FedAFD: Multimodal Federated Learning via Adversarial Fusion and Distillation

Supplementary Material

A. Algorithm and Implementation Details

A.1. Algorithm Training Framework

FedAFD consists of two alternate steps, *i.e.*, distributed local training and centralized server update.

A.1.1. Local Training Step

Taking image clients as an example, clients update the local model w_c, consisting of the feature extractor ϕ_c, intra-modal discriminator $\mathcal{D}_c^{in}$, cross-modal discriminator $\mathcal{D}_c^{cr}$, GFF module $\mathcal{A}_c$, and local classification $\mathcal{C}_c$, by solving the following function:

$$\begin{aligned}
&(\mathcal{D}_c^{in*}, \mathcal{D}_c^{cr*}) = \arg\max_{\mathcal{D}_c^{in}, \mathcal{D}_c^{cr}} (\mathcal{L}_{adv}),\\
&(\mathcal{A}_c^*, \mathcal{C}_c^*) = \arg\min_{\mathcal{A}_c, \mathcal{C}_c} (\mathcal{L}_{task}),\\
&(\phi_c^*) = \arg\min_{\phi_c} (\mathcal{L}_{task} + \beta \cdot \mathcal{L}_{adv}),\\
&s.t. \begin{cases} \mathcal{L}_{task} = \frac{1}{|\mathcal{I}_c|} \sum_{k=1}^{|\mathcal{I}_c|} l(\widetilde{i}_c^k, y_c^k; w_c),\\ \mathcal{L}_{adv} = \frac{1}{|\mathcal{P}|} \sum_{k=1}^{|\mathcal{P}|} (\mathcal{L}_{\text{in}}^k + \mathcal{L}_{\text{cr}}^k),\\ \widetilde{i}_c^k = \mathcal{A}_c(i_c^k, i_g^k), \end{cases}
\end{aligned} \tag{1}$$

where β is a tradeoff hyperparameter. Note that, in Eq.(1), the task loss $\mathcal{L}_t$ is computed on the fused feature $\widetilde{i}_c^k$.

A.1.2. Centralized Training Step

The global model is updated in two stages. In the first stage, we optimize the global model on public data using a standard task-specific loss function:

$$w_g := w_g - \eta \nabla \mathcal{L}(\mathcal{P}; w_g^t), \tag{2}$$

where $l(\cdot)$ is task-specific loss computed on public dataset, η is the learning rate, and ∇ denotes the gradient operator.

Afterwards, we leverage the representations generated by the heterogeneous local models to further enhance the global model via ensemble distillation. The corresponding update rule for this stage is given by:

$$w_g := w_g - \eta \cdot \gamma \nabla \mathcal{L}_{kd}, \tag{3}$$

where γ is a balancing coefficient.

A.2. Network Architecture Specifications

A.2.1. Discriminator Networks in BAA

Each adversarial discriminator $\mathcal{D}$ in the Bi-level Adversarial Alignment (BAA) module is implemented as a binary classification network. Its objective is to distinguish between feature vectors originating from the client's local distribution and those from the server's global distribution.

The discriminator $\mathcal{D} : \mathbb{R}^d \to [0, 1]$ is a multilayer perceptron composed of the following sequence of layers:

1. A linear layer that maps the input feature of dimension $d = 256$ to a hidden dimension of 128.
2. A LeakyReLU activation with a negative slope of 0.2.
3. A linear layer that maps from 128 to 64 dimensions.
4. Another LeakyReLU activation with a slope of 0.2.
5. A final linear layer that projects the 64-dimensional hidden representation to a single scalar.
6. A Sigmoid activation function that squashes the output into a range [0,1], interpreted as the probability that the input feature comes from the server's global distribution.

A.2.2. Attention Network in GFF

The transformations are implemented as:

$$T_1(x) = \text{B}(\text{Conv}_2(\sigma(\text{B}(\text{Conv}_1(\text{GAP}(x)))))), \tag{4}$$

$$T_2(x) = \text{B}(\text{Conv}_2(\sigma(\text{B}(\text{Conv}_1(x))))), \tag{5}$$

where Conv denotes point-wise convolutional layers, B denotes batch normalization and GAP denotes to global average pooling.

B. Additional Theoretical Analysis

B.1. Motivation from Domain Adaptation

The design of the Bi-level Adversarial Alignment (BAA) module is theoretically motivated by the foundational theory of domain adaptation. The seminal work of Ben-David et al. [1] provides a generalization bound that quantifies the expected error of a model on a target domain. Let $L_{\mathcal{D}_S}(h)$ and $L_{\mathcal{D}_T}(h)$ be the expected errors of a hypothesis on the source and target domains, respectively. The theory states:

$$L_{\mathcal{D}_T}(h) \leq L_{\mathcal{D}_S}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda, \tag{6}$$

where $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T)$ is the $\mathcal{H}\Delta\mathcal{H}$-divergence, measuring the distribution discrepancy between the source domain $\mathcal{D}_S$ and the target domain $\mathcal{D}_T$, and λ is the error of an ideal joint hypothesis on both domains.

We treat the challenge of representation inconsistency due to modality and task heterogeneity as a generalized federated domain adaptation problem. Specifically, the feature space learned by each client is considered the source domain $\mathcal{D}_S$, while the server's representation space is the target domain $\mathcal{D}_T$. In this view, both the modality gap and

Method	Clients				Server		
	CIFAR-100	AGNEWS	Flickr30k		MS-COCO		
	acc@1	acc@1	i2t R@1	t2i R@1	i2t R@1	t2i R@1	rsum R@1
LOCAL	28.07	48.35	22.33	18.44	32.48	25.06	57.54
FedMD	22.54	48.18	19.13	15.63	33.00	25.47	58.47
FedGEMS	22.84	48.30	18.93	16.05	33.12	25.50	58.62
FedET	31.86	49.38	22.63	18.22	33.20	25.72	58.92
CreamFL	22.14	42.16	18.38	15.49	33.72	25.89	59.61
FedMKD	24.99	47.99	22.33	18.37	33.24	25.94	59.18
FedDFA	23.09	43.79	19.68	17.13	33.56	25.54	59.10
FedAFD	33.18	51.98	32.48	25.68	33.98	26.18	60.16
FedAFD(K=10)	32.91	51.40	30.63	24.09	33.76	26.12	59.88

Table 1. Performance comparison with baselines on diverse clients and the server under Non-IID settings.

Method	Upload	Download	Sum
FedMD	19.54MB	19.54MB	39.08MB
FedGEMS	19.54MB	19.54MB	39.08MB
FedET	51.80MB	51.80MB	103.60MB
CreamFL	19.54MB	19.54MB	39.08MB
FedMKD	19.54MB	19.54MB	39.08MB
FedDFA	19.54MB	19.54MB	39.08MB
FedAFD	19.54MB	49.26MB	68.80MB

Table 2. Comparison of communication cost.

Method	Time(seconds)
FedMD	959
FedGEMS	974
FedET	503
CreamFL	975
FedMKD	498
FedDFA	954
FedAFD	1034

Table 3. Average Training Time per Round.

FedMD	Ours w/o BAA	Ours
(1975.38M,14.26GB)	(947.81M,15.64GB)	(2774.19M,15.65GB)

Table 4. Training cost analysis. A cell refers to (Flops, Memory).

the task gap can be interpreted as manifestations of the distribution discrepancy $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T)$. The performance of the global model on its tasks is thus upper-bounded by the errors of the local models and the distribution discrepancy.

C. Extra Experimental Results And Details

C.1. Communication Cost Analysis

The communication cost per round in FedAFD is primarily composed of two parts: 1) Server-to-Client (Download): The server broadcasts the global encoders (ϕ_g^i, ϕ_g^t) and the global representations of the public dataset to the participating clients; 2) Client-to-Server (Upload): Clients upload their locally generated representations of the public dataset to the server. The cost is calculated based on our experimental setup where the public dataset size is $|\mathcal{P}| = 10,000$ and the feature dimension $d = 256$.

In the vanilla FedAFD framework, the primary communication overhead comes from the repeatedly transmitting the global encoders to all participating clients at every communication round. To address this bottleneck, we send the global encoder to clients only once every K communication rounds (e.g., K=10), instead of every round. This approach reduces the download cost by approximately 90%, transforming the communication complexity from $O(T \cdot M)$ to $O(\frac{T}{K} \cdot M)$ where T is the total number of rounds and M is the size of global encoders.

As shown in Table 1 and Table 2, FedAFD($K = 10$) maintains competitive performance across all tasks. On the server-side MS-COCO task, it achieves a 59.88 rsum R@1, representing only a 0.28 performance drop compared to the full FedAFD (60.16), while significantly outperforming all baseline methods. More importantly, this makes our method's communication cost comparable to or better than other representation-based methods like FedET, while providing superior performance.

C.2. Computational Cost Analysis

We evaluate the computational overhead by measuring the average training time per communication on two NVIDIA RTX3090 GPUs.

Method	Clients				Server		
	CIFAR-100	AGNEWS	Flickr30k		Flickr30k		
	acc@1	acc@1	i2t R@1	t2i R@1	i2t R@1	t2i R@1	rsum R@1
FedMD	20.63	44.37	20.93	17.75	45.50	37.32	82.82
FedGEMS	20.69	44.57	20.83	16.55	45.20	37.28	82.48
FedET	<u>32.33</u>	<u>51.03</u>	22.38	<u>18.30</u>	45.00	<u>37.86</u>	82.86
CreamFL	17.68	44.15	20.90	16.91	**46.50**	36.74	83.24
FedMKD	24.58	47.11	<u>22.65</u>	18.22	45.30	37.72	83.02
FedDFA	20.60	45.12	22.05	18.13	<u>45.80</u>	37.52	<u>83.32</u>
FedAFD	**32.58**	**51.26**	**35.38**	**29.74**	**46.50**	**37.94**	**84.44**

Table 5. Impact of Discrepancy between public and private datasets. The best and second-best results are highlighted in boldface and underlined, respectively.

β	Clients				Server		
	CIFAR-100	AGNEWS	Flickr30k		MS-COCO		
	acc@1	acc@1	i2t R@1	t2i R@1	i2t R@1	t2i R@1	rsum R@1
0	**33.56**	49.03	32.13	25.56	33.70	25.59	59.29
0.3	32.96	50.31	<u>32.38</u>	25.50	<u>33.90</u>	25.98	<u>59.88</u>
0.5	<u>33.18</u>	**51.98**	**32.48**	**25.68**	**33.98**	**26.18**	**60.16**
0.7	32.52	<u>50.69</u>	32.33	<u>25.62</u>	33.80	25.98	59.78
1.0	32.29	50.66	31.93	25.35	33.82	<u>26.03</u>	59.85

Table 6. FedAFD with varying β. The best and second-best results are highlighted in boldface and underlined, respectively.

By distributing the global encoders only once every K rounds as described above, the computational overhead on clients can also be significantly reduced. Since the global encoders remain fixed during each K-round interval, clients can cache the features generated by the global encoder for local data. During the subsequent $(K \times E - 1)$ local training epochs, these cached features can be directly reused, substantially reducing the overall computational cost.

As shown in Table 3, FedAFD takes 1034 seconds per round, which is slightly longer than most other methods, but its performance is significantly improved (Table 1). The primary sources of overhead are the client-side bi-level adversarial training and the granularity-aware feature fusion, which involve forward passes through both local and global models. Although this results in higher communication costs per round, considering the significant performance advantages of FedAFD mentioned in the main text and the fact that it requires fewer rounds to achieve the goal, the total computational budget required to reach a high-performance target is often acceptable. Reducing the cost per round is a promising direction for future research.

Compared to FedMD and our ablation without BAA (Table.4), BAA incurs only modest training overhead without additional test latency, and offers an acceptable trade-off between performance (+1.47% gain) and cost.

Client Composition	i2t R@1	t2i R@1	rsum R@1
FedAFD	33.78	26.02	59.80
Fewer Image	33.40	25.88	59.28
Fewer Text	33.38	25.95	59.33
Fewer Multimodal	32.96	25.89	58.85

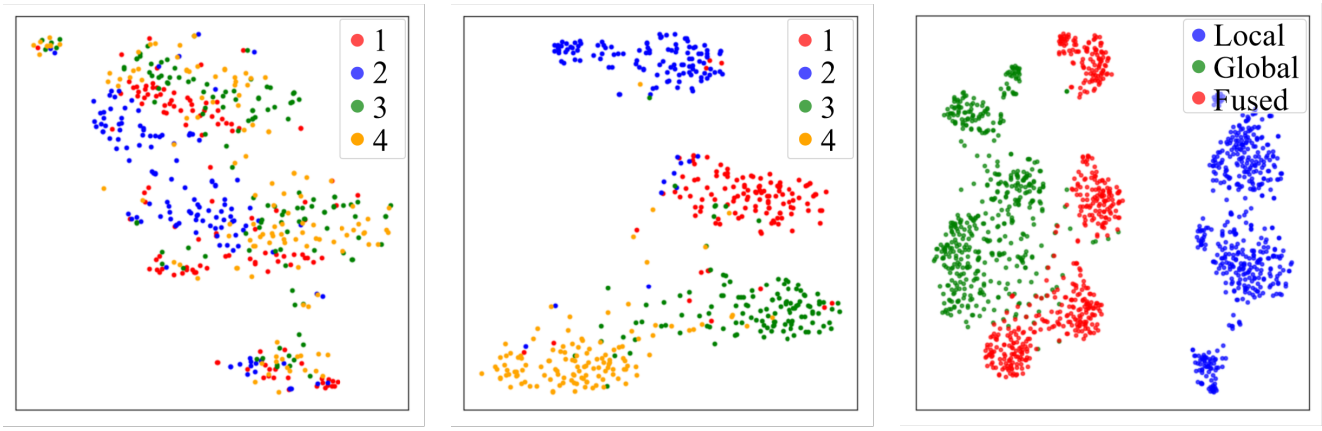
Table 7. Impact of client composition on server performance.

Method	Clients			Server
	CIFAR-100	AGNEWS	Flickr30k	MS-COCO*
	acc@1	acc@1	rsum R@1	rsum R@1
LOCAL	28.07	48.35	40.77	55.97
FedAFD	**32.03**	**50.90**	**58.27**	**58.88**

Table 8. OOD test with semantical-irrelevant subset MS-COCO*.

Method	Clients			Server
	CIFAR-100	AGNEWS	Flickr30k	MS-COCO
	acc@1	acc@1	rsum R@1	rsum R@1
Averaging	32.21	50.20	56.97	59.56
Aggr-MM	33.06	51.68	58.09	59.91
SED	**33.18**	**51.98**	**58.16**	**60.16**

Table 9. Performance under varied server aggregation methods.



(a) Locally-trained (b) Fused by Category (c) Entire spaces

Figure 1. T-SNE visualization of different features on AGNEWS, in terms of both text category and feature types.

C.3. Impact of Discrepancy between public and private datasets

A key question in federated learning with a public dataset is how does the discrepancy between the public data and the clients' private data affect performance. To investigate this, we designed a controlled experiment in which we replaced the server's public MS-COCO dataset with 10K Flickr30k samples, which are aligned with the multimodal clients' private data. This setup minimizes the domain gap between public and private multimodal datasets.

As shown in Table 5, FedAFD still achieves the best performance among all methods. There is no significant change in performance on the unimodal client compared to using MS-COCO as the public dataset. However, performance on multimodal clients and servers showed significant improvements, indicating that the smaller the difference between public and private data, the better the performance.

In a new extreme OOD setting (see Table.8), removing all client-related categories from MS-COCO, FedAFD still performs the best, demonstrating robustness across OOD levels.

C.4. Effect of Alignment Loss Weight β

We vary the alignment loss weight β in the BAA module to investigate its impact on the trade-off between representation alignment and task performance. Specifically, we evaluate $\beta \in \{0, 0.3, 0.5, 0.7, 1.0\}$ under the Non-IID setting. As shown in Table 6, moderate values (e.g., $\beta = 0.5$) yield the best performance across clients and the server, confirming that appropriate adversarial alignment helps bridge the modality and task gaps without overwhelming task learning. Overly high or low values of β adversely affect both local adaptation and global representation alignment.

C.5. Impact of Client Composition

To investigate how the composition of client modalities affects global model performance, we conduct sensitivity analysis by systematically reducing the number of clients in each modality. The original FedAFD configuration uses 3 image clients, 3 text clients, and 4 multimodal clients. We compare three reduced scenarios:
1 Fewer Image: 1 image, 3 text, 4 multimodal clients.
2) Fewer Text: 3 image, 1 text, 4 multimodal clients.
3) Fewer Multimodal: 3 image, 3 text, 2 multimodal clients.

The performance on the server's MS-COCO retrieval task is summarized in Table 7. It can be observed that reducing clients from any modality impairs server performance, and reducing multimodal clients results in a greater decrease than unimodal clients. This suggests that for the server's cross-modal retrieval tasks, multimodal clients contribute more knowledge than unimodal clients.

C.6. Visualization of GFF Module

To further validate the impact of the GFF module on client-side feature representation, we conduct a comparative t-SNE visualization analysis using the AGNEWS dataset. As shown in Fig. 1, we visualize features from the four classes under three settings: using the local encoder only ("Local"), using the global encoder ("Global"), and using the GFF-based fusion ("Fused"). Fig. 1(a) and Fig. 1(b) show class-wise distributions, while Fig. 1(c) presents an overall comparison. The results reveal that the fused features exhibit more precise inter-class boundaries than "Local", demonstrating that GFF effectively improves local feature discriminability by integrating global semantics. This confirms that FedAFD balances global knowledge integration with client-specific task optimization.

Local features, focused on specific classification tasks, capture partial information and thus appear noisier. Global

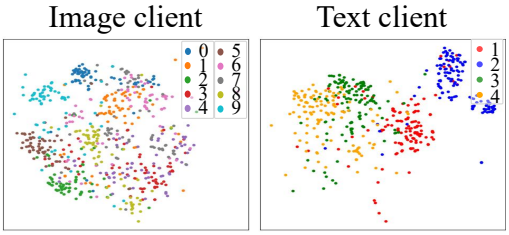


Figure 2. Per-category global features.

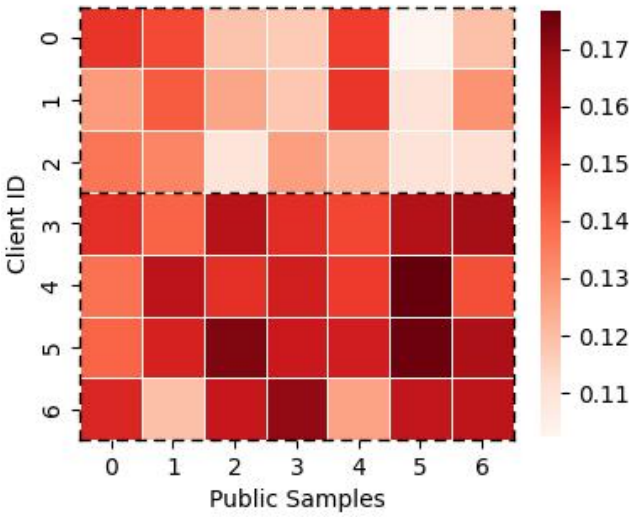


Figure 3. Aggregation weight heatmap of image clients

features, distilled from multiple clients, incorporate broader knowledge and retain local task separability (see Fig.2). Fusing them via GFF produces the most discriminative, least noisy clusters.

C.7. Ablation of SED Module

An ablation replacing SED with server aggregation using only multimodal clients ("Aggr-MM") shows that SED outperforms both naive averaging and retrieval-only aggregation (see Table.9). The heatmap Fig. 3 illustrates the final aggregation weights assigned to clients with image modality across seven random public samples. Rows Client ID 0, 1 and 2 correspond to unimodal image clients, while the rest correspond to multimodal clients. Each column represents the client-wise weight distribution for a specific public sample. It can be observed that multimodal clients generally receive higher weights compared to unimodal clients. This phenomenon can be attributed to the fact that multimodal clients can access and integrate both image and text modalities and their extracted features are naturally more aligned with the server's modality reference features. Consequently, they achieve higher similarity scores during the aggregation process. However, unimodal clients may also receive relatively high weights on certain samples, suggesting that they can also contribute to the aggregation process.

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1): 151–175, 2010. 1