Supplementary Material: Data-Free Diversity-Based Ensemble Selection For One-Shot Federated Learning

1 Execution process of DeDES

Fig. 1 gives an case study of `DeDES` when we use the last layer of model parameters as the model representation and *Kernel-PCA* as the dimension reduction method. Under this case, we assume that all models are well-trained and no low-quality model(s) so that we don't need to do the model filter; the choice of clustering method will depend on the data partition which are shown in section 3.

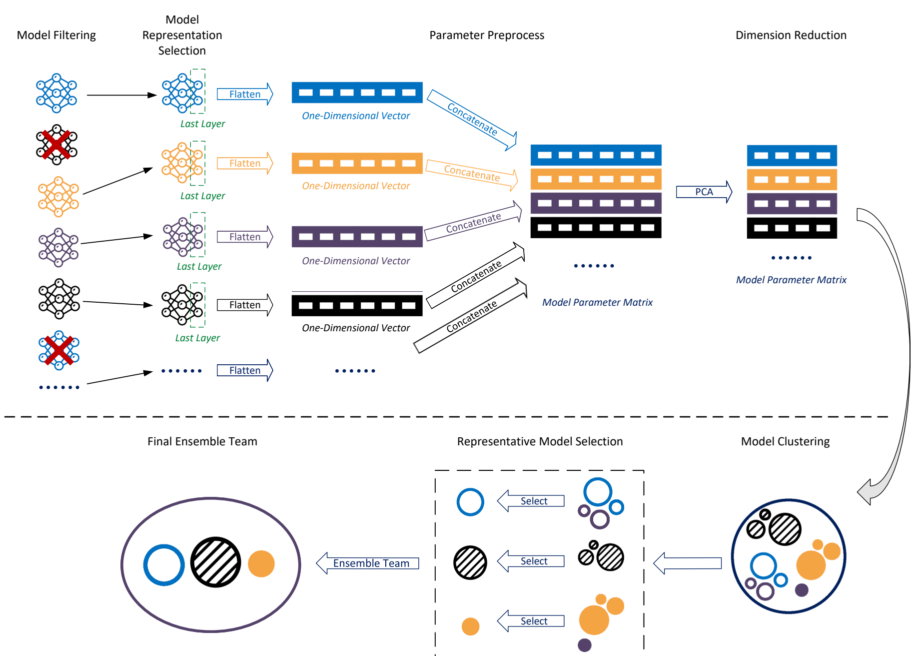


Fig. 1: A case study of the execution process of `DeDES` framework, where Kernel-PCA is used as the dimension reduction method.

2 MODEL FILTER ALGORITHM

As in Alg. 1 from the ***main paper***, we use the `OutlierFilter` to obtain the outlier models $\mathcal{O}$ based on the model scores $\mathcal{S}$ provided from each party, which can be the local validation accuracy or prediction confidence. `OutlierFilter` can be any score-based unsupervised outlier detection methods, we apply a variation of the commonly-used box-plot in our experiment, which is show in Alg. 1 of this *supplementary material.*

As shown in Fig. 2, due to the inappropriate learning rate, the party 8 (p8) was not well-trained (not converge) thus has a bad validation accuracy (20.9%), our method can successfully remove this party's model when selecting ensemble teams, which can improve the final performance of ensemble learning.

Algorithm 1: Model filter `OutlierFilter` method

Input: model set $\mathcal{M}$, truncated threshold pair (p_{low}, p_{high}), interval scale s, and model scores $\mathcal{S} = \{s_i\}_{i=1}^{m}$.

Output: Outlier model set $\mathcal{O}$.

▷ *Sort the score set $\mathcal{S}$, such as local validation accuracy, by ascending order.*

$\mathcal{S} \leftarrow AscendingSort(\mathcal{S})$

▷ *Get the value of the p_{low}-th, p_{high}-th element of $\mathcal{S}$.*

$q_{low}, q_{high} \leftarrow \mathcal{S}p_{low}, \mathcal{S}p_{high}$

$interval \leftarrow q_{high} - q_{low}$

$outlier\_threshold \leftarrow q_{low} - s * interval$

$\mathcal{O} \leftarrow \emptyset$

for $i = 1$ *to* m **do**

  if $s_i < outlier\_threshold$ **then**

    $\mathcal{O} \leftarrow \mathcal{O} \cup \{M_i\}$

return $\mathcal{O}$

3 Experiment Setup

3.1 Data partition method explanation

Examples of the four types of data partitions are shown below.

1. Homogeneous (homo): all parties have the same amount of data and the same distribution of data, i.e., this data distribution is independent and identical distribution (IID).

E.g, if we divide the MNIST dataset which has 280,000 images into 100 parties with *homo* setting, then all parties will have the same amount of data (all round 280,000 / 100 = 2,800) and the same label distribution of data (every party will have around 2,800 / 10 = 280 images for every label from 0 to 9).

2. IID but with different quantity (iid-dq): all parties have the same distribution of data but with different data amount. E.g., every party will have similar

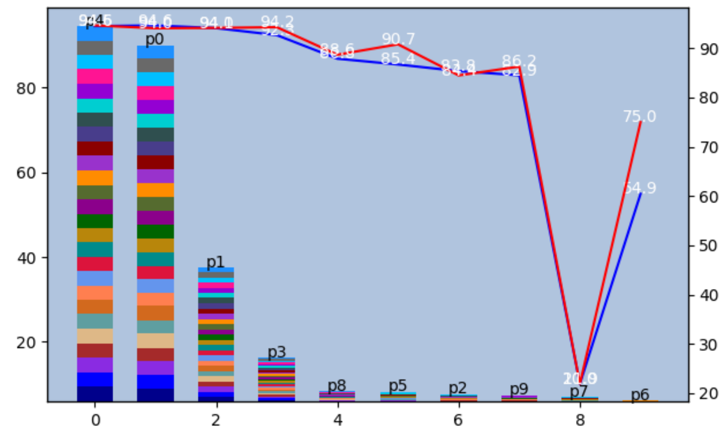


Fig. 2: Data distributions for the iid-dq partition of the EMNIST letters dataset with local validation accuracy and test accuracy for the whole test set when m=10. Every dot in the red line shows the best local validation accuracy after training for 200 epochs while the blue line shows the whole test accuracy of the k-th party pk.

amount of images of 0 to 9, but the amount of total images is different for each party.

3. Non-IID label distribution skew (noniid-lds): the label distributions $P(y_i)$ vary across parties. Each party will be allocated a proportion of the samples of each label according to *Dirichlet Distribution*. That is, the label distribution of each party is skewed and data amount among parties is different with each other.

4. Extreme Non-IID setting with only k labels of each party (noniid-lk): Under this partition setting, each party will only have k labels of data. For example, if we set $k = 3$, then each party will only have 3 labels of data, e.g., 0, 3 and 7.

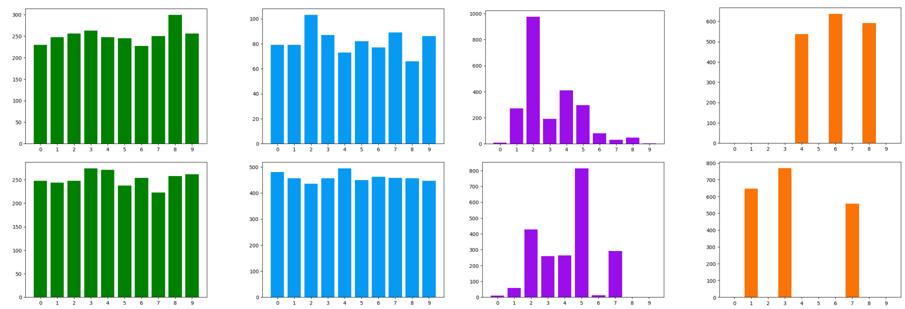
Figure 3 shows the distribution of data amount and label distribution of each party under these four different data partition settings.

3.2 Component Configurations

In this subsection, we will describe the default configurations of our `DeDES` framework for the experiments in the main paper.

For local model training, we utilize the *SGD* optimizer to train model for every party for 200 epochs with learning rate started at 0.1 and decrease at later epochs. After the training finished, we select the model with the highest local validation accuracy of each party to upload to the server of model market.

In our experiments, we select the final model layer as the model representation in our experiments; we utilize the $MINMAX$ scaler to preprocess the



(a) $homo$ (b) $iid - dq$ (c) $noniid - lds$ (d) $noniid - l3$

Fig. 3: Example distribution of four dataset partition strategies for the *EMNIST digits* dataset with party number of 100. For each of subfigures (a), (b), (c), and (d), we randomly picked two parties from a total of 100 parties and depicted their distributions of data quantity and label distribution.

Model Representation Matrix, and the *Gaussian Normalization* scaler to preprocess the *Label Distribution* ground-truth input data; we do not utilize any dimension reduction strategies for the model representation matrix because the final layer of model parameters are already few in number.

Spectral clustering is used for the *homo* and *iid-dq* distribution partition methods; *K-Means* clustering is used for the *noniid-lds* and *noniid-lk* partition methods.

Our proposed model representative selection approach is utilized to determine the representative model within each cluster; as we have frequently stated, we use plurality voting to perform ensemble learning; we use the test accuracy as the evaluation metric for all the methods.

To compare the efficiency of different methods, we also recorded the running time of all methods, details will be shown in the experiment section. More details about the experimental settings, such as the devices we use or examples about four data partitions, are mentioned in the *supplementary material*.

3.3 Environment

All our experiments are running on a single machine with 1TB RAM and 256 cores AMD EPYC 7742 64-Core Processor @ 3.4GHz CPU. The GPU we use is NVIDIA A100 SXM4 with 40GB memory. The OS we use is Ubuntu 20.04.4 LTS.

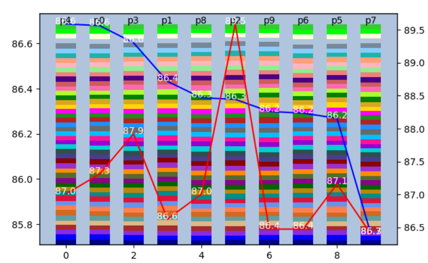
Our software environment settings are: Python 3.9.12, PyTorch 1.12.1 with CUDA 11.6.

All experiments were tested 3 times, and the average results are presented in the experiment section.

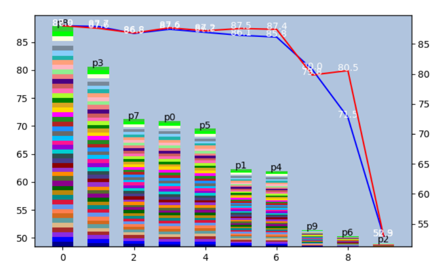
4 ADDITIONAL EXPERIMENTS

4.1 The importance of ensemble learning

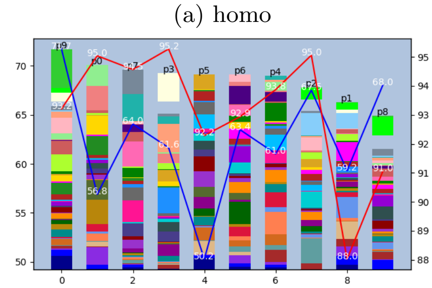
Fig. 4 shows the data distributions of the EMNIST balanced dataset with local validation accuracy and test accuracy for the whole test set when m=10. We can see that the capacity of a single model is weak, even their local validation accuracy (validation accuracy for their own dataset) is high, therefore validates the importance of ensemble learning which can utilize the collaborative power of multiple models.



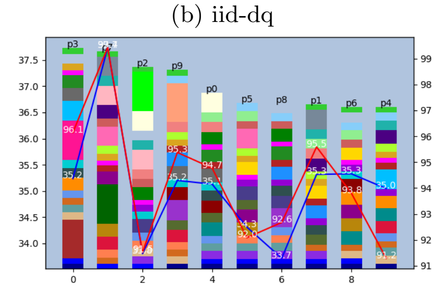
(a) homo



(b) iid-dq



(c) noniid-lds



(d) noniid-l4

Fig. 4: Different data distributions of the EMNIST balanced dataset with local validation accuracy and test accuracy for the whole test set when m=10. Every dot in the red line shows the best local validation accuracy after training for 200 epochs while the blue line shows the whole test accuracy of the k-th party pk.

4.2 Performance Analysis (Supplementary)

Table. 1 shows the performance of different methods when we apply them on the 5 datasets with 4 partitions with different model structures than the main paper. We can see that when $m = 200, K = 120$ for the EMNIST Letters Dataset, the test

Table 1: Test accuracy (%) comparison for different dataset on different data partitions and model structures. The best and next best methods are **bolded** and underlined, respectively. If our `DeDES` method is better than the *LD* ground-truth method, the value of *LD* method will be marked in skyblue.

Dataset	Partition	m	K	`DeDES`	AS	CV	DS	RS	FedAvg	MeanAvg	LD	Oracle
EMNIST Digits (Resnet-50)	homo	400	150	96.33	96.46	**96.65**	96.23	96.31	10.25	10.22	96.70	99.71
	iid-dq	400	150	**98.13**	98.01	98.08	98.07	97.94	10.63	10.64	98.13	99.70
	noniid-ld	400	150	**96.52**	96.50	86.58	95.48	96.04	10.24	10.19	94.80	99.70
	noniid-l3	400	150	96.64	**96.81**	89.21	58.59	94.72	9.74	9.61	96.83	99.67
EMNIST Letters (Resnet-50)	homo	200	120	78.01	77.91	**78.77**	77.13	77.08	3.86	3.89	77.41	94.76
	iid-dq	200	120	88.88	88.88	88.88	**88.89**	88.45	3.82	3.80	88.85	95.13
	noniid-ld	200	120	79.78	**80.55**	79.53	77.27	78.65	4.23	4.28	78.37	94.86
	noniid-l8	200	120	81.10	**82.79**	80.54	78.86	80.28	3.74	3.69	82.33	95.08
EMNIST Balanced (Resnet-50)	homo	100	50	80.12	80.11	**80.33**	79.38	79.20	2.15	2.18	78.93	89.44
	iid-dq	100	50	85.68	85.68	85.68	**85.71**	84.54	2.11	2.14	85.76	89.12
	noniid-ld	100	50	76.34	**77.56**	71.64	74.16	75.00	2.23	2.21	70.75	89.52
	noniid-l18	100	50	77.99	**80.28**	77.71	77.98	77.40	2.13	2.13	78.58	89.39
CIFAR10 (Densenet)	homo	200	100	**46.84**	46.30	46.47	45.37	45.68	10.49	10.08	46.34	90.57
	iid-dq	200	100	52.38	53.01	**53.55**	53.47	51.76	10.45	10.56	54.03	90.38
	noniid-ld	200	100	**44.90**	43.91	40.89	40.54	41.49	10.38	10.52	41.61	91.01
	noniid-l4	200	100	47.04	**48.51**	46.08	40.92	45.43	9.71	9.47	47.45	90.79
CIFAR100 (Deep Layer Aggregation)	homo	20	12	23.01	**24.80**	22.47	22.27	22.63	0.95	0.94	22.12	52.63
	iid-dq	20	12	39.18	39.18	39.18	39.18	37.04	1.01	0.95	39.18	55.61
	noniid-ld	20	12	22.29	**25.11**	21.37	21.88	21.15	0.94	0.94	21.42	55.94
	noniid-l45	20	12	24.41	**27.42**	24.07	23.08	23.59	0.87	0.85	23.19	54.90

accuracy of `DeDES` and AS, CV are the same, which means they both selected the same ensemble teams.

Table 2 enumerated the accuracy of all 1024 teams and the ranking of selected teams generated by different methods for another data partition of the EMNIST balanced dataset. We can see that the ensemble team selected by `DeDES` is ranked higher than other baseline methods, which validates the efficacy of our method.

4.3 Impact on Efficiency (Supplementary)

Fig.5 gives another plot of the relationship between K and test accuracy for the EMNIST Balanced dataset. The conclusion remains the same as the main paper that we don't have to select all models to form an ensemble team for most of the cases, which saves the inference time and also keeps good performance.

4.4 Ablation Studies

- **Performance Comparison on different model structures and datasets** Our method is solid for various model structures and datasets. As shown in Table 2. in the main paper and Table. 1 of this supplementary material, we can see that no matter what model structures/datasets we use, our method can achieve better performance than other baselines methods for ensemble learning.

Method	Rank	Accuracy (%)
`DeDES`	**31/1024**	**55.6**
CV	40/1024	55.25
AS	50/1024	54.9
LD	78/1024	54.57
DS	136/1024	53.12
RS	311/1024	51.09

Table 2: Complete inspection on ensemble teams for *EMNIST Balanced* dataset with $m = 10$, *noniid-l*18 partition.

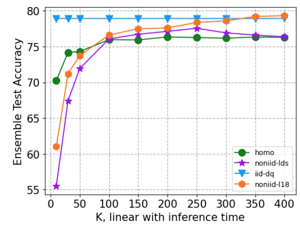


Fig. 5: The relationship of K and Ensemble Test Accuracy of `DeDES` for the *EMNIST Balanced* Dataset when m=400.

The remaining parts are still updating because the author has been affected by COVID-19 and is now having a high fever, but we will finish this supp by Dec 25, 2022, thank you for your patience!

- **Performance Comparison on different model representation** It is better to use the models' later layer's parameters for representation than utilizing their front layer's parameters.
- **Importance of Dimension Reduction Methods.** *Kernel-PCA* is better than other dimension reduction methods such as *PCA* and *non-compression*.
- **Clustering/Diversity validation** Our method can really cluster similar models together and the whole team's diversity is higher than other methods.