Contents of Appendix

7. Proof of Aggregation Weights

Theorem 7.1 (Aggregation weights)**.** *Define the following objective function*

$$\max_{p^t_{i,g}} \mathcal{L}_{agg} = \sum_{i \in \mathcal{S}^t} p^t_{i,g} \left(\sum_{\tau=1}^{t} \gamma^{t-\tau} \mathit{Sim}(\beta_i, \mathcal{S}^\tau) \right) + \lambda_1 \sum_{i \in \mathcal{S}^t} p^t_{i,g} \log \frac{q_i^t}{p^t_{i,g}} + \lambda_0 (\sum_{i \in \mathcal{S}^t} p^t_{i,g} - 1) \,, \tag{5}$$

where $p^t_{i,g}$ is the aggregation weights on communication round t, S is the similarity function, and q_i^t is a prior distribution. Solving this optimization problem, the optimal $p^t_{i,g}$ is given by

$$p^t_{i,g} = \frac{q_i^t \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \mathit{Sim}(\beta_i, \mathcal{S}^\tau) \right)}{\sum_{j \in \mathcal{S}^t} q_j^t \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \mathit{Sim}(\beta_j, \mathcal{S}^\tau) \right)} \,. \tag{6}$$

Proof. Taking the derivation, we have

$$\frac{\partial \mathcal{L}_{agg}}{\partial p^t_{i,g}} = \sum_{\tau=1}^{t} \gamma^{t-\tau} \text{Sim}(\beta_i, \mathcal{S}^\tau) + \lambda_1 \left(\log q_i^t - \log p^t_{i,g} - 1 \right) + \lambda_0 \,, \tag{7}$$

then we have

$$p^t_{i,g} = \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \text{Sim}(\beta_i, \mathcal{S}^\tau) + \log q_i^t - 1 + \frac{\lambda_0}{\lambda_1} \right) \,. \tag{8}$$

Because $\sum_{i \in \mathcal{S}^t} p^t_{i,g} = 1$, we have

$$1 - \frac{\lambda_0}{\lambda_1} = \log \left(\sum_{i \in \mathcal{S}^t} \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \text{Sim}(\beta_i, \mathcal{S}^\tau) + \log q_i^t \right) \right) \tag{9}$$

$$= \log \left(\sum_{i \in \mathcal{S}^t} q_i^t \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \text{Sim}(\beta_i, \mathcal{S}^\tau) \right) \right) \,, \tag{10}$$

Then combine Equations (8) and (10) we have

$$p^t_{i,g} = \frac{q_i^t \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \text{Sim}(\beta_i, \mathcal{S}^\tau) \right)}{\sum_{j \in \mathcal{S}^t} q_j^t \exp\left(\frac{1}{\lambda_1} \sum_{\tau=1}^{t} \gamma^{t-\tau} \text{Sim}(\beta_j, \mathcal{S}^\tau) \right)} \tag{11}$$

□

8. Related Works

Distribution shifts in FL. Federated Learning (FL) is introduced as a methodology for training machine learning models in a distributed manner, wherein local data is retained and not exchanged between the central server and individual clients. FedAvg [37, 41], serving as a foundational algorithm in this domain, advocates the use of local Stochastic Gradient Descent (local SGD) to alleviate the communication burden. Nevertheless, the performance of FL algorithms is substantially impeded by distribution shifts among clients. Addressing these local distribution shifts has emerged as a primary focus in FL research [18, 25, 27, 28, 34]. Many existing works address label distribution shifts by incorporating additional regularization terms [16, 28, 29, 34, 42], enhancing feature learning [32, 51, 55, 70], and improving classifiers [35, 40]. Regarding feature distribution shifts, the majority of FL methods concentrate on the out-of-domain generalization problem. This objective aims to train robust models capable of generalizing to previously unseen feature distributions [17, 33, 45]. Approaches include investigating special cases [49], integrating domain generalization algorithms in FL scenarios, such as domain-robust optimization [10, 44], and training domain-invariant features [13, 47, 50, 53, 57]. Notably, recent research has also considered concept shifts by leveraging clustering methods [15, 19, 26]. In this study, we address the challenge of distribution shifts in FL from another perspective—enhancing the performance of FL algorithms prior to the training stage. Our approach holds the potential for seamless integration with the aforementioned algorithms, and consider both feature and label distribution shifts.

Information sharing in FL. Various methods have been developed to address the challenge of distribution shifts among clients [24, 38, 68]. One approach involves the sharing of information among clients, such as the exchange of local distribution statistics [52, 69], data representations [20, 54], and prediction logits [6, 40]. Additionally, techniques leveraging global proxy datasets have been introduced to enhance FL training [11, 36]. Notably, FedMix [64] and FedBR [18] generate privacy-protected augmentation data by averaging local batches, subsequently improving the local training process. VHL [55] employs randomly initialized generative models to produce virtual data, compelling local features to closely align with those of same-class virtual data. FedFed [63] proposes a dataset distillation method, amalgamating distilled datasets into all clients' local datasets to mitigate distribution shifts. In comparison to existing approaches, Client2Vec presents several advantages: (1) the index generation process is decoupled from the FL training process, thereby avoiding any additional burden on FL training; (2) Client2Vec generates only one index vector per client, enhancing efficiency; (3) Client2Vec contributes to the whole FL training stage, encompassing client sampling, model aggregation, and local training processes.

9. Preliminaries

In this section, we present essential background information on the techniques and definitions employed in this paper to facilitate comprehension.

9.1. Domain Indexing

The Domain Generalization (DG) tasks are designed to address the cross-domain generalization problem by generating domain-invariant features. Typically, DG methods aim to establish independence between a data point's latent representation and its domain identity, represented by a one-hot vector indicating the source domain [14, 56, 67]. However, recent studies have demonstrated that utilizing a domain index, which is a real-value scalar (or vector) embedding domain semantics, as a substitute for domain identity, significantly enhances domain generalization performance [59, 61].

For example, in the work by Wang et al. [59], sleeping stage prediction models were adapted across patients with varying ages, using "age" as the domain index. This approach yielded superior performance compared to traditional models that categorized patients into groups based on age, employing discrete group IDs as domain identities.

Nevertheless, obtaining domain indices may not always be feasible in practical scenarios. To overcome this challenge, Xu et al. [62] formally defined the domain index and introduced variational domain indexing (VDI) to infer domain indices without prior knowledge. The definition of the domain index in [62] is illustrated as follows.

Definition of domain index. Consider the unsupervised domain adaptation setting involving a total of N domains, each characterized by a domain identity $k \in \mathcal{K} = [N] \triangleq \{1, \ldots, N\}$. Here, k belongs to either the source domain identity set $\mathcal{K}_s$ or the target domain identity set $\mathcal{K}_t$. Every domain k comprises D_k data points. The task involves n labeled data points $\{(\mathbf{x}i^s, y_i^s, k_i^s)\}_{i=1}^n$ originating from source domains ($k_i^s \in \mathcal{K}_s$) and m unlabeled data points $\{\mathbf{x}i^t, k_i^t\}_{i=1}^m$ from target domains ($k_i^t \in \mathcal{K}_t$). The objectives are twofold: (1) predict the labels $\{y_i^t\}_{i=1}^m$ for the target domain data, and (2) deduce global domain indices $\boldsymbol{\beta}_k \in \mathbb{R}^{B_\beta}$ for each domain and local domain indices $\mathbf{u}_i \in \mathbb{R}^{B_u}$ for each data point. It is important to note that each domain possesses a single global domain index but multiple local domain indices, with one corresponding to each data point in the domain. The data encoding generated from an encoder that takes $\mathbf{x}$ as input is represented as $\mathbf{z} \in \mathbb{R}^{B_z}$. The mutual information is denoted by $I(\cdot;\cdot)$.

Definition 9.1 (Domain Index)**.** *Given data $\mathbf{x}$ and label y, a domain-level variable $\boldsymbol{\beta}$ and a data-level variable $\mathbf{u}$ are called global and local domain indices, respectively, if there exists a data encoding $\mathbf{z}$ such that the following holds:*

- *Independence between $\boldsymbol{\beta}$ and $\mathbf{z}$: Global domain index β is independent of data encoding $\mathbf{z}$, i.e., $\boldsymbol{\beta} \perp\!\!\!\perp \mathbf{z}$, or equivalently $I(\boldsymbol{\beta}; \mathbf{z}) = 0$. This is to encourage domain-invariant data encoding $\mathbf{z}$.*
- *Information Preservation of $\mathbf{z}$: Data encoding $\mathbf{z}$, local domain index $\mathbf{u}$, and global domain index $\boldsymbol{\beta}$ preserves as much information on $\mathbf{x}$ as possible, i.e., maximizing $I(\mathbf{x}; \mathbf{u}, \boldsymbol{\beta}, \mathbf{z})$. This is to prevent $\boldsymbol{\beta}$ and $\mathbf{u}$ from collapsing to trivial solutions.*
- *Label Sensitivity of $\mathbf{z}$: The data encoding $\mathbf{z}$ should contain as much information on the label y as possible to maximize prediction power, i.e., maximizing $I(y; \mathbf{z})$ conditioned on $\mathbf{z} \perp\!\!\!\perp \boldsymbol{\beta}$. This is to make sure the previous two constraints on $\boldsymbol{\beta}$, $\mathbf{u}$, and $\mathbf{z}$ do not harm prediction performance.*

In this paper, we extend the Definition 9.1 to Definition 3.1 by incorporating both client feature index and client label index.

9.2. CLIP

CLIP [48] is a cross-modal model that establishes a connection between vision and natural language by projecting image and text embeddings onto a shared space. When presented with an image $\mathbf{I}$ and a corresponding descriptive sentence denoted as $\mathbf{T}$, the CLIP image encoder and text encoder encode the image and text into image embedding $\mathbf{D}$ and text embedding $\mathbf{L}$, respectively. Subsequently, the embeddings $\mathbf{D}$ and $\mathbf{L}$ are aligned to achieve a large cosine similarity, thereby harmonizing the vision and language embedding spaces.

10. Additional Experiment Results

10.1. Workflow of Client2Vec on Language Datasets

In Figure 6, we depict the workflow of Client2Vec on language datasets. The primary distinction between Figure 1 and Figure 6 arises from the methods employed for encoding data and labels. Specifically, for language datasets, particularly in the context of the next character prediction task, the data is encoded as "The next character of {data}", while the label is encoded as "Character {label}". In both cases, the CLIP text encoder is utilized by both the data encoder and label encoder for this task.

10.2. Experiment Settings

Dataset partition. The dataset partition follows the widely used settings in FL. In detail, we consider three datasets in this paper, and the details are listed as the follows.

- **Shakespeare:** The partition of Shakespeare dataset directly use the partition method provided by LEAF benchmark [5], and we set the fraction of data sample to 0.1, fraction of data in training set is set to 0.8, and minimum number of samples per user is set to 40.
- **CIFAR10:** We use the Latent Dirichlet Allocation (LDA) [23, 65] method with parameter $\alpha = 0.1$ to introduce label distribution shifts among clients. The dataset is partitioned into 100 clients.
- **DomainNet:** We randomly choose 50 classes from the overall 345 classes from DomainNet dataset. Sub-datasets of each domain are partitioned into 10 clients, resulting in 60 clients in total. Images are resized to 64×64.

Training details and hyper-parameters. For every dataset and algorithm, we randomly select 10% of clients in each communication round and execute a total of 100 communication rounds. We employ the SGD optimizer, with a momentum setting of 0.9 for the DomainNet dataset, and a weight decay set to 5e-5. The number of local epochs is fixed at 5, and the learning rate is set to 1e-2. The experiments are conduct on single NVIDIA 3090 GPU. The hyperparameters for our enhanced case studies are detailed below.

- **Improved client sampling.** The heat parameter τ in Eq (1) is tuned in $[0.1, 0.5, 1.0, 2.0]$.
- **Improved model aggregation.** We choose the optimal results by choosing $\gamma = [0.1, 0.5, 0.9]$, and set $\lambda_1 = 1.0$ by default in Eq (8).
- **Improved local training.** For algorithms without extra local regularization terms, such as FedAvg, FedAvgM, and FedLC, the weights assigned to $\mathcal{L}_{orth}$ and $\mathcal{L}_{dist}$ are explicitly fixed at 1.0. In contrast, for approaches incorporating additional local regularization terms, such as Moon, FedDyn, and FedIIR, the weights assigned to $\mathcal{L}_{orth}$ and $\mathcal{L}_{dist}$ are set equal to the respective values of those additional local regularization terms in the respective algorithms.

The hyper-parameters utilized for each baseline algorithms are listed below.

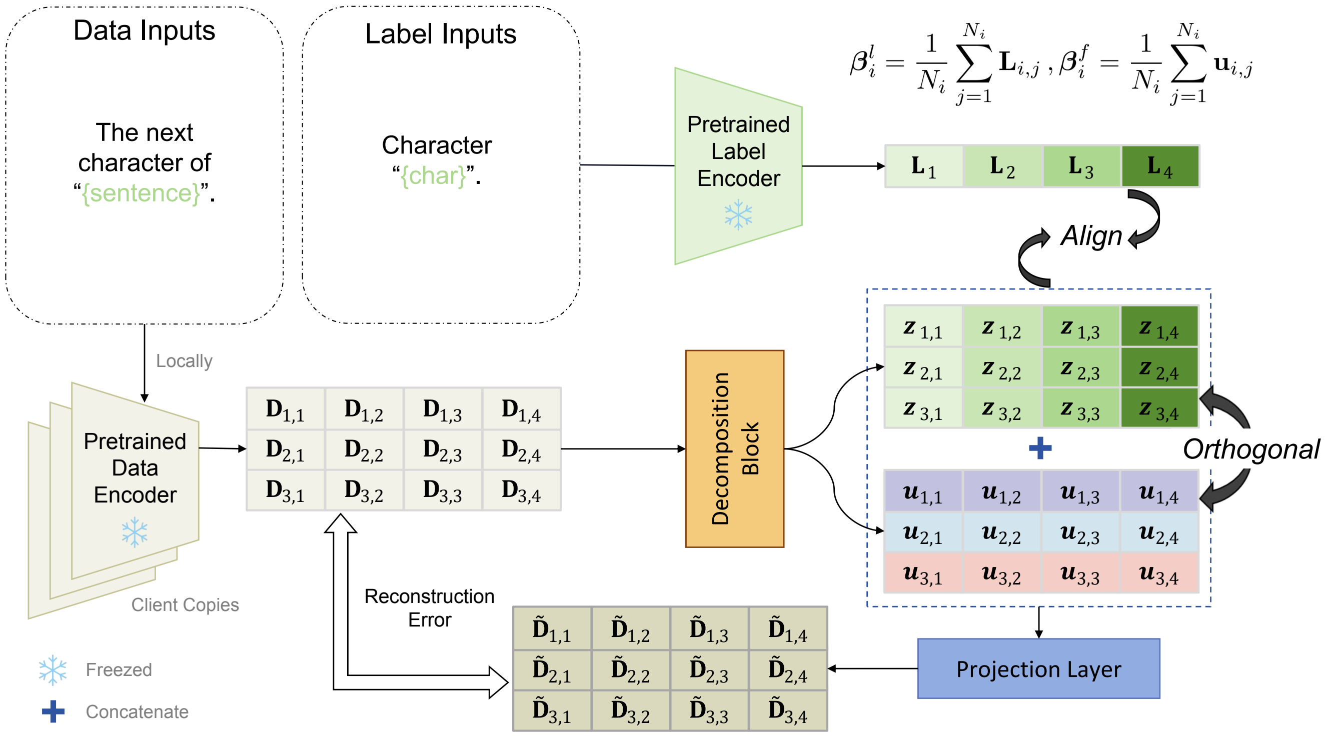

Data Inputs
The next character of "{sentence}".
Label Inputs
Character "{char}".
Pretrained Label Encoder
$\beta_i^l = \frac{1}{N_i}\sum_{j=1}^{N_i}\mathbf{L}_{i,j}, \beta_i^f = \frac{1}{N_i}\sum_{j=1}^{N_i}\mathbf{u}_{i,j}$
$\mathbf{L}_1$ $\mathbf{L}_2$ $\mathbf{L}_3$ $\mathbf{L}_4$
Align
Locally
Pretrained Data Encoder
Client Copies
$\mathbf{D}_{1,1}$ $\mathbf{D}_{1,2}$ $\mathbf{D}_{1,3}$ $\mathbf{D}_{1,4}$
$\mathbf{D}_{2,1}$ $\mathbf{D}_{2,2}$ $\mathbf{D}_{2,3}$ $\mathbf{D}_{2,4}$
$\mathbf{D}_{3,1}$ $\mathbf{D}_{3,2}$ $\mathbf{D}_{3,3}$ $\mathbf{D}_{3,4}$
Decomposition Block
$z_{1,1}$ $z_{1,2}$ $z_{1,3}$ $z_{1,4}$
$z_{2,1}$ $z_{2,2}$ $z_{2,3}$ $z_{2,4}$
$z_{3,1}$ $z_{3,2}$ $z_{3,3}$ $z_{3,4}$
+
$u_{1,1}$ $u_{1,2}$ $u_{1,3}$ $u_{1,4}$
$u_{2,1}$ $u_{2,2}$ $u_{2,3}$ $u_{2,4}$
$u_{3,1}$ $u_{3,2}$ $u_{3,3}$ $u_{3,4}$
Orthogonal
Projection Layer
Reconstruction Error
$\tilde{\mathbf{D}}_{1,1}$ $\tilde{\mathbf{D}}_{1,2}$ $\tilde{\mathbf{D}}_{1,3}$ $\tilde{\mathbf{D}}_{1,4}$
$\tilde{\mathbf{D}}_{2,1}$ $\tilde{\mathbf{D}}_{2,2}$ $\tilde{\mathbf{D}}_{2,3}$ $\tilde{\mathbf{D}}_{2,4}$
$\tilde{\mathbf{D}}_{3,1}$ $\tilde{\mathbf{D}}_{3,2}$ $\tilde{\mathbf{D}}_{3,3}$ $\tilde{\mathbf{D}}_{3,4}$
❄ Freezed
+ Concatenate

Figure 6. **Overview of the workflow of the Client2Vec on language datasets.**

- **FedAvgM:** The server momentum is tuned in $[0.1, 0.5, 1.0]$.
- **FedDyn:** We set $\alpha = 0.1$, and the max gradient norm to 10.
- **Moon:** The heat parameter is set to 0.5, and the weights of local regularization term is tuned in $[0.01, 0.1, 1.0]$.
- **FedLC:** We set $\tau = 1.0$.
- **FedIIR:** We tuned $ema = [0.95, 0.5, 0.1]$, and the weights of local regularization term are set to $1e - 3$.

Model architectures and training details of DSA-IGN. The projection layer utilizes a three-layer transformer encoder. Each transformer encoder layer consists of 8 attention heads, with the model dimension set to 32, and the feed-forward layer dimension set to 2048. The projection layer is represented as a matrix with dimensions 1024×512. Given a batch of CLIP embeddings $\mathbf{D} \in \mathbb{R}^{N \times 512}$, the input for the decomposition block is constructed as $\mathbf{I}_{i,j} = [\mathbf{D}, \mathbf{D}] \in \mathbb{R}^{N \times 1024}$. Subsequently, $\mathbf{I}$ is reshaped into $\tilde{\mathbf{I}} = (N \times 32 \times 32)$, indicating that each sample comprises 32 patches, and each patch has a dimension of 32.

The reshaped $\tilde{\mathbf{I}}$ is fed into the decomposition block, producing an output $\tilde{\mathbf{O}} \in (N \times 32 \times 32)$, which is then reshaped to $\mathbf{O} = (N \times 1024) = [\mathbf{Z}, \mathbf{U}]$. Here, $\mathbf{Z} \in \mathbb{R}^{N \times 512}$ represents the data encoding $\mathbf{z}$ as defined in Definition 3.1, and $\mathbf{U} \in \mathbb{R}^{N \times 512}$ corresponds to the sample feature index $\mathbf{u}$. The input to the projection layer is identical to the output of the decomposition block, represented as $\mathbf{O}$.

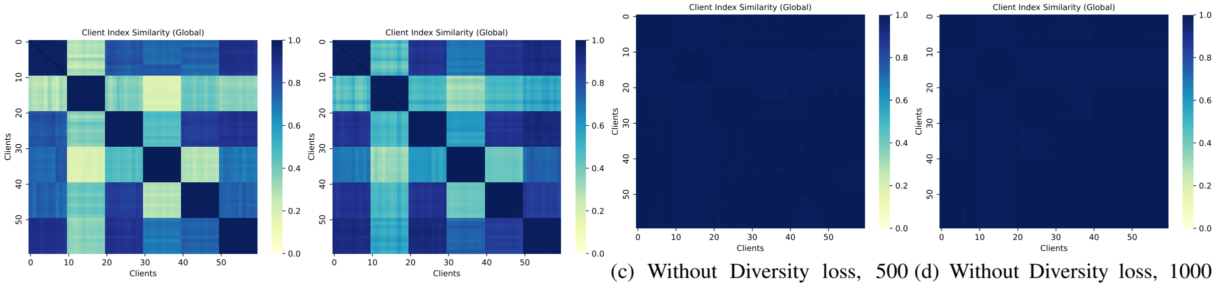
10.3. Ablation Studies on Client Index Generation

Generating client index w/o the use of the diversity loss $\mathcal{L}_{div}$. As shown in Figure 7, the client feature index β_i^f become close to identical when do not use the diversity loss. This result suggest the necessity of using the diversity loss to obtain the meaningful results.

Using different projection layers in DSA-IGN. In Figure 8, we use single Linear layer and two-layer MLP as projection layers in DSA-IGN. Results show that both architectures can obtain sufficient meaningful results.

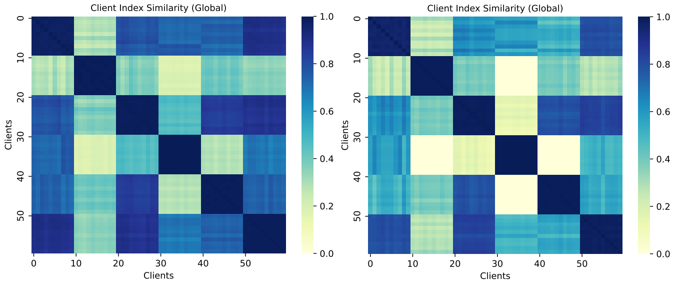
10.4. Ablation Studies on Case Studies

In Tables 5, 6, and 7, we conduct ablation studies on the three case studies we introduced in Section 4.



(a) Diversity loss, 500 epochs (b) Diversity loss, 1000 epochs (c) Without Diversity loss, 500 epochs (d) Without Diversity loss, 1000 epochs

Figure 7. **Comparison between client indexed generated with/without diversity loss.** We use the DomainNet dataset with 60 clients, and use the *Global* training strategy. The DSA-IGN is trained by 500 and 1000 global epochs. We resport the cos-similarities of the client feature index β_i^f.



(a) Linear Decoder (b) MLP Decoder

Figure 8. **Comparison between client indexed generated using different projection layers.** We use the DomainNet dataset with 60 clients, and use the *Global* training strategy. The DSA-IGN is trained by 500 global epochs. We resport the cos-similarities of the client feature index β_i^f.

Table 5. **Ablation studies on improved client sampling.** We conduct ablation studies on hyper-parameter τ in Equation (1). The term 'Original' refers to the algorithm in its initial form, where the improved client sampling is not applied. This ablation study focuses on improved client sampling, without integrating the other case studies involving enhanced model aggregation and improved local training.

CIFAR10	Original	Client2Vec (Federated)				Client2Vec (Global)			
	-	$\tau = 0.1$	$\tau = 0.5$	$\tau = 1.0$	$\tau = 2.0$	$\tau = 0.1$	$\tau = 0.5$	$\tau = 1.0$	$\tau = 2.0$
FedAvg	42.24	44.60	44.21	42.88	42.49	41.28	43.10	**45.56**	43.28
FedAvgM	42.56	45.81	44.22	43.74	43.11	42.50	44.80	**46.55**	44.62
Moon	41.12	43.86	44.28	43.23	42.82	42.15	42.80	**44.85**	44.74

Table 6. **Ablation studies on training epochs of Client2Vec.** We perform ablation studies on the training epochs of DSA-IGN, incorporating all three case studies.

CIFAR10	Original	Client2Vec (Federated)		Client2Vec (Global)	
	-	$E = 100$	$E = 500$	$E = 100$	$E = 500$
FedAvg	42.24	59.58	59.29	**61.55**	58.28
FedAvgM	42.56	61.84	63.48	61.12	**69.37**
Moon	41.12	63.61	60.26	63.79	**65.55**
FedDyn	37.22	**80.75**	69.10	78.01	70.59

Table 7. **Ablation studies on improved local training.** We conduct ablation studies on the weights of the improved local training. All three case studies are incorporated in this setting.

CIFAR10	Original	Client2Vec (Federated)			Client2Vec (Global)		
	-	1.0	5.0	10.0	1.0	5.0	10.0
FedAvg	42.24	59.29	42.76	**66.02**	48.83	58.28	34.86
FedAvgM	42.56	63.48	**70.04**	68.34	49.77	69.37	35.51
Moon	41.12	60.25	51.41	59.02	46.61	**60.53**	33.39
FedDyn	37.22	69.10	**79.96**	78.70	43.87	70.59	69.57

Table 8. **Ablation studies on only use local training.** We perform ablation studies on only using the improved local training.

CIFAR10	Original	Client2Vec (Local)		Client2Vec (Global)	
	-	Local Training Only	All Case Studies	Local Training Only	All Case Studies
FedAvg	42.24	62.22	59.29	**63.86**	58.28
FedAvgM	42.56	63.89	63.48	61.09	**69.37**
Moon	41.12	59.50	60.26	60.62	**65.55**

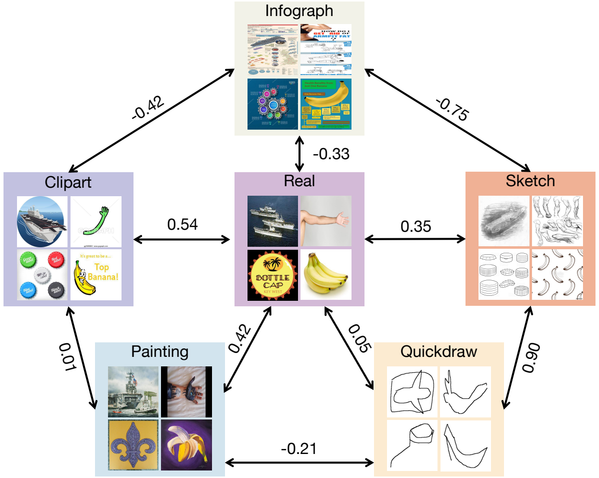


Figure 9. **Illustration of feature index similarities between different domains.** We present an analysis of cos-similarities across various domains. The results are acquired employing the GLOBAL training strategy.

10.5. Ablation Studies on Various Model Architectures.

In Table 9, we show how Client2Vec improves performance with different model architectures. Our results reveal that: (1) Client2Vec significantly boosts the performance of original algorithms in all settings, and (2) pre-trained models like MobileNet V2 and ResNet18 produce better results, while Client2Vec also enhances the performance of VIT models trained from scratch.

10.6. Ablation Studies on Level of Data Heterogeneity

In Table 10, we present the performance of Client2Vec in situations of extreme data heterogeneity, where each client possesses data from only two classes. The results indicate that Client2Vec significantly surpasses the original methods by a considerable margin.

10.7. Inter-Domain Similarity Assessment.

Utilizing the feature index β_i^f for clients, we quantify similarity across different domains. Figure 9 illustrates the average cosine similarities of client feature index β_i^f between clients belonging to different domains. The results align with human intuitions,

Table 9. **Performance of Client2Vec on various network architectures.** We evaluate the performance of Client2Vec on the DomainNet dataset using diverse network architectures. The term 'Original' refers to the initial form of the algorithms, while Client2Vec (FEDERATED) and Client2Vec (GLOBAL) applied all three case studies. Each experiment involves 100 communication rounds, with the number of local epochs set to 5. We gauge the average test accuracy of all clients in each communication round and report the highest performance achieved across all rounds. The results are averaged over three seeds. For the VIT experiments, we use the CCT-7/3x1 models [21].

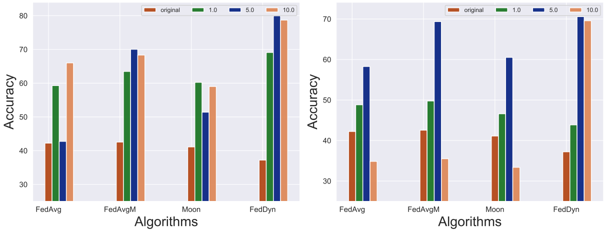
DomainNet	MobileNet V2 (Pre-Trained)			ResNet18 (Pre-Trained)			VIT (From Scratch)		
	Original	Client2Vec		Original	Client2Vec		Original	Client2Vec	
		FEDERATED	GLOBAL		FEDERATED	GLOBAL		FEDERATED	GLOBAL
FedAvg	46.31 ±1.36	56.43 ±3.08	57.43 ±0.13	56.66 ±0.50	61.27 ±0.05	60.95 ±0.09	33.09 ±0.01	33.50 ±0.20	33.86 ±0.02
FedAvgM	45.50 ±1.21	58.34 ±0.01	57.44 ±1.04	57.44 ±0.42	61.22 ±0.11	60.81 ±0.18	33.67 ±0.56	34.47 ±0.20	34.21 ±0.11
FedDyn	45.41 ±0.89	51.49 ±0.17	53.33 ±0.26	58.17 ±0.61	61.67 ±0.42	59.88 ±0.42	29.57 ±0.40	31.64 ±0.13	31.36 ±0.12
MOON	50.56 ±0.89	57.03 ±0.60	57.50 ±0.52	53.80 ±0.46	60.76 ±0.25	59.90 ±0.17	32.29 ±0.52	33.58 ±0.12	33.73 ±0.03

CIFAR10	FedAVG	FedAVG + Client2Vec	FedAvgM	FedAvgM + Client2Vec
two classes each client	21.35	66.43	18.05	63.30

Table 10. **Ablation studies on level of data heterogeneity.**

Dataset	Method	Acc	DSA_IGN TT	FL TT	Total TT
CIFAR10	FedProx	44.14	0	2446.56	2446.56
CIFAR10	FedProx + Client2Vec	74.23	140.96	2580.27	2721.23
CIFAR10	FedDisco	22.54	0	2401.61	2401.61
CIFAR10	FedDisco + Client2Vec	44.37	140.96	2426.21	2567.17
CIFAR10	Clustered Sampling	42.88	0	2064.49	2064.49
CIFAR10	Client2Vec (Sampling)	44.49	140.96	2075.61	2216.57
CIFAR10	Clustered Sampling + Client2Vec Local Training	68.80	140.96	2394.29	2535.25
Domainet	FedProx	58.38	0	30384.98	30384.98
Domainet	FedProx + Client2Vec	60.89	151.47	30336.78	30448.25

Table 11. We train DSA-IGN using the FEDERATED strategy, with 128 samples per client for 20 epochs. The acronym TT stands for Training Time. The performance of SCAFFOLD is omitted due to the model's failure to converge. FedProx and SCAFFOLD employ the FL-bench framework, while FedDisco integrates the officially implemented weight adjustment function within FL-bench and follows the original paper's recommended settings.



(a) Local FEDERATED

(b) Local textscGlobal

Figure 10. **Ablation studies on hyper-parameter of local training.**

with the "Real" domain showing greater proximity to "Clipart", "Painting", and "Sketch", while exhibiting significant differences from "Infograph" and "Quickdraw". These findings validate the effectiveness of our generated client index.

10.8. Additional baselines and overhead comparison.

As presented in Table 11, we compare Client2Vec with additional baselines, which demonstrate the following insights: (1) Client2Vec consistently improves the performance of baseline algorithms across all evaluated scenarios. (2) The additional overhead introduced by Client2Vec does not significantly increase the total training time. (3) Client2Vec outperforms the original Clustered Sampling method, highlighting the importance of an appropriate client distance measurement.

Ablation studies on hyper-parameters of improved local training. In Figure 10, ablation studies on the weight of the local regularization term (Eq (4)) were conducted. The findings suggest that: (1) Using weights of 1.0 for the FEDERATED strategy and 5.0 for the GLOBAL strategy yields favorable results for all algorithms. (2) FedDyn exhibits higher resilience to changes in the weights of the local regularization terms.