Enhancing Storage and Computational Efficiency in Federated Multimodal Learning for Large-Scale Models

Zixin Zhang [1] **Fan Qi** [1] **Changsheng Xu** [2]

Abstract

The remarkable generalization of large-scale models has recently gained significant attention in multimodal research. However, deploying heterogeneous large-scale models with different modalities under Federated Learning (FL) to protect data privacy imposes tremendous challenges on clients' limited computation and storage. In this work, we propose M^{2}FEDSA to address the above issue. We realize modularized decomposition of large-scale models via Split Learning (SL) and only retain privacy-sensitive modules on clients, alleviating storage overhead. By freezing large-scale models and introducing two specialized lightweight adapters, the models can better focus on task-specific knowledge and enhance modality-specific knowledge, improving the model's adaptability to different tasks while balancing efficiency. In addition, M^{2}FEDSA further improves performance by transferring multimodal knowledge to unimodal clients at both the feature and decision levels, which leverages the complementarity of different modalities. Extensive experiments on various multimodal classification tasks validate the effectiveness of our proposed M^{2}FEDSA. The code is made available publicly at https://github.com/M2FedSA/M-2FedSA.

1. Introduction

With the swift advancement of sensing technologies, there has been an explosive growth in multimodal data. These multimodal data, such as medical diagnostic records and vehicular sensory data, are typically collected and stored in a decentralized and privacy-sensitive manner. Therefore, federated multimodal learning (MFL) has garnered widespread attention (Liu et al., 2020; Xiong et al., 2022; Che et al., 2023). MFL is centered on facilitating collaborative training of models relevant to multimodal tasks among distributed clients equipped with diverse sensors, all while ensuring the protection of data privacy.

Previous work in MFL has primarily focused on addressing the challenges of data heterogeneity and modality inconsistency (Chen & Li, 2022; Chen & Zhang, 2022; Yang et al., 2022; Yu et al., 2023a), achieving notable progress in the process. As large-scale models (Houlsby et al., 2019; Howard & Ruder, 2018; Jiang et al., 2019; Radford et al., 2021; Yuan et al., 2021; Tong et al., 2022; Jia et al., 2021; Wang et al., 2022) have increasingly demonstrated remarkable capabilities in fields such as Natural Language Processing (NLP), and Computer Vision (CV), an increasing number of efforts in MFL are beginning to harness large-scale models for extracting superior features or directly deploying such models on local clients within the FL framework. However, these approaches are subject to at least two significant shortcomings. Firstly, large-scale models usually maintain a large number of parameters, and updating all model parameters represents an impractical computational and communication cost. To address this challenge, existing work (Chen et al., 2023; Li et al., 2023; Chen et al., 2022a; Zhao, 2023; Yu et al., 2023b; Cai et al., 2022; Jiang et al., 2023) has introduced various efficient fine-tuning strategies in FL. However, considering that the size of models is snowballing now, even efficient fine-tuning strategies will still result in unavoidable storage costs. Moreover, the large-scale model has significant intellectual property value and it is questionable whether it is fully accessible to clients.

To address the above issues, we propose a unified MFL framework for large-scale models, M^{2}FEDSA. Specifically, to alleviate the challenge of limited storage resources on the client, we adopt a U-shaped split learning method (Vepakomma et al., 2018) that splits the large-scale models and only keeps privacy-sensitive modules that interact directly with the raw data and labels on the client. In addition, to balance efficiency and performance, we carefully design a **Dual Adaptive Fine-tuning Strategy (DAFS)**. We introduce task and modality adapters to focus

[1]School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China [2]Institute of Automation, Chinese Academy of Sciences, Beijing, China. Correspondence to: Fan Qi <fanqi@email.tjut.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

on task-specific and modality-specific knowledge, respectively. Compared with aggregating all model parameters, by aggregating parameters of two lightweight adapters separately, we can achieve efficient knowledge sharing within and across modalities, thus improving the efficiency and performance of the adapters. In addition, we devise a **Dual Knowledge Transfer Strategy (DKTS)** to exploit the intrinsic connections between multimodal further and facilitate cross-client multimodal knowledge complementarity. The transformer-based multimodal fusion module deployed on the main server learns semantically rich multimodal knowledge and then transfers the multimodal knowledge to clients at the feature and decision levels.

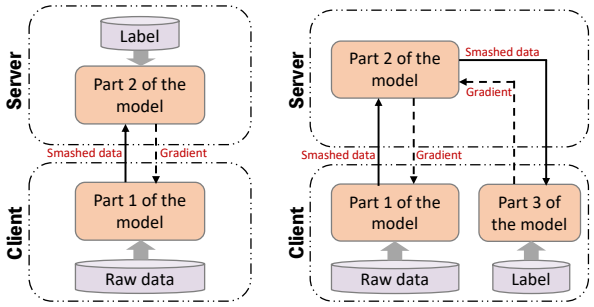
Our main contributions are summarized as follows:

- We introduce M^{2}FEDSA, a first-of-its-kind framework merging split learning with multimodal federated learning, achieving substantial reductions in computational and memory requirements, and leveraging inter-modality complementarities.
- The proposed Dual Adaptive Fine-tuning Strategy makes the model more focused on task-specific and modality-specific knowledge, balancing computational efficiency and model performance. The simplicity of this strategy renders it compatible with a broad spectrum of large-scale models.
- Our experiments on different types of multimodal tasks (i.e., action recognition (Damen et al., 2022; Monfort et al., 2019), hate speech recognition (Kiela et al., 2020), and emotion recognition (Poria et al., 2018)) demonstrate that M^{2}FEDSA significantly outperforms existing MFL methods, achieving higher efficiency with fewer trainable parameters.

2. Related Work

2.1. Multimodal Federated Learning

Existing MFL typically adopts a method of modular training and aggregation to facilitate knowledge sharing among clients (Yang et al., 2022; Zhang et al., 2023b; Chen & Zhang, 2022; Cho et al., 2022) or employs the concept of contrastive learning, using public datasets for aligning modality representations between clients (Yu et al., 2023a). In addition, some methods (Xiong et al., 2022; Feng et al., 2023) simulate a more straightforward setup, i.e., modal homogeneity between clients and perform multimodal fusion within the client. With the rise of large-scale models, some researchers have begun to exploit the large-scale models to push the upper limit of MFL further. These methods (Chen et al., 2023; Li et al., 2023) typically use efficient parameter fine-tuning techniques to empower clients to fine-tune the large-scale models. However, they overlook the substantial storage cost required by the large-scale models and the knowledge infringement issues faced by the full deployment of the models, resulting in the limited scalability of such methods in practical federated environments. Therefore, there is a considerable research gap in effectively applying large-scale models within a MFL framework.



(a) Vanilla Split Learning (b) U-shaped Split Learning

Figure 1. Illustration of the data flow in Split Learning (Vepakomma et al., 2018).

2.2. Split Learning

Split Learning (SL) (Vepakomma et al., 2018) splits the neural network into multiple parts, which will be deployed on different devices. Like FL, all raw training data are stored on the client and not transferred to the server. For the simplest SL, the model is split into two parts deployed on the client and server sides. The raw data pass through the client's submodel and become the smashed data, then are sent to the server side to complete the forward propagation. After calculating the loss function, the server starts the backward propagation, and the client waits for the gradient of the smashed data from the server to update the local submodel. The forward and backward propagation between the client and server continue until the whole model converges. The above process is illustrated in Figure 1(a). Figure 1(b) shows a variant of SL, i.e., U-shaped SL, in which the data labels are no longer shared with the server.

Split Federated Learning (SFL) (Thapa et al., 2022) is a recent collaborative training paradigm combining SL and FL's advantages. It offers significant advantages over FL regarding reduced computation and memory usage. Current works on SFL focus on unimodal settings (Thapa et al., 2022; Park et al., 2021; Tian et al., 2022; Deng et al., 2023; Tran et al., 2022), neglecting multimodal scenarios. The proposed M^{2}FEDSA innovatively applies SFL to multimodal learning, filling this study gap.

2.3. Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning technique is first proposed in NLP. The goal is to achieve efficient adaptation of large-scale models for various downstream tasks without the need

to fine-tune all the parameters, thus reducing the computational cost while achieving performance comparable to full fine-tuning. The most representative methods are adapters (Chen et al., 2022b;c; Houlsby et al., 2019; Sung et al., 2022), prompt tuning (Jia et al., 2022; Liu et al., 2023; Zhou et al., 2022a;b), low-rank adaptation (Hu et al., 2021), and their unified variants (He et al., 2021; Mao et al., 2021; Zhang et al., 2022a). Several recent works have researched efficient fine-tuning in the federated setting based on these classical methods. Zhang et al. (Zhang et al., 2023a) introduce Federated Instruction Tuning (FedIT), which utilizes FL as the framework for guiding the fine-tuning of large language models. Lu et al. (Lu et al., 2023) focus on homogeneous data with the same input and output space. PromptFL in (Guo et al., 2023) lets participants in a federated setup collaboratively learn prompts rather than entire models. Other recent works include FedLLM (Xu et al., 2023), FedAdapter (Cai et al., 2022), AUG-FedPrompt (Cai et al., 2023), FedLogic (Xing et al., 2023), and FedPrompt (Zhao et al., 2023). Similarly, the works above primarily focus on unimodal or modality homogeneity scenarios and cannot be directly applied to setups with cross-client modality heterogeneity.

3. Methodology

3.1. Problem Formulation

Under the SFL setting, we consider one main server $\mathcal{S}_{main}$, one federated server $\mathcal{S}_{fed}$, and $\mathcal{K}$ clients, assuming each client has $\mathcal{M}$ sensor devices (i.e., corresponding to $\mathcal{M}$ modalities). The clients have relatively limited computational resources and storage capacity, while the $\mathcal{S}_{main}$ has abundant computational resources and storage. The $\mathcal{S}_{fed}$ is primarily used to aggregate the updated parameters.

The client k with m devices has a private multimodal dataset $D_k = \{\{\boldsymbol{x}_{m,k}\}_{m=1}^{\mathcal{M}}, \boldsymbol{y}_k\}$. During training, all devices of each client collaborate and interact synchronously with the $\mathcal{S}_{main}$. While in the inference stage, each device can perform inference independently based on its private unimodal data. The overall objective of the proposed framework is:

$$\min_{\theta} \sum_{k=1}^{\mathcal{K}} \sum_{m=1}^{\mathcal{M}} \ell(\theta_{m,k}, D_{m,k}), \tag{1}$$

where $\theta_{m,k}$ is the complete model associated with the device m of client k. ℓ is the total loss function.

3.2. Model Architecture after Splitting

We adopt SL to exploit the benefits of large-scale models without overburdening client storage. We choose the U-shaped variant (Figure 1(b)) for its ability to retain data labels locally, bolstering privacy protection. Specifically, for the k-th client's device belonging to modality m, we define the complete model $\theta_{m,k}$ into four parts:

- the embedded module θ^{em} (on *Clients*),
- the low-level encoder module θ^{le} (on *Clients*),
- the high-level encoder module θ^{he} (on *Main Server*),
- the classifier module θ^{c} (on *Clients*).

The high-level encoder module, which has more parameters (more layers), is deployed on the $\mathcal{S}_{main}$. The other modules, which have fewer parameters (fewer layers), are deployed on the clients. The client's data are sequentially fed into the four modules, and then the loss is computed locally with the groundtruth. Subsequently, the gradient is backward propagated in the reverse direction of the above process to update the parameters of the four modules.

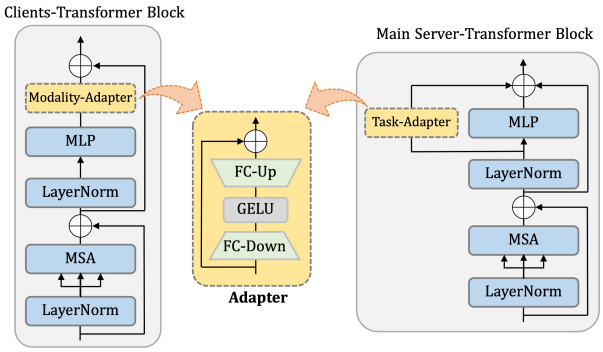


Figure 2. The proposed Modality Adapter (Serial) and Task Adapter (Residual).

3.3. Dual Adaptive Fine-tuning Strategy (DAFS)

Compared with full fine-tuning, efficient fine-tuning strategies can significantly reduce computational resource consumption and enable efficient porting of large-scale models to multimodal applications. Therefore, we propose two lightweight adapters, Task Adapter (TA) and Modality Adapter (MA). As shown in Figure 2, the adapters are bottlenecked, with the first layer downscaling to extract key features, non-linear activations added in the middle, and the third layer restoring the original dimensionality. This design is both efficient and well-integrated with frozen large-scale models. By selectively incorporating lightweight adapters into frozen large-scale models, our method enables non-intrusive task- and modality-specific enhancements while improving training efficiency without overwriting the generalizable representations.

Task Adapter θ^{ta}. In order to adapt to target tasks, we propose incorporating task adapters in the high-level encoders deployed on the $\mathcal{S}_{main}$. This design choice is motivated by the fact that task-discriminative features tend to emerge in higher-level semantic feature space rather than lower-level representations (Pan et al., 2023; Meng et al., 2022).

Taking emotion recognition as an example, compared with low-level features such as color and texture, a better understanding of high-level semantics such as scenery and prosody is much more critical for improving performance. As shown in Figure 2, we incorporate the TA in a residual connection to the MLP of the transformer layers. It generates the task-adapted representation z_l^{ta} given an immediate input z_l' by:

$$z_l^{ta} = z_l' + \theta^{ta}(\mathrm{LN}(z_l')) + \mathrm{MLP}(\mathrm{LN}(z_l')). \tag{2}$$

Modality Adapter θ^{ma}. We incorporate modality adapters into every layer of the low-level encoders deployed on edge devices to enhance the sensitivity and discriminability of distinctive information in each modality. The modality adapters re-encode the preliminary representations to accentuate semantics tightly coupled with each modality. For instance, the visual MA may amplify features for color when recognizing a lemon, while the text MA highlights sourness semantics. As shown in Figure 2, we insert the MA serially after the original MLP layer in each transformer layer. Given an immediate input z_l', it generates the modality-adapted representation z_l^{ma} by:

$$z_l^{ma} = z_l' + \theta^{ma}(\mathrm{MLP}(\mathrm{LN}(z_l'))). \tag{3}$$

3.4. Dual Knowledge Transfer Strategy (DKTS)

Multimodal fusion can make better use of the complementary information between multimodal data. Therefore, we incorporate an attention-based fusion module θ^{fu} in the $\mathcal{S}_{main}$. We use a late-fusion mechanism, which first connects the output features of the high-level encoder of different modalities $\{\ddot{\mathcal{F}}^1, ..., \ddot{\mathcal{F}}^m\}$ to obtain $\bar{\mathcal{F}}$ and then inputs it into a multi-head self-attention network to obtain the final multimodal embedding $\tilde{\mathcal{F}}$. We use two methods to knowledge transfer, complementing unimodal features $\{\ddot{\mathcal{F}}^m\}_{m=1}^{\mathcal{M}}$ with multimodal knowledge:

Feature-level Transfer: We encourage multimodal feature $\tilde{\mathcal{F}}$ and unimodal feature $\ddot{\mathcal{F}}^m$ to exhibit consistent high-level semantic information by bringing them closer together, represented as follows:

$$\ell^{ftran} = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} (1 - \cos\_\mathrm{sim}(\ddot{\mathcal{F}}^m, \tilde{\mathcal{F}})), \tag{4}$$

where $\cos\_\mathrm{sim}(\cdot,\cdot)$ denotes the calculation of the cosine similarity between two features.

Decision-level Transfer: We further refine the unimodal model's predictive capability through the adaptive distillation (Wu et al., 2022). This process strives to align the decision boundaries of the unimodal and multimodal classifiers by minimizing the discrepancy in their output predictions, represented as follows:

Algorithm 1 M^{2}FEDSA (Stage ①)

Input: Local Dataset, $D_k = \{\{\boldsymbol{x}_{m,k}\}_{m=1}^{\mathcal{M}}, \boldsymbol{y}_k\}_{k=1}^{\mathcal{K}}$; learning rate, η; the low-level encoders output, $\dot{\mathcal{F}}$; the high-level encoders output, $\ddot{\mathcal{F}}$; the number of rounds in the Stage ①, $\mathcal{T}'$.
Initialize models deployed on $\mathcal{S}_{main}$ and clients;
for each round $t = 1, 2, ..., \mathcal{T}'$ **do**
  for each client k in parallel **do**
    for each device $m \in$ client k in parallel **do**
      Send $\dot{\mathcal{F}}_{m,k}^t \leftarrow \theta_{m,k}^{le,t}(\theta_{m,k}^{em,t}(\boldsymbol{x}_{m,k}))$ to $\mathcal{S}_{main}$;
      $\ddot{\mathcal{F}}_{m,k}^t \leftarrow \theta_m^{he}(\dot{\mathcal{F}}_{m,k}^t)$;
    end for
    $\bar{\mathcal{F}}_k^t \leftarrow \mathrm{Concat}(\ddot{\mathcal{F}}_{1,k}^t, ..., \ddot{\mathcal{F}}_{m,k}^t), \tilde{\mathcal{F}}_k^t \leftarrow \theta^{fu,t}(\bar{\mathcal{F}}_k^t)$, $\hat{y}_k^{f,t} \leftarrow \theta^{fc,t}(\tilde{\mathcal{F}}_k^t)$;
    Send $\{\ddot{\mathcal{F}}_{m,k}^t\}_{m=1}^{\mathcal{M}}$, $\tilde{\mathcal{F}}_k^t$, and $\hat{y}_k^{f,t}$ to the client k;
    Compute $\ell_k^{ce,f}$ with $\{\hat{y}_k^{f,t}, \boldsymbol{y}_k\}$, then send to $\mathcal{S}_{main}$;
    for each device $m \in$ client k in parallel **do**
      $\hat{y}_{m,k}^t \leftarrow \theta_{m,k}^{c,t}(\ddot{\mathcal{F}}_{m,k}^t)$;
      Compute $\ell_{m,k}^{ce}$ with $\hat{y}_{m,k}^t$ and $\boldsymbol{y}_k$;
      Update $\theta_{m,k}^c$ with $\ell_{m,k}^{ce}$, get $\theta_{m,k}^{c,t+1}$;
      Send $d\ddot{\mathcal{F}}_{m,k}^t := \nabla \ell_{m,k}^{ce}(\theta_{m,k}^{c,t})$ to $\mathcal{S}_{main}$;
    end for
    Update θ^{fu} and θ^{fc} with $\ell_k^{ce,f}$, get $\theta_k^{fu,t+1}$ and $\theta_k^{fc,t+1}$, update θ_m^{ta} with $d\ddot{\mathcal{F}}_{m,k}^t$, get $\theta_{m,k}^{ta,t+1}$;
    Send $\{d\dot{\mathcal{F}}_{m,k}^t := \nabla d\ddot{\mathcal{F}}_{m,k}^t(\theta_m^{ta,t})\}_{m=1}^{\mathcal{M}}$ to client k, $\{\theta_k^{fu,t+1}, \theta_k^{fc,t+1}, \{\theta_{m,k}^{ta,t+1}\}_{m=1}^{\mathcal{M}}\}_{k=1}^{\mathcal{K}}$ to $\mathcal{S}_{fed}$;
    for each device $m \in$ client k in parallel **do**
      Update $\theta_{m,k}^{ma,t}$ with $d\dot{\mathcal{F}}_{m,k}^t$, get $\theta_{m,k}^{ma,t+1}$;
      Send $\theta_{m,k}^{ma,t+1}$ and $\theta_{m,k}^{c,t+1}$ to $\mathcal{S}_{fed}$;
    end for
  end for
  Average aggregation of $\theta_k^{fu,t+1}$ and $\theta_k^{fc,t+1}$ for k clients, get $\theta^{fu,t+1}$ and $\theta^{fc,t+1}$;
  Intra-modality aggregation $\{\{\theta_{m,k}^{ta,t+1}\}_{m=1}^{\mathcal{M}}\}_{k=1}^{\mathcal{K}}$, get $\theta^{ta,t+1}$;
  Inter-modality aggregation $\theta_{m,k}^{ma,t+1}$ and $\theta_{m,k}^{c,t+1}$, get $\theta^{ma,t+1}$ and $\theta^{c,t+1}$;
  Send the models of round $t+1$ to the clients and $\mathcal{S}_{main}$.
end for

$$\ell^{dtran} = \frac{\mathrm{KL}(\hat{y}^f \,||\, \hat{y})}{\ell^{ce,f} + \ell^{ce}}, \tag{5}$$

where $\hat{y}$ and $\hat{y}^f$ represent the logit features obtained by the unimodal classifier θ^c on the client and multimodal classifier θ^{fc} on $\mathcal{S}_{main}$, respectively. $\ell^{ce,f}$ and ℓ^{ce} represent the cross-entropy loss calculated using multimodal and unimodal features, respectively.

Algorithm 2 M^{2}FEDSA (Stage ②)

Input: Same as Algorithm 1
for each round $t = \mathcal{T}' + 1, \mathcal{T}' + 2, ..., \mathcal{T}$ **do**
  for each client k in parallel **do**
    for each device $m \in$ client k in parallel **do**
      See lines 6-7 of Algorithm 1;
    end for
    See lines 9-11 of Algorithm 1;
    for each device $\mathbf{m} \in$ client $\mathbf{k}$ in parallel **do**
      See line 13 of Algorithm 1;
      Compute $\ell_{m,k}$ in Equation 1;
      `/* Replace` $\ell^{ce}_{m,k}$ `in Algorithm 1 with` $\ell_{m,k}$ `*/`
      See lines 15-16 of Algorithm 1;
    end for
    Update θ^{ta}_m with $d\ddot{\mathcal{F}}^t_{m,k}$, get $\theta^{ta,t+1}_{m,k}$;
    Send $\{d\dot{\mathcal{F}}^t_{m,k} := \nabla d\ddot{\mathcal{F}}^t_{m,k}(\theta^{ta,t}_m)\}_{m=1}^{\mathcal{M}}$ to client k, $\{\{\theta^{ta,t+1}_{m,k}\}_{m=1}^{\mathcal{M}}\}_{k=1}^{\mathcal{K}}$ to $\mathcal{S}_{fed}$;
    for each device $m \in$ client k in parallel **do**
      See lines 21-22 of Algorithm 1;
    end for
  end for
  See lines 26-28 of Algorithm 1;
end for

3.5. Federated Optimization and Inference

We divide the M^{2}FEDSA's training into two stages. In Stage ①, the parameters of the multimodal fusion module θ^{fu} and multimodal classifier module θ^{fc} are randomly initialized; directly transferring knowledge from the unoptimized multimodal fused features would impair the semantic integrity of the unimodal representations, resulting in fluctuations or even decreases in model performance. Therefore, we use separate optimization goals (i.e., ℓ^{ce} and $\ell^{ce,f}$) for the unimodal clients and multimodal fusion module to realize decoupled training between them in Stage ①.

In Stage ②, we freeze the multimodal fusion module θ^{fu} and multimodal classifier θ^{fc} while supplementing multimodal knowledge to unimodal features through feature- and decision-level transfers. At this stage, the local optimization goal for each client's device is as follows:

$$\ell = \lambda_0(\ell^{ce} + \ell^{ce,f}) + \lambda_1 \ell^{ftran} + \lambda_2 \ell^{dtran}, \quad (6)$$

where λ_0, λ_1, and λ_2 denote the trade-off hyperparameters. Throughout the training process, the $\mathcal{S}_{fed}$ performs intra-modality average aggregation for MAs and classifiers deployed on clients and inter-modality average aggregation for TAs deployed on the $\mathcal{S}_{main}$ after each round of communication.

We next describe the termination criteria for the Stage ① of training. To determine whether the performance of the multimodal fusion module has reached stability, we retain the cross-entropy loss values of the multimodal classification feature from the last ten communication rounds and calculate the coefficient of variation ζ of these values. When the ζ is below the threshold γ, the training switches from the Stage ① to the Stage ②. Specifically, we first calculate the absolute difference δ between two consecutive rounds of multimodal classification losses as follows:

$$\delta_t = |\ell^{ce,f}_t - \ell^{ce,f}_{t-1}|, \quad (7)$$

where $\ell^{ce,f}_t$ denotes the multimodal classification loss at the t-th round. The coefficient of variation ζ serves as an indicator of the relative change within the sequence. It is defined as the ratio of the standard deviation $\sigma(\delta)$ to the mean $\mu(\delta)$ of these differences:

$$\mu(\delta) = \frac{\sum_{i=1}^{\mathrm{N}} \delta_i}{\mathrm{N}}, \sigma(\delta) = \sqrt{\frac{\sum_{i=1}^{\mathrm{N}} (\delta_i - \mu(\delta))^2}{\mathrm{N}}}, \quad (8)$$

$$\zeta = \frac{\sigma(\delta)}{\mu(\delta)}. \quad (9)$$

We set $\mathrm{N} = 10$ in our experiments. The ζ lower than the threshold γ indicates that the change in loss has been minimal, i.e., the performance of the multimodal fusion module has reached stability.

Inference. In the test time, each sample x belonging to modality m on client k will be classified according to the following formula:

$$\hat{y}_k = \theta^c_{m,k}(\theta^{he}_{m,k}(\theta^{le}_{m,k}(\theta^{em}_{m,k}(\boldsymbol{x}_{m,k})))). \quad (10)$$

We detail the two stages of the training process in Algorithm 1 and Algorithm 2, respectively[1].

Table 1. Statistics for four multimodal datasets.

DATASET	CLIENTS	MODALITY	INSTANCES	CLASSES	TASK
Epic-Kitchens-100	50/100	$\{V, A\}$	89K+	97	MAR
MiT	50/100	$\{V, A\}$	1M	339	MAR
Hateful-Memes	100/200	$\{V, T\}$	10K	2	HSR
MELD	15/30	$\{V, A, T\}$	17K+	7	MER

4. Experiment

4.1. Experimental setup

Datasets. We evaluate the proposed M^{2}FEDSA using four datasets from different tasks. (1) **Epic-Kitchens-100** (Damen et al., 2022) is an egocentric video dataset for Multimodal Action Recognition (MAR). It has 97 action classes

[1]We use orange text for steps on $\mathcal{S}_{main}$, blue text for steps on $\mathcal{S}_{fed}$, and the rest for steps on the $\mathcal{K}$ clients.

Table 2. Comparison to state-of-the-art on four datasets. "TPs." represents the average number of trainable parameters per client. **Bold number**: The best result in each column.

TYPE	METHOD	EPIC-KITCHENS (ACC↑)				MIT (ACC↑)				HATEFUL-MEMES (AUC↑)				MELD (UAR↑)				
		V	*A*	Avg.	TPs.	*V*	*A*	Avg.	TPs.	*V*	*T*	Avg.	TPs.	*V*	*T*	*A*	Avg.	TPs.
Model Homo.	FedAvg	28.74	37.12	32.93±0.43	74.79K	34.49	36.15	35.32±0.28	74.79K	54.88	56.16	55.52±0.87	1.54K	50.75	48.92	46.55	48.74±0.37	5.99K
	FedProx	29.21	39.97	34.59±0.25	74.79K	38.39	36.59	37.49±0.33	74.79K	54.09	59.83	56.96±0.53	1.54K	51.82	54.72	47.63	51.39±0.21	5.99K
	SCAFFOLD	28.93	40.36	34.65±0.16	74.79K	38.21	38.01	38.11±0.21	74.79K	56.21	58.61	57.41±0.42	1.54K	52.89	53.73	44.52	50.38±0.11	5.99K
	FedRS	28.69	39.75	34.22±0.26	74.79K	39.14	37.44	38.29±0.11	74.79K	56.49	59.21	57.85±0.68	1.54K	53.43	50.80	48.86	51.03±0.16	5.99K
	FedOpt	29.92	41.34	35.63±0.50	74.79K	39.86	39.62	39.74±0.24	74.79K	56.52	60.14	58.61±0.34	1.54K	54.87	52.75	51.69	48.92±0.08	5.99K
Model Hetero.	FedMSplit	31.39	40.66	36.03±0.31	1.36M	42.59	38.75	40.67±0.19	1.36M	60.21	63.15	61.68±0.49	1.58M	55.00	53.57	50.70	53.09±0.19	1.05M
	FDARN	32.48	40.85	36.67±0.18	6.11M	43.02	39.68	41.35±0.40	6.11M	59.98	62.88	61.43±0.51	6.43M	54.36	53.69	50.89	52.98±0.25	6.09M
	CreamFL	33.71	**45.63**	39.67±0.07	26.04M	43.75	40.21	41.98±0.22	26.04M	60.14	**64.56**	62.35±0.28	60.85M	54.12	51.54	-	52.83±0.23	63.16M
	M^{2}FedSA(ours)	**35.45**	45.57	**40.51±0.23**	0.12M	**45.16**	**42.12**	**43.64±0.31**	0.12M	**63.21**	64.27	**63.74±0.36**	62.49K	**56.22**	**55.48**	**51.29**	**54.33±0.12**	69.79K

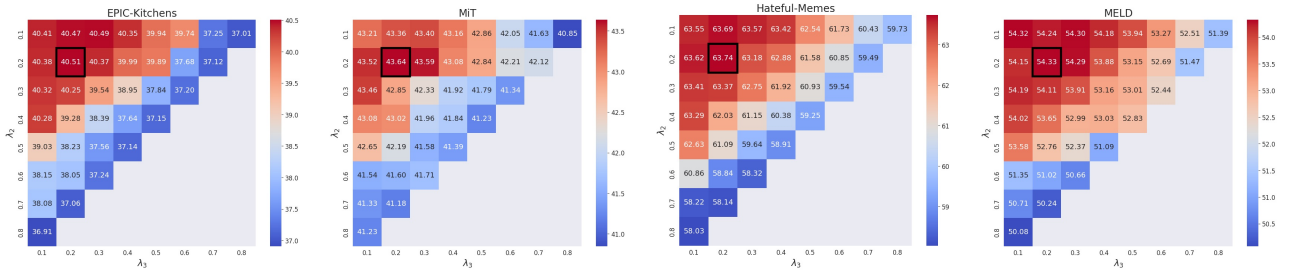
Table 3. Ablation studies on the proposed Task Adapter and Modality Adapter. Yellow background: the baseline performance used per column; Red background: negative transfer; Green background: positive transfer. "DP" indicates the use of differential privacy (Dwork, 2006).

METHOD	MODEL SIZE	EPIC-KITCHENS (ACC↑)			MIT (ACC↑)			HATEFUL-MEMES (AUC↑)			MELD (UAR↑)		
		TPs.(%)	Avg.	Δ	TPs.(%)	Avg.	Δ	TPs.(%)	Avg.	Δ	TPs.(%)	Avg.	Δ
Fine-Tune		0.0	37.03	0.0	0.0	40.33	0.0	0.0	60.54	0.0	0.0	52.02	0.0
Linear Probe	**Setup I** :	-99.769	33.82	-3.21	-99.769	39.75	-3.21	-99.999	54.27	-6.27	-99.994	47.56	-4.46
M^{2}FedSA(TA)	Swin-T 2D/3D (Tiny)	-99.446	39.54	+2.51	-99.446	42.62	+2.51	-99.907	62.81	+2.27	-99.870	52.73	+0.71
M^{2}FedSA(MA)	RoBERTa (Base)	-99.607	39.01	+1.98	-99.607	41.82	+1.98	-99.945	62.12	+1.58	-99.911	52.14	+0.12
M^{2}FedSA(TA+MA)	Whisper (Tiny)	-98.714	39.88	+2.85	-98.714	43.11	+2.85	-99.855	63.19	+2.65	-99.785	53.62	+1.60
M^{2}FedSA(TA+MA)+DP		-98.714	36.97	-0.06	-98.714	40.78	-0.06	-99.855	60.35	-0.19	-99.785	50.99	-1.03
Fine-Tune		+152.015	38.67	+1.64	+152.015	41.03	+1.64	+197.145	61.33	+0.79	+161.343	52.29	+0.27
Linear Probe	**Setup II** :	-98.251	34.15	-2.88	-98.251	40.27	-2.88	-99.158	56.42	-4.12	-99.235	48.31	-3.71
M^{2}FedSA(TA)	Swin-T 2D/3D (Base)	-98.036	40.13	+3.10	-98.036	43.19	+3.10	-99.033	63.27	+2.73	-98.738	53.74	+1.72
M^{2}FedSA(MA)	RoBERTa (Large)	-98.193	39.68	+2.65	-98.193	42.56	+2.65	-99.099	62.59	+2.05	-98.924	53.82	+1.80
M^{2}FedSA(TA+MA)	Whisper (Base)	-96.552	**40.51**	+3.48	-96.552	**43.64**	+3.48	-98.975	**63.74**	+3.20	-98.055	**54.33**	+2.31
M^{2}FedSA(TA+MA)+DP		-96.552	37.25	+0.22	-96.552	41.94	+0.22	-98.975	61.18	+0.64	-98.055	52.95	+0.93

and involves 89,979 video segments. Each video segment contains audio of the participant's narration of the current behavior. To assess our method under varying scales of distributed settings, we respectively scatter each modality onto 25 and 50 clients, forming a total of 50 and 100 clients for federated simulation. (2) **MiT** (Monfort et al., 2019) is a large-scale MAR dataset (1M) with short (3 seconds) videos with overall list of 339 action labels. Its data partitioning strategy is the same as Epic-Kitchens-100. (3) **Hateful Memes** (Kiela et al., 2020) is a multimodal dataset compiled for Hate Speech Recognition (HSR) in meme images. It contains over 10,000 meme images with textual captions, manually labeled as hateful or benign content. Similarly, we partition the image and text modalities across 50 and 100 clients, forming 100 and 200 clients for exploring federated hate speech detection. (4) **MELD** (Poria et al., 2018) is a Multimodal Emotion Recognition (MER) dataset with text, audio, and visual modalities from dialog videos. It contains over 17,000 utterances labeled with emotions like *joy*, *surprise*, etc. We distribute each modality over 5 and 10 clients, forming 15 and 30 clients for federated simulation. See Table 1 for details of the four datasets [2].

Implementation details. In FL, networks with a Transformers architecture achieve higher similarity and synchronization efficiency among the participants (Gao et al., 2023). Therefore, for the three modalities involved in our experiments (i.e., visual, text, and audio), we use Swin-Transformer2D/3D (Liu et al., 2021; 2022), RoBERTa (Liu et al., 2019), and Whisper (Radford et al., 2023) as backbones, respectively. The overall framework of our proposed is implemented with Pytorch (Paszke et al., 2019). We use eight NVIDIA Tesla V100 PCIe GPUs for training the framework. The balance weights λ_0, λ_1, and λ_2 of the loss function in Equation 6 are set to 0.6, 0.2, and 0.2. We set the threshold $\gamma = 0.3$ in our experiments. The multimodal fusion module deployed on the main server side is implemented as a multi-head attention network, where we set

[2]Unless otherwise noted, the results for the four datasets in the main text are obtained under the $\mathcal{K}$=50/50/100/15 setting. The results for the more challenging client partitioning setting, i.e., $\mathcal{K}$=100/100/200/30, are shown in the appendix.



(a) EPIC-Kitchens-100 (b) MiT (c) Hateful-Memes (d) MELD

Figure 3. Effect of λ_1, λ_2, and λ_3 for total loss on the three datasets. Note: $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

Table 4. Ablation studies on the proposed knowledge transfer strategies.

METHOD	EPIC-KITCHENS	MIT	HATEFUL-MEMES	MELD
$w/o\ \ell_{ft}$	40.12	42.89	63.07	53.95
$w/o\ \ell_{dt}$	39.98	40.77	62.76	53.06
$w/o\ (\ell_{ft}, \ell_{dt})$	38.54	40.09	60.93	52.24
M^{2}FedSA	**40.51**	**43.64**	**63.74**	**54.33**

the head to 8 and the hidden dimension to 512. Additionally, the multimodal classifier is implemented as two-layer perceptrons with the activation function of GELU, where the dimension of the hidden layer is set to 256. In the initial training phase (Stage ①), we specifically tailored the learning rates for the multimodal fusion module and the classifier, setting them to 1e-2, 1e-3, and 1e-3 for the three datasets. For unimodal classifiers deployed on the client side, we use a one-layer perceptron that projects the feature dimensions to a specific number of classification categories. More implementation details are listed in the appendix.

Baselines. We consider two types of baselines: 1) *Model Homogeneity*, including FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), SCAFFOLD (Karimireddy et al., 2020), FedRS (Li & Zhan, 2021), and FedOpt (Reddi et al., 2020). 2) *Model Heterogeneity*, including FedMSplit (Chen & Zhang, 2022), FDARN (Yang et al., 2022), and CreamFL (Yu et al., 2023a). For the model homogeneity methods, we use the same backbone network as M^{2}FEDSA for feature extraction. Subsequently, a MLP network of identical architecture is deployed across clients for each modality. For the model heterogeneity methods, we follow its model setup. We run experiments with three different random seeds. See the appendix for more detailed results.

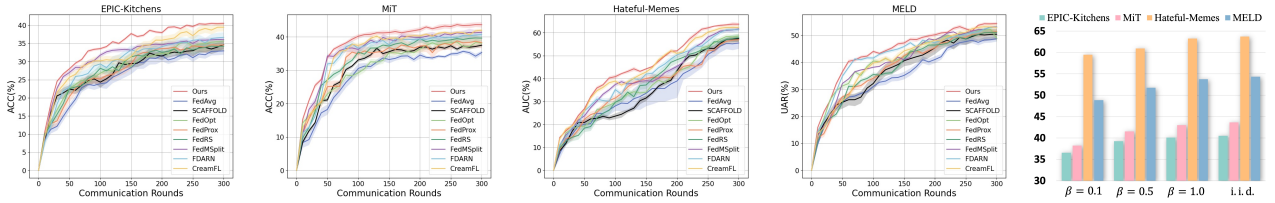
4.2. Comparisons to the State of the Art

Table 2 shows the main results of all methods on the four datasets. Our method achieves an overall performance improvement of 0.84%, 1.66%, 1.39%, and 1.24% over the second-best method in these four datasets. Methods with homogeneous model (FedAvg, FedProx, Scaffold, FedRS, FedOpt) generally perform poorly, as there are substantial differences in feature representation between modalities despite consistent model structure. Directly aggregating model parameters across modalities can lead to confusion. M^{2}FEDSA also achieves superior performance compared with modal heterogeneity MFL methods, which demonstrates the effectiveness of our DAFS and DKTS, highlighting the potential and benefits of the large-scale models. It also supports our view on the importance of adequate knowledge complementarity between modalities.

In addition, our method achieves a better balance between performance and efficiency according to the statistics of trainable parameters in Table 2. It eliminates the need for an additional feature extraction step. Consequently, there is no requirement to store extra feature models locally, significantly reducing storage overhead. Compared with other model heterogeneity methods, our method does not introduce additional networks, thereby avoiding increased complexity. Yet, it achieves superior performance with fewer trainable parameters. For instance, our method improves the accuracy by 4.48% compared with FedMSplit for the EPIC-Kitchens dataset, and the average trainable parameters per client are reduced by 91.18%.

4.3. Ablation Study

To demonstrate the effectiveness of our proposed two adapters, we perform comprehensive ablation studies and present the results in Table 3. Firstly, we compare our method with both full fine-tuning and linear probing. Full fine-tuning is slightly better than linear probing but is explosive for trainable parameters. Our method reduces the trainable parameters by 98%+ on average and exceeds the performance of full fine-tuning. Based on the results, it can be found that both proposed adapters improve the performance of fine-tuning, and the TA has a more significant impact on the performance improvement. Furthermore, as previous work shows (Qu et al., 2022), the larger-scale



(a) (b) (c) (d) (e)

Figure 4. (a)-(d): The effect of communication rounds on the performance of the four datasets. (e): The effect of different data heterogeneity on model performance.

Table 5. Comparison results on the different settings (position, connection type, and width) of the proposed adapters on four datasets. "R" denotes the residual connection type and "S" denotes the serial connection type.

DATASET	Position (TA on the $\mathcal{S}_{main}$)				Connection Type (TA / MA)				Width (TA & MA)				
	Top-1	Top-3	Bottom-1	Bottom-3	R / R	R / S	S / R	S / S	8	16	32	64	128
EPIC-Kitchens (ACC↑)	40.24	40.51	37.38	37.49	40.22	40.51	39.57	39.12	39.16	39.77	40.03	40.51	40.48
MiT (ACC↑)	43.49	43.64	40.53	40.81	42.98	43.64	42.10	41.66	42.37	42.79	43.31	43.64	43.68
Hateful-Memes (AUC↑)	63.15	63.74	61.92	62.16	63.19	63.74	62.33	61.85	60.61	62.45	63.74	63.62	63.75
MELD (UAR↑)	53.89	54.33	50.75	51.27	53.94	54.33	53.79	53.11	52.29	53.18	53.85	54.33	54.37

model has a more robust performance. We also provide results for the version using Differential Privacy (Dwork, 2006), and despite the added noise, our method still achieves competitive results.

Table 4 shows the effectiveness of the proposed knowledge transfer strategies. Both feature-level and decision-level transfers significantly enhance overall performance, indicating their indispensable roles in improving model performance. Feature-level transfer deepens the understanding of data, while decision-level transfer optimizes task decisions. The absence of either would lead to a performance decrement. Therefore, the comprehensive utilization of multimodal information is crucial for achieving optimal performance.

4.4. Hyperparametric Analysis

We validate the choice of $\lambda_1 = 0.6$, $\lambda_2 = 0.2$, and $\lambda_3 = 0.2$ as illustrated in Figure 3. The axes of the figure represent the values of λ_2 and λ_3, respectively. We observe an increment in the evaluation metrics across all four datasets when λ_2 and λ_3 improve from 0.1 to 0.2. However, as the values of λ_2 and λ_3 exceed 0.2, a decline in the evaluation metrics ensues, suggesting that the combination of ℓ^{ce}, ℓ^{ftran} and ℓ^{dtran} has reached optimal performance. Furthermore, an overemphasis on knowledge transfer coefficients λ_2 and λ_3 results in performance inferior to scenarios where cross-entropy loss operates in isolation. This phenomenon underscores the detrimental impact of excessive reliance on knowledge transfer mechanisms, potentially leading to an overfitting scenario or the dilution of individual loss component effectiveness. It highlights the necessity for a balanced approach in integrating multiple objectives to avoid overshadowing the primary task objective, thereby ensuring a synergistic effect.

4.5. Further Analysis

Number of Communication Rounds. Figure 4(a)-4(d) show the average test result across all clients for different communication rounds. With fewer rounds (e.g., less than 30 on the Epic-Kitchens), our framework performs similarly to the baselines. Thanks to the proposed DAFS and DKTS, our method consistently outperforms all baselines after more rounds of training (e.g., about 70 on the Epic-Kitchens).

Degree of Heterogeneity of Data Distribution. We investigate the performance of M^{2}FEDSA under different degrees of data heterogeneity. Following (Zhang et al., 2022b), we use Dirichlet distribution $D(\beta)$ to simulate the data heterogeneity of clients. As shown in previous work (Qu et al., 2022; Yang et al., 2023), large-scale models are inherently more generalizable and can ameliorate the data heterogeneity problem well. As shown in Figure 4(e), M^{2}FEDSA still exhibits competitive performance even with highly heterogeneous settings such as $\beta = 0.1$.

Position of Task Adapters. We study the effect of adding Task Adapters in different layers. As shown in Table 5, adding TA to the bottom layer of the high-level encoders (Bottom-1) yields the worst performance compared with other settings. We hypothesize that the shallow layers learn

Table 6. Comparison of the storage size (MB) and the communication time (s) with the baselines. "Mem_{TP}": the storage required for trainable parameters, "Mem_{FP}": the storage required for frozen parameters, "$\text{Mem}_{\text{total}}$": the total storage required, "Time": the communication time required by the client during the testing stage. The results in table are averaged across all clients.

METHOD	EPIC-KITCHENS / MIT				HATEFUL-MEMES				MELD			
	Mem_{TP}	Mem_{FP}	$\text{Mem}_{\text{total}}$	Time	Mem_{TP}	Mem_{FP}	$\text{Mem}_{\text{total}}$	Time	Mem_{TP}	Mem_{FP}	$\text{Mem}_{\text{total}}$	Time
FedAvg	0.3	309	309.3	0.45	0.01	832	832.01	0.18	0.02	649	649.02	0.25
FedProx	0.3	309	309.3	0.72	0.01	832	832.01	0.44	0.02	649	649.02	0.61
SCAFFOLD	0.3	309	309.3	0.63	0.01	832	832.01	0.37	0.02	649	649.02	0.55
FedRS	0.3	309	309.3	1.41	0.01	832	832.01	0.92	0.02	649	649.02	1.07
FedOpt	0.3	309	309.3	1.65	0.01	832	832.01	1.16	0.02	649	649.02	1.34
FedMSplit	5	309	314	3.73	6	832	838	2.76	4	649	653	3.24
FDARN	23	309	332	5.39	25	832	857	3.02	23	649	672	4.75
CreamFL	99	-	99	4.88	232	-	232	2.95	241	-	241	4.14
M^2FedSA(ours)	0.5	11	11.5	7.59	0.2	121	121.2	5.97	0.3	66	66.3	6.31

generic representations that do not need much adaptation, while deeper layers learn task-specific features. In contrast, TAs in the top three layers (Top-3) performed similarly to their placement in only the top layer (Top-1), suggesting that Top-1 could be a more resource-efficient option in limited-resource settings.

Connection types of Adapters. As shown in Table 5, the combination of residual TAs and serial MAs achieves optimal performance. The residual TA adds task-specific enhancements without disturbing the original feature information, thereby preserving the model's original generalization ability while fully utilizing information specific to the task. Meanwhile, through serial layer-by-layer processing, the MA effectively strengthens the capture of modality-specific knowledge, significantly enhancing the model's performance in processing specific modality data.

Width of Adapters. We study how the width of the adapters affects the final performance. The results in Table 5 show that wider adapters tend to achieve better performance but will introduce more tunable parameters. For the four datasets, the width of the adapters gradually stabilizes beyond 64, 64, 32, and 64. It is worth noting that the adapters with a width of only eight still achieve a performance of 39.16% on EPIC-Kitchens, which is competitive with the other methods in Table 2.

Cost Analysis. As illustrated in Table 6, our methodology eliminates the need for prior feature extraction, enabling end-to-end training. We have notably reduced the storage burden on local clients by employing SL. Compared to the CreamFL method, which has the second-lowest storage cost, our approach achieves a remarkable reduction in local storage requirements by 88.4%, 88.4%, 47.7%, and 72.5% across the four datasets, respectively.

Furthermore, despite the necessity of additional communication with the main server, our method remains time-efficient. This efficiency is mainly because the model is almost frozen except for the lightweight adapters and classifiers. Consequently, compared with conventional MFL methods, our method's communication time is acceptable. This balance of communication efficiency and reduced client-side storage underscores the practicality and innovation of our FL framework, particularly in multimodal contexts.

5. Conclusion & Limitation

In this paper, we investigate MFL based on large-scale models and introduce a novel framework, M^2FEDSA, which empowers resource-constrained clients to train large-scale models using their data. The proposed dual adaptive fine-tuning strategy and dual knowledge transfer strategy efficiently leverage the complementarity of multimodal data, balancing efficiency with performance. We achieve comparable or even better performance than prior arts on four benchmarks. However, a limitation of the framework is that M^2FEDSA relies on complete multimodal data during the training fine-tuning phase, a requirement that could pose constraints in practical applications.

Acknowledgements

This work was supported by NSFC (No.62206200, 62206137, 62036012, 62376196, U23A20387), and Tianjin Natural Science Foundation (No.22JCQNJC00940, 22JCYBJC00030).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Cai, D., Wu, Y., Wang, S., Lin, F. X., and Xu, M. Fedadapter: Efficient federated learning for modern nlp. *arXiv preprint arXiv:2205.10162*, 2022.

Cai, D., Wu, Y., Yuan, H., Wang, S., Lin, F. X., and Xu, M. Towards practical few-shot federated nlp. In *Proceedings of the 3rd Workshop on Machine Learning and Systems*, pp. 42–48, 2023.

Che, L., Wang, J., Zhou, Y., and Ma, F. Multimodal federated learning: A survey. *Sensors*, 23(15):6986, 2023.

Chen, H., Zhang, Y., Krompass, D., Gu, J., and Tresp, V. Feddat: An approach for foundation model finetuning in multi-modal heterogeneous federated learning. *arXiv preprint arXiv:2308.12305*, 2023.

Chen, J. and Zhang, A. Fedmsplit: Correlation-adaptive federated multi-task learning across multimodal split networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 87–96, 2022.

Chen, J., Xu, W., Guo, S., Wang, J., Zhang, J., and Wang, H. Fedtune: A deep dive into efficient federated fine-tuning with pre-trained transformers. *arXiv preprint arXiv:2211.08025*, 2022a.

Chen, S. and Li, B. Towards optimal multi-modal federated learning on non-iid data with hierarchical gradient blending. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 1469–1478. IEEE, 2022.

Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., and Luo, P. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022b.

Chen, Z., Duan, Y., Wang, W., He, J., Lu, T., Dai, J., and Qiao, Y. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022c.

Cho, Y. J., Manoel, A., Joshi, G., Sim, R., and Dimitriadis, D. Heterogeneous ensemble knowledge transfer for training large models in federated learning. *arXiv preprint arXiv:2204.12703*, 2022.

Damen, D., Doughty, H., Farinella, G. M., Furnari, A., Kazakos, E., Ma, J., Moltisanti, D., Munro, J., Perrett, T., Price, W., et al. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision*, pp. 1–23, 2022.

Deng, R., Du, X., Lu, Z., Duan, Q., Huang, S.-C., and Wu, J. Hsfl: Efficient and privacy-preserving offloading for split and federated learning in iot services. In *2023 IEEE International Conference on Web Services (ICWS)*, pp. 658–668. IEEE, 2023.

Dwork, C. Differential privacy. In *International colloquium on automata, languages, and programming*, pp. 1–12. Springer, 2006.

Feng, T., Bose, D., Zhang, T., Hebbar, R., Ramakrishna, A., Gupta, R., Zhang, M., Avestimehr, S., and Narayanan, S. Fedmultimodal: A benchmark for multimodal federated learning. *arXiv preprint arXiv:2306.09486*, 2023.

Gao, Y., Sun, H., Li, Z., and Yu, H. The prospect of enhancing large-scale heterogeneous federated learning with transformers. *arXiv preprint arXiv:2308.03945*, 2023.

Guo, T., Guo, S., Wang, J., Tang, X., and Xu, W. Promptfl: Let federated participants cooperatively learn prompts instead of models-federated learning in age of foundation model. *IEEE Transactions on Mobile Computing*, 2023.

He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., and Neubig, G. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pp. 2790–2799. PMLR, 2019.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pp. 4904–4916. PMLR, 2021.

Jia, M., Tang, L., Chen, B.-C., Cardie, C., Belongie, S., Hariharan, B., and Lim, S.-N. Visual prompt tuning. In *European Conference on Computer Vision (ECCV)*, 2022.

Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Zhao, T. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*, 2019.

Jiang, J., Liu, X., and Fan, C. Low-parameter federated learning with large language models. *arXiv preprint arXiv:2307.13896*, 2023.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Kiela, D., Firooz, H., Mohan, A., Goswami, V., Singh, A., Ringshia, P., and Testuggine, D. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in neural information processing systems*, 33: 2611–2624, 2020.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Li, X.-C. and Zhan, D.-C. Fedrs: Federated learning with restricted softmax for label distribution non-iid data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 995–1005, 2021.

Li, Z., Hou, Z., Liu, H., Wang, Y., Li, T., Xie, L., Shi, C., Yang, C., Zhang, W., and Liu, Z. Federated learning in big model era: Domain-specific multimodal large models. *arXiv preprint arXiv:2308.11217*, 2023.

Liu, F., Wu, X., Ge, S., Fan, W., and Zou, Y. Federated learning for vision-and-language grounding problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 11572–11579, 2020.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., and Hu, H. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3202–3211, 2022.

Lu, W., Hu, X., Wang, J., and Xie, X. Fedclip: Fast generalization and personalization for clip in federated learning. *arXiv preprint arXiv:2302.13485*, 2023.

Mao, Y., Mathias, L., Hou, R., Almahairi, A., Ma, H., Han, J., Yih, W.-t., and Khabsa, M. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv preprint arXiv:2110.07577*, 2021.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Meng, Q., Zhou, F., Ren, H., Feng, T., Liu, G., and Lin, Y. Improving federated learning face recognition via privacy-agnostic clusters. *arXiv preprint arXiv:2201.12467*, 2022.

Monfort, M., Andonian, A., Zhou, B., Ramakrishnan, K., Bargal, S. A., Yan, T., Brown, L., Fan, Q., Gutfreund, D., Vondrick, C., et al. Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):502–508, 2019.

Pan, M.-H., Xin, H.-Y., Xia, C.-Q., and Shen, H.-B. Few-shot classification with task-adaptive semantic feature learning. *Pattern Recognition*, 141:109594, 2023.

Park, S., Kim, G., Kim, J., Kim, B., and Ye, J. C. Federated split vision transformer for covid-19 cxr diagnosis using task-agnostic training. *arXiv preprint arXiv:2111.01338*, 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Poria, S., Hazarika, D., Majumder, N., Naik, G., Cambria, E., and Mihalcea, R. Meld: A multimodal multi-party dataset for emotion recognition in conversations. *arXiv preprint arXiv:1810.02508*, 2018.

Qu, L., Zhou, Y., Liang, P. P., Xia, Y., Wang, F., Adeli, E., Fei-Fei, L., and Rubin, D. Rethinking architecture design for tackling data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10061–10071, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pp. 28492–28518. PMLR, 2023.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Sung, Y.-L., Cho, J., and Bansal, M. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5227–5237, 2022.

Thapa, C., Arachchige, P. C. M., Camtepe, S., and Sun, L. Splitfed: When federated learning meets split learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8485–8493, 2022.

Tian, Y., Wan, Y., Lyu, L., Yao, D., Jin, H., and Sun, L. Fedbert: When federated learning meets pre-training. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4):1–26, 2022.

Tong, Z., Song, Y., Wang, J., and Wang, L. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

Tran, N.-P., Dao, N.-N., Nguyen, T.-V., and Cho, S. Privacy-preserving learning models for communication: A tutorial on advanced split learning. In *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1059–1064. IEEE, 2022.

Vepakomma, P., Gupta, O., Swedish, T., and Raskar, R. Split learning for health: Distributed deep learning without sharing raw patient data, 2018.

Wang, W., Bao, H., Dong, L., Bjorck, J., Peng, Z., Liu, Q., Aggarwal, K., Mohammed, O. K., Singhal, S., Som, S., et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022.

Wu, C., Wu, F., Lyu, L., Huang, Y., and Xie, X. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1):2032, 2022.

Xing, P., Lu, S., and Yu, H. Fedlogic: Interpretable federated multi-domain chain-of-thought prompt selection for large language models. *arXiv preprint arXiv:2308.15324*, 2023.

Xiong, B., Yang, X., Qi, F., and Xu, C. A unified framework for multi-modal federated learning. *Neurocomputing*, 480:110–118, 2022.

Xu, M., Wu, Y., Cai, D., Li, X., and Wang, S. Federated fine-tuning of billion-sized language models across mobile devices. *arXiv preprint arXiv:2308.13894*, 2023.

Yang, T., Zhu, Y., Xie, Y., Zhang, A., Chen, C., and Li, M. Aim: Adapting image models for efficient video action recognition. *arXiv preprint arXiv:2302.03024*, 2023.

Yang, X., Xiong, B., Huang, Y., and Xu, C. Cross-modal federated human activity recognition via modality-agnostic and modality-specific representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 3063–3071, 2022.

Yu, Q., Liu, Y., Wang, Y., Xu, K., and Liu, J. Multimodal federated learning via contrastive representation ensemble. *arXiv preprint arXiv:2302.08888*, 2023a.

Yu, S., Muñoz, J. P., and Jannesari, A. Federated foundation models: Privacy-preserving and collaborative learning for large models. *arXiv preprint arXiv:2305.11414*, 2023b.

Yuan, L., Chen, D., Chen, Y.-L., Codella, N., Dai, X., Gao, J., Hu, H., Huang, X., Li, B., Li, C., et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

Zhang, J., Vahidian, S., Kuo, M., Li, C., Zhang, R., Wang, G., and Chen, Y. Towards building the federated gpt: Federated instruction tuning. *arXiv preprint arXiv:2305.05644*, 2023a.

Zhang, R., Chi, X., Liu, G., Zhang, W., Du, Y., and Wang, F. Unimodal training-multimodal prediction: Cross-modal federated learning with hierarchical aggregation. *arXiv preprint arXiv:2303.15486*, 2023b.

Zhang, Y., Zhou, K., and Liu, Z. Neural prompt search. *arXiv preprint arXiv:2206.04673*, 2022a.

Zhang, Z., Yang, Y., Dai, Y., Qu, L., and Xu, Z. When federated learning meets pre-trained language models' parameter-efficient tuning methods. *arXiv preprint arXiv:2212.10025*, 2022b.

Zhao, H., Du, W., Li, F., Li, P., and Liu, G. Fedprompt: Communication-efficient and privacy-preserving prompt tuning in federated learning. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Zhao, J. Privacy-preserving fine-tuning of artificial intelligence (ai) foundation models with federated learning, differential privacy, offsite tuning, and parameter-efficient fine-tuning (peft). 2023.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16816–16825, 2022a.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022b.

A. Appendix

In this appendix, we first elaborate on additional implementation details (see Appendix B). Subsequently, we present supplementary experimental results (see Appendix C). Finally, we show how to add modality and task adapters to the original ViT block (see Appendix D).

B. Implementation Details

B.1. EPIC-Kitchens-100 / MiT

We process multimodal data encompassing video and audio modalities for the EPIC-Kitchens-100 and MiT datasets. We implement a resolution standardization for video data, resizing the raw frame to a uniform 224×224 pixels. In addition, we extract a consistent set of 16 frames from each video, ensuring standardized dimensions in the data representation. Regarding the audio modality, we employ Mel-Frequency Cepstral Coefficients (MFCC) to extract pivotal audio features, effectively capturing the audio data's essence.

The architecture of our model uses Swin-Transformer 3D and Whisper as the backbone networks for video and audio modalities, respectively. The embedding module and the first transformer block are strategically deployed on the client side, with the primary intent of ensuring data privacy by keeping modules that directly interact with raw data localized. Subsequent transformer blocks of the encoder are established on the main server, handling more complex computations. This architecture is further augmented by serial adding the Modality Adapter within the transformer block on the client side. Concurrently, the Task Adapters are residual added within the selected transformer blocks on the main server side. Each adapter maintains a standardized width of 64. The first FC layer in Adapters is randomly initialized, and the second FC layer is initialized to zero. In this way, the adapted model is close to the pre-trained model at the beginning of training.

The entire framework is trained using the AdamW optimizer for 300 communication rounds with a batch size of 16. The base learning rate is 1e-5, and the weight decay is 5e-2.

B.2. Hateful-Memes

For the Hateful-Memes dataset, we use both image and text data modalities. For image data processing, we adopt a uniform resizing method, standardizing all images to a resolution of 224×224 pixels, ensuring consistency in visual data representation. In the realm of text modality, we utilize the RobertaTokenizer for token extraction. Moreover, we normalize the token lengths to align the maximum found within the dataset, incorporating zero-padding for shorter sequences to maintain data representation uniformity.

Our model's architecture uses Swin-Transformer 2D and RoBERTa as the backbone networks for image and text modalities, respectively. The configuration and deployment of the model, along with the settings for the adapters, are in line with the methodologies applied to the EPIC-Kitchens dataset.

Table 7. Comparison to state-of-the-art on three datasets. "TPs." represents the average number of trainable parameters per client. **Bold number**: The best result in each column. Note: The results in the table are obtained under $\mathcal{K} = 50/50/100/15$.

Model Size	Type	Method	EPIC-Kitchens		MiT		Hateful-Memes		MELD	
			ACC	TPs.	ACC	TPs.	AUC	TPs.	UAR	TPs.
Setup I	Model Homogeneity	FedAvg	32.69	55.97K	34.76	55.97K	54.79	1.23K	47.07	4.48K
		FedProx	34.14	55.97K	36.21	55.97K	56.44	1.23K	48.12	4.48K
		SCAFFOLD	34.08	55.97K	36.89	55.97K	56.82	1.23K	47.39	4.48K
		FedRS	34.15	55.97K	37.33	55.97K	57.01	1.23K	48.24	4.48K
		FedOpt	34.05	55.97K	39.08	55.97K	57.18	1.23K	48.37	4.48K
	Model Heterogeneity	FedMSplit	35.71	0.69M	39.97	0.69M	60.95	0.79M	52.05	0.66M
		FDARN	35.25	5.58M	40.83	5.58M	60.51	5.78M	52.43	5.52M
		CreamFL	39.19	25.35M	41.45	25.35M	62.18	7.2M	52.51	17.8M
		M^2FedSA(ours)	**39.88**	0.11M	**43.17**	0.11M	**63.19**	59.49K	**53.62**	65.19K

Table 8. Comparison to state-of-the-art on three datasets. "TPs." represents the average number of trainable parameters per client. **Bold number**: The best result in each column. Note: The results in the table are obtained under $\mathcal{K} = 100/100/200/30$.

Model Size	Type	Method	EPIC-Kitchens		MiT		Hateful-Memes		MELD	
			ACC	TPs.	ACC	TPs.	AUC	TPs.	UAR	TPs.
Setup I	Model Homogeneity	FedAvg	30.26	55.97K	31.58	55.97K	51.96	1.23K	45.82	4.48K
		FedProx	31.82	55.97K	32.79	55.97K	53.28	1.23K	47.25	4.48K
		SCAFFOLD	30.89	55.97K	33.14	55.97K	54.18	1.23K	46.66	4.48K
		FedRS	31.97	55.97K	33.26	55.97K	54.62	1.23K	47.13	4.48K
		FedOpt	32.47	55.97K	34.53	55.97K	55.89	1.23K	48.05	4.48K
	Model Heterogeneity	FedMSplit	35.09	0.69M	35.24	0.69M	58.61	0.79M	50.23	0.66M
		FDARN	34.93	5.58M	35.82	5.58M	58.37	5.78M	50.72	5.52M
		CreamFL	35.35	25.35M	37.09	25.35M	59.26	7.2M	50.39	17.8
		M^2FedSA(ours)	**36.77**	0.11M	**39.17**	0.11M	**61.03**	59.49K	**52.19**	65.19K
Setup II	Model Homogeneity	FedAvg	31.29	74.79K	32.37	74.79K	53.18	1.54K	46.93	5.99K
		FedProx	32.14	74.79K	33.26	74.79K	54.94	1.54K	47.42	5.99K
		SCAFFOLD	31.95	74.79K	33.61	74.79K	54.63	1.54K	47.25	5.99K
		FedRS	32.08	74.79K	34.14	74.79K	55.22	1.54K	47.87	5.99K
		FedOpt	32.51	74.79K	35.38	74.79K	56.45	1.54K	48.11	5.99K
	Model Heterogeneity	FedMSplit	35.19	1.36M	36.02	1.36M	59.17	1.58M	51.17	1.05M
		FDARN	35.24	6.11M	36.29	6.11M	59.63	6.43M	51.68	6.09M
		CreamFL	36.28	26.04M	37.43	26.04M	60.55	60.85M	51.95	63.16M
		M^2FedSA(ours)	**37.25**	0.12M	**39.95**	0.12M	**61.94**	62.49K	**53.28**	69.79K

The entire framework is trained using the AdamW optimizer for 300 communication rounds with a batch size of 32. The base learning rate is 1e-4, and the weight decay is 5e-2.

B.3. MELD

We use three modalities for the MELD dataset: video, audio, and text. The preprocessing steps for each modality are consistent with the methods mentioned in our previous detailed descriptions of the Hateful-Memes and the EPIC-Kitchens datasets. To this end, we incorporate Swin-Transformer 3D, RoBERTa, and Whisper as the backbone networks for video, text, and audio modalities, respectively. Regarding model deployment and adapter configuration, we ensure consistency with the methods adopted for the abovementioned datasets.

The entire framework is trained using the AdamW optimizer for 300 communication rounds with a batch size of 32. The base learning rate is 1e-4, and the weight decay is 5e-2.

C. Additional Results

Table 7 shows a comparison of our M^2FEDSA with other multimodal federated learning methods, where we set the total number of clients to $\mathcal{K} = 50/50/100/15$ and employ the Type I modal size setup, i.e., a relatively small model size. Table 8 shows similar comparison results while the total number of clients is expanded to $\mathcal{K} = 100/100/200/30$.

The results in Table 7 and Table 8 suggest a notable trend: utilizing larger model sizes tends to enhance performance metrics. This observation underscores the great potential of larger large-scale models in improving overall system performance. The proposed M^2FEDSA, characterized by its simplicity and broad applicability, is particularly well-suited for future integration with more advanced large-scale models, potentially leading to substantial performance enhancements. Additionally, it is evident that larger-scale models demonstrate superior generalizability. It is particularly apparent in the model homogeneity methods (FedAvg, FedProx, SCAFFOLD, FedRS, and FedOpt), which performance is overly dependent on the quality of the features extracted by the large-scale model. The results in Table 7 and Table 8 corroborate this assertion, with further improvements in the performance of these methods as the model size increases.

Algorithm 3 Pseudo-code of an adapted ViT block

```
class ViTBlockWithModalityAdapter(nn.Module):

    def __init__(self, dim, num_heads, mlp_ratio, m_adapter):
        # Layers in the original ViT block
        self.mha = MultiHeadAttention(dim, num_heads)
        self.ffn = FeedForwardNetwork(dim, int(dim * mlp_ratio))

        # Modality Adapter
        self.adapter = m_adapter

    def forward(self, x):
        x = self.mha(x)

        # modality adaption (serial)
        x = x + self.adapter(self.ffn(x))

        return x

class ViTBlockWithTaskAdapter(nn.Module):

    def __init__(self, dim, num_heads, mlp_ratio, t_adapter):
        # Layers in the original ViT block
        self.mha = MultiHeadAttention(dim, num_heads)
        self.ffn = FeedForwardNetwork(dim, int(dim * mlp_ratio))

        # Task Adapter
        self.adapter = t_adapter

    def forward(self, x):
        x = self.mha(x)

        # task adaption (residual)
        x = x + self.ffn(x) + self.adapter(x)

        return x
```

Furthermore, as shown in Table 8, the growth in the number of clients leads to severe performance degradation for all methods. However, the proposed M^{2}FEDSA consistently outperforms existing multimodal federated learning algorithms, even under more challenging and complex training settings.

D. Pseudo-code of the ViT Block with Adapters

In Algorithm 3, we show the PyTorch style pseudo-code on how to add the Task Adapter and the Modality Adapter to a ViT block.