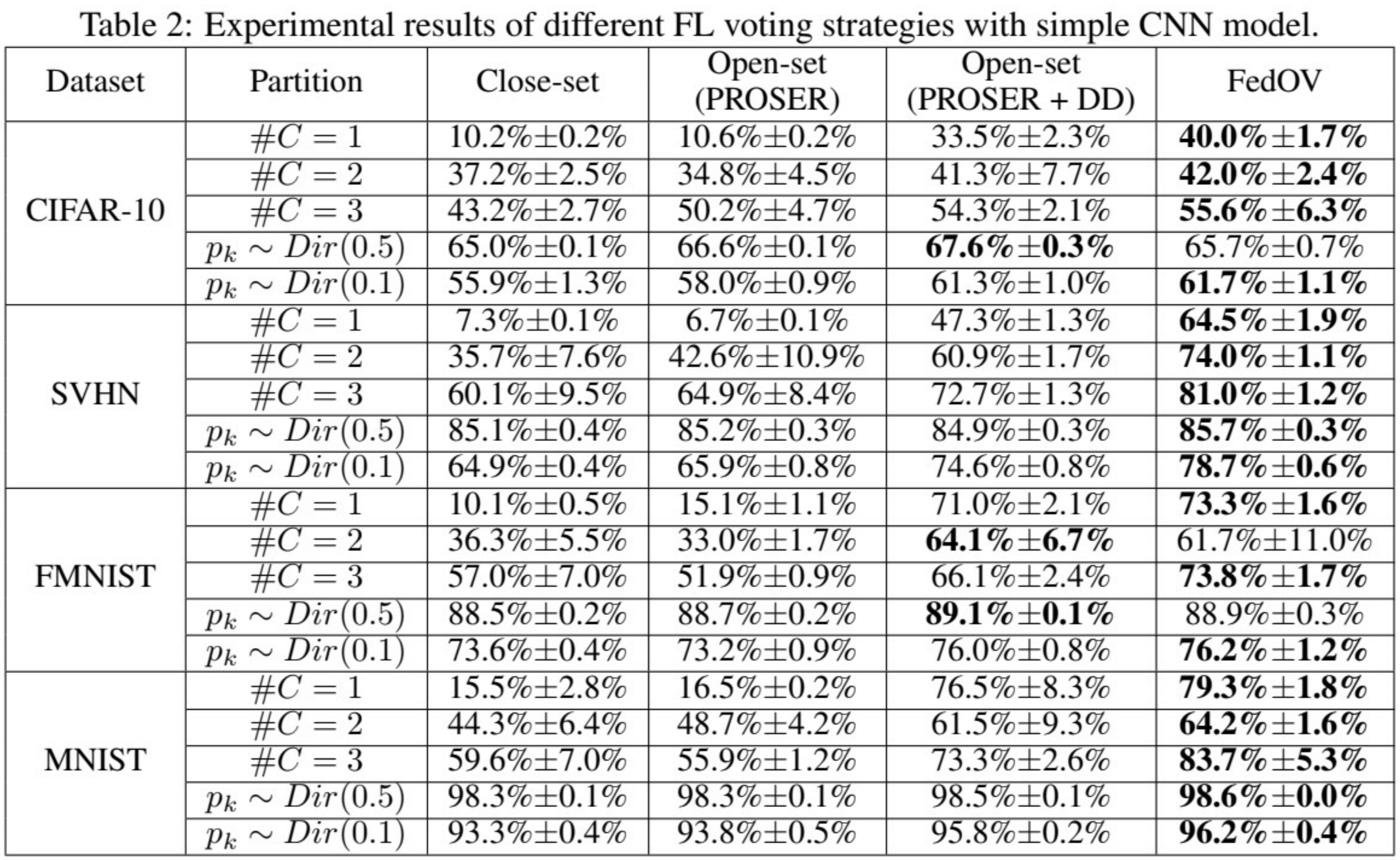


Table 2: Experimental results of different FL voting strategies with simple CNN model.

Dataset	Partition	Close-set	Open-set (PROSER)	Open-set (PROSER + DD)	FedOV
CIFAR-10	$\#C = 1$	10.2%±0.2%	10.6%±0.2%	33.5%±2.3%	**40.0%±1.7%**
	$\#C = 2$	37.2%±2.5%	34.8%±4.5%	41.3%±7.7%	**42.0%±2.4%**
	$\#C = 3$	43.2%±2.7%	50.2%±4.7%	54.3%±2.1%	**55.6%±6.3%**
	$p_k \sim Dir(0.5)$	65.0%±0.1%	66.6%±0.1%	**67.6%±0.3%**	65.7%±0.7%
	$p_k \sim Dir(0.1)$	55.9%±1.3%	58.0%±0.9%	61.3%±1.0%	**61.7%±1.1%**
SVHN	$\#C = 1$	7.3%±0.1%	6.7%±0.1%	47.3%±1.3%	**64.5%±1.9%**
	$\#C = 2$	35.7%±7.6%	42.6%±10.9%	60.9%±1.7%	**74.0%±1.1%**
	$\#C = 3$	60.1%±9.5%	64.9%±8.4%	72.7%±1.3%	**81.0%±1.2%**
	$p_k \sim Dir(0.5)$	85.1%±0.4%	85.2%±0.3%	84.9%±0.3%	**85.7%±0.3%**
	$p_k \sim Dir(0.1)$	64.9%±0.4%	65.9%±0.8%	74.6%±0.8%	**78.7%±0.6%**
FMNIST	$\#C = 1$	10.1%±0.5%	15.1%±1.1%	71.0%±2.1%	**73.3%±1.6%**
	$\#C = 2$	36.3%±5.5%	33.0%±1.7%	**64.1%±6.7%**	61.7%±11.0%
	$\#C = 3$	57.0%±7.0%	51.9%±0.9%	66.1%±2.4%	**73.8%±1.7%**
	$p_k \sim Dir(0.5)$	88.5%±0.2%	88.7%±0.2%	**89.1%±0.1%**	88.9%±0.3%
	$p_k \sim Dir(0.1)$	73.6%±0.4%	73.2%±0.9%	76.0%±0.8%	**76.2%±1.2%**
MNIST	$\#C = 1$	15.5%±2.8%	16.5%±0.2%	76.5%±8.3%	**79.3%±1.8%**
	$\#C = 2$	44.3%±6.4%	48.7%±4.2%	61.5%±9.3%	**64.2%±1.6%**
	$\#C = 3$	59.6%±7.0%	55.9%±1.2%	73.3%±2.6%	**83.7%±5.3%**
	$p_k \sim Dir(0.5)$	98.3%±0.1%	98.3%±0.1%	98.5%±0.1%	**98.6%±0.0%**
	$p_k \sim Dir(0.1)$	93.3%±0.4%	93.8%±0.5%	95.8%±0.2%	**96.2%±0.4%**

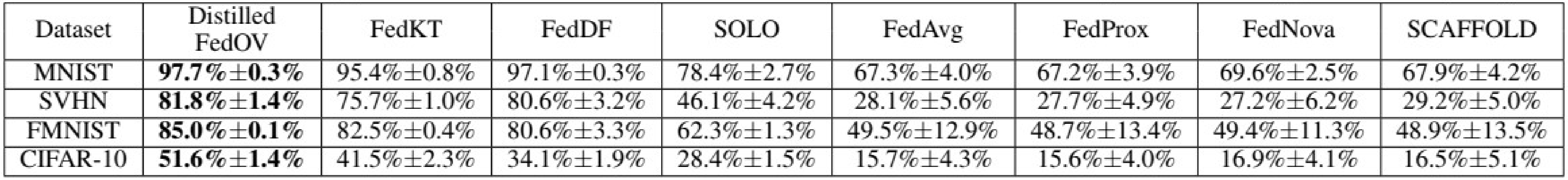
Experiment

Combining with Knowledge Distillation

One shortage of FedOV is that the final model is an ensemble of local models, therefore its prediction and storage costs may be large especially when the number of clients is large.

Assuming that there exist unlabeled public data on the server, they can combine FedOV with knowledge distillation called Distilled FedOV to transform the ensemble of local models into a single global model. Then, it can significantly reduce the storage and prediction costs of the final model.

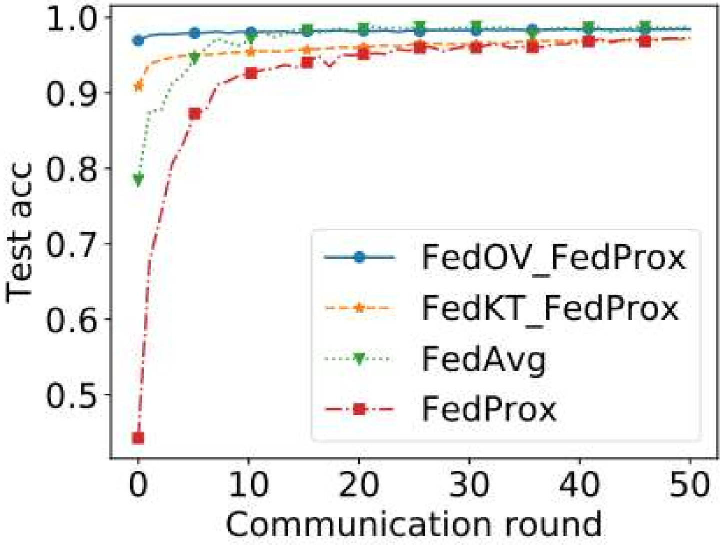
Table 3: Comparing distilled FedOV with the other baselines. The partition is $p_k \sim Dir(0.5)$.



Dataset	Distilled FedOV	FedKT	FedDF	SOLO	FedAvg	FedProx	FedNova	SCAFFOLD
MNIST	**97.7%±0.3%**	95.4%±0.8%	97.1%±0.3%	78.4%±2.7%	67.3%±4.0%	67.2%±3.9%	69.6%±2.5%	67.9%±4.2%
SVHN	**81.8%±1.4%**	75.7%±1.0%	80.6%±3.2%	46.1%±4.2%	28.1%±5.6%	27.7%±4.9%	27.2%±6.2%	29.2%±5.0%
FMNIST	**85.0%±0.1%**	82.5%±0.4%	80.6%±3.3%	62.3%±1.3%	49.5%±12.9%	48.7%±13.4%	49.4%±11.3%	48.9%±13.5%
CIFAR-10	**51.6%±1.4%**	41.5%±2.3%	34.1%±1.9%	28.4%±1.5%	15.7%±4.3%	15.6%±4.0%	16.9%±4.1%	16.5%±5.1%

Extension to Multiple Rounds

Moreover, considering the final distilled model as the initialized model for iterative federated learning algorithms (e.g., FedAvg, FedProx, etc), they can conduct multi-round federated learning to further improve the model.



Experiment

Scalability

They test the scalability of FedOV by varying the number of clients.

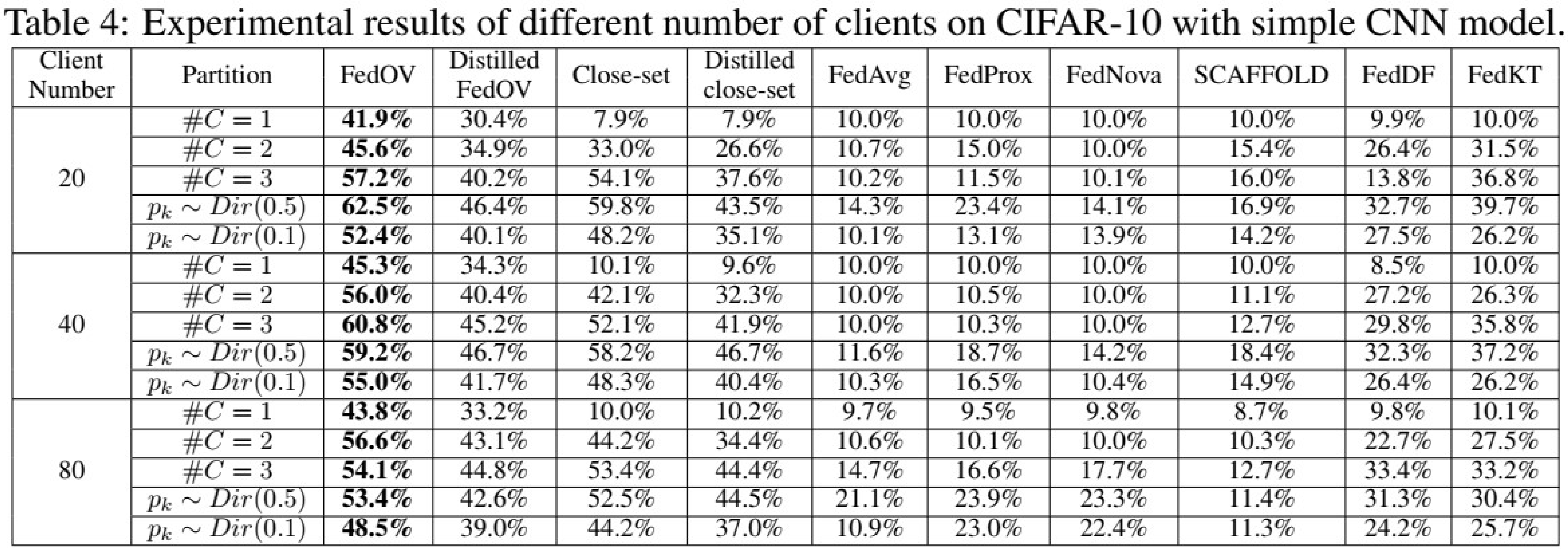


Table 4: Experimental results of different number of clients on CIFAR-10 with simple CNN model.

Client Number	Partition	FedOV	Distilled FedOV	Close-set	Distilled close-set	FedAvg	FedProx	FedNova	SCAFFOLD	FedDF	FedKT
20	$\#C = 1$	**41.9%**	30.4%	7.9%	7.9%	10.0%	10.0%	10.0%	10.0%	9.9%	10.0%
	$\#C = 2$	**45.6%**	34.9%	33.0%	26.6%	10.7%	15.0%	10.0%	15.4%	26.4%	31.5%
	$\#C = 3$	**57.2%**	40.2%	54.1%	37.6%	10.2%	11.5%	10.1%	16.0%	13.8%	36.8%
	$p_k \sim Dir(0.5)$	**62.5%**	46.4%	59.8%	43.5%	14.3%	23.4%	14.1%	16.9%	32.7%	39.7%
	$p_k \sim Dir(0.1)$	**52.4%**	40.1%	48.2%	35.1%	10.1%	13.1%	13.9%	14.2%	27.5%	26.2%
40	$\#C = 1$	**45.3%**	34.3%	10.1%	9.6%	10.0%	10.0%	10.0%	10.0%	8.5%	10.0%
	$\#C = 2$	**56.0%**	40.4%	42.1%	32.3%	10.0%	10.5%	10.0%	11.1%	27.2%	26.3%
	$\#C = 3$	**60.8%**	45.2%	52.1%	41.9%	10.0%	10.3%	10.0%	12.7%	29.8%	35.8%
	$p_k \sim Dir(0.5)$	**59.2%**	46.7%	58.2%	46.7%	11.6%	18.7%	14.2%	18.4%	32.3%	37.2%
	$p_k \sim Dir(0.1)$	**55.0%**	41.7%	48.3%	40.4%	10.3%	16.5%	10.4%	14.9%	26.4%	26.2%
80	$\#C = 1$	**43.8%**	33.2%	10.0%	10.2%	9.7%	9.5%	9.8%	8.7%	9.8%	10.1%
	$\#C = 2$	**56.6%**	43.1%	44.2%	34.4%	10.6%	10.1%	10.0%	10.3%	22.7%	27.5%
	$\#C = 3$	**54.1%**	44.8%	53.4%	44.4%	14.7%	16.6%	17.7%	12.7%	33.4%	33.2%
	$p_k \sim Dir(0.5)$	**53.4%**	42.6%	52.5%	44.5%	21.1%	23.9%	23.3%	11.4%	31.3%	30.4%
	$p_k \sim Dir(0.1)$	**48.5%**	39.0%	44.2%	37.0%	10.9%	23.0%	22.4%	11.3%	24.2%	25.7%

FedOV still achieves the best accuracy when increasing the number of clients. Moreover, with the help of knowledge distillation, Distilled FedOV can outperform distilled close-set and other iterative FL algorithms with the same storage and inference cost.

Thanks