arXiv:2602.07081v1 [cs.MM] 6 Feb 2026

# 基于异构且不完整的多模态客户端数据进行联邦提示调优

Thu Hang Phung [1] Duong M. Nguyen [1] Thanh Trung Huynh [2] Quoc Viet Hung Nguyen [3]Trong Nghia Hoang [4*] Phi Le Nguyen [1*1] 人工智能创新与社会影响研究所, 河内科技大学[2] 洛桑联邦理工学院 [3] 格里菲斯大学 [4] 华盛顿州立大学

## 摘要

本文提出了一种通用的联邦提示调优框架，适用于本地数据集为多模态、且输入层面缺失特征的分布模式各不相同的实际场景。所提出的框架弥合了联邦学习与多模态提示调优之间的差距；传统上，这两者分别侧重于单模态数据或集中式数据。在此场景中，一个关键挑战是：不同客户端上编码相似缺失数据分布模式的提示指令之间缺乏语义对齐。为此，我们的框架设计了专门的客户端调优和服务器聚合机制，同时对各客户端及数据模态上的提示调优指令进行优化、对齐和聚合。这使提示指令能够相互补充并得到有效组合。在多个多模态基准数据集上进行的大量评估表明，我们的方法始终优于最先进水平（*SOTA*）的基线方法。

## 1. 引言

近期，超大型模型的兴起 [3] 得益于在海量通用数据上进行预训练，这些模型规模庞大且用途广泛，进而推动了机器学习（ML）范式的转变。如今，大多数特定任务模型都是通过微调这些大型模型得到的，而非从头开始训练。例如，解决特定自然语言处理任务的大多数最先进水平（SOTA）模型，都是基于现有的大型预训练模型或基础模型进行微调而来，例如 BERT [12], GPT [5], 和 T5 [40]。同样，各类计算机视觉任务的最先进水平模型也源自基础模型，例如在 ImageNet [13] 等大型数据集上预训练的视觉 Transformer [11]。然而，标准的微调方法假设数据是集中式的，而在实际场景中，这并非总是可行，因为微调用数据分散在多个私有孤岛式系统中。

为克服这种数据孤岛挑战，近期研究开始采用联邦微调，在运行各自（本地）微调流程的私有设备之间聚合微调后的更新 [9, 32, 47]。这种方法还可以与一种现有的参数高效调优方法——提示调优 [25]相结合。该方法着重于创建轻量级信号，例如在预训练模型输入前添加额外词元 [41, 44, 59, 60], , 为模型提供有价值的上下文信息，使其能够更有效地执行特定任务。随后，可以通过联邦学习中的传统参数聚合流程来聚合本地提示集合 [16, 29, 47, 50, 58]。

然而，这类联邦提示调优技术仅限于使用单模态数据进行学习。在更实际的场景中，传感器和设备制造技术的进步，日益推动多种数据模态（例如文本、图像）的采集，从而为微调提供更丰富的数据集 [6]。利用这种多样化的多模态信息，可以显著提升模型性能，CLIP [39] 和 ViLT [21]等多模态基础模型的成功便证明了这一点。这也进一步推动了多模态联邦学习 [8, 33, 37, 52] 以及预训练多模态大语言模型微调的研究，即多模态联邦微调 [56]。

尽管取得了这些进展，多模态联邦学习研究往往采用特定的本地架构，而这些架构无法通过对现有预训练多模态基础模型进行微调来学习 [8, 33, 37, 52]。另一方面，现有的多模态联邦微调研究假设所有本地数据集都具有相同的模态集合 [56]。因此，这些方法并不适用于不同客户端可能使用不同类型的传感器采集数据的情形。此外，各客户端的多模态传感器数据甚至可能具有不同的数据条目缺失模式。这导致本地数据集同时存在内部异质性与客户端间异质性两者。内部异质性出现在单个数据集内的各个样本

*通讯作者

对不同特征子集存在缺失条目时。当不同客户端对其数据缺失模式具有不同分布时，便会产生客户端间异质性。

提示调优领域也有近期研究，会针对每一类模态缺失子集学习一个特定提示 [24]。具有相同模态缺失集合的输入样本随后在训练期间用同一个（可学习）提示进行增强，使该提示专门用于补足相应的模态缺失。然而，该方法仅限于集中式数据设置（即不存在客户端间异质性），并且不能很好地泛化到联邦设置；在联邦设置中，不同的本地提示集合可能会偏向不同的缺失数据分布模式。因此，对这些提示进行朴素聚合，可能会使它们坍缩为信息量更低的提示，从而忽略各客户端之间缺失数据模式的异质性，导致性能较差（见图1）。

为弥合多模态提示调优与联邦学习之间的这一差距，我们提出一种多模态联邦提示调优方法，为每个客户端优化并聚合两组不同的（本地）提示：客户端间提示和客户端内提示。客户端间提示分解并编码缺失输入数据的典型分布模式，这些模式可以在各客户端之间对齐并聚合，以表示共享的分布信息。相比之下，客户端内提示编码的信息专门用于支持特定的本地模态缺失子集，而与每个客户端缺失数据的分布模式无关。这通过以下贡献实现：

**1.** 我们提出一种本地输入自适应微调算法，可识别并更新对每个样本最相关的客户端间和客户端内提示子集。我们将此过程表述为提示检索任务，重点学习映射函数，使输入在嵌入空间中靠近相关提示，同时远离不相关提示。这有助于将本地微调知识分解并提炼到提示集的不同部分，避免针对不同微调模式学习到的指令过度集中在同一组提示上（第 3.2 节）。

**2.** 我们开发了一种服务器端模型，用于聚合并对齐各客户端的客户端间提示。两者均.....
.....成与对齐均被纳入聚类任务的范畴，在该任务中，将编码了相似输入级缺失数据分布模式的客户端间提示识别出来，并合并为更全面、通用性更强的提示指令。通过这些组合提示降低本地训练损失的实证效果，被用作学习反馈，以优化该聚类模型（第 3.3 节）。

**3.** 我们通过在两个基准数据集 MM-IMDB [1] 和 UPMC Food- 上开展广泛实验，评估并展示了所提出方法的有效性。

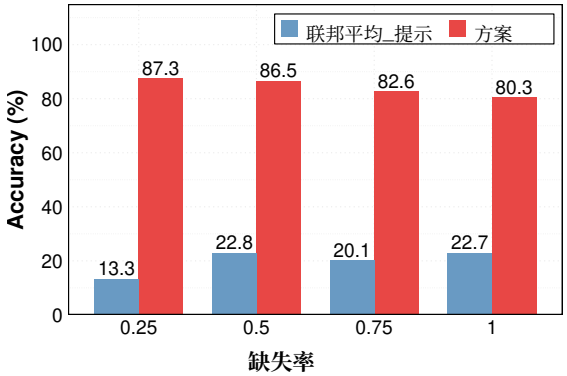

图 1. 在不同数据缺失率的设置下，我们采用提示对齐的方法优于 Fe-dAvg 提示调优（未进行对齐）。

101 [46], 将其性能与多种现有的多模态联邦学习方法以及针对联邦设置重新设计的提示调优基线进行比较。结果表明，在各种模态缺失场景下，我们的方法始终显著优于基线方法，并在多模态联邦学习领域取得了新的最先进水平（第 4节）。

图示概览了我们提出的框架——面向异构且不完整的多模态客户端数据的联邦提示调优（FED-PRIME），如图2所示。

## 2. 相关工作

**多模态联邦学习**。联邦学习（FL） [17, 31, 53, 54] 是一种协作学习范式，使多个设备能够在保护本地数据隐私的同时训练共享模型。这种设计解决了隐私、安全性以及传输大型数据集效率低下等问题 [45, 57]。尽管联邦学习已得到广泛研究
[10, 14, 15, 18, 26, 27, 30, 34–36, 42, 43], , 但现有方法大多聚焦于单模态数据（如图像或文本）。然而，随着移动电话和物联网设备 [6], 的快速发展，本地数据越来越多地来自多种模态。整合此类异构信号能够产生更丰富、更稳健的表征 [19, 48], , 因此人们对多模态联邦学习的兴趣日益增长。然而，当前许多方法依赖定制架构，无法利用预训练多模态模型。近期研究开始探索对预训练多模态模型进行联邦微调，但通常假设每个客户端拥有相同的模态集合 [56], , 这限制了其在现实异构环境中的适用性。

**多模态联邦学习中的缺失数据**。 在现实场景中，本地设备从不同类型的传感器收集数据，因而形成特征模态各异的数据集（外部异质性）。此外，传感器故障或数据损坏可能导致特征缺失，且不同设备呈现出不同的缺失模式（内部异质性）。

现有解决方案要么将每种模态 [55, 61] 独立处理，要么局限于集中式学习 [7, 38, 51], ，两者都忽略了模态间的依赖关系。近期的联邦学习方法处理了外部异质性 [8, 37], 但未解决内部异质性；而 Fed-InMM 等方法 [52] 和 FedMAC [33] 针对客户端内缺失数据，但忽略了跨客户端对齐，因而限制了实际效果。另一类相关工作，包括 FedMSplit [8] 和 CACMRN， [49], 在个性化联邦学习框架下处理模态不完整数据，其中每个客户端都保留自己的模型。这种范式不同于旨在改进单一共享模型的标准联邦学习，因此这些方法无法直接比较。

**使用提示调优应对数据缺失**。提示调优 [41, 44, 59, 60]通过学习能够为输入嵌入添加上下文信息的提示，为适配预训练模型提供了一种有效方法。近期研究将这一思路拓展到模态缺失场景，为每种缺失特征子集学习一个提示 [20, 24], ，从而用合适的提示增强具有相似缺失模式的输入。然而，这些方法是为集中式训练而开发的，并未应对联邦学习中新增的异质性；在联邦学习中，提示必须在客户端之间进行对齐和聚合。因此，如何为此类提示设计有效的对齐与聚合策略是一项关键挑战，正如我们在框架中所展示的那样。

## 3. 多模态联邦提示调优

多模态联邦学习面临着重大挑战：数据模态缺失会导致客户端间和客户端内的异质性，使知识聚合更加复杂，并降低整体模型性能——见图1。为应对这一挑战（第3.1节），我们提出了一种基于预训练视觉语言 Transformer（ViLT）模型的联邦提示调优框架，该框架由基于提示的客户端设计和服务器端提示聚合算法组成。客户端设计采用一种微调算法，将微调后学得的知识分解为若干独立的客户端间提示和客户端内提示，分别针对不同的输入级缺失数据模式以及输入无关的模态缺失模式进行微调（第3.2节）。随后，服务器端算法学习针对相似输入级缺失数据模式的客户端间提示之间的对齐关系，并将其聚合，以生成适用性更广的提示。此外，该算法还结合了标准的基于FedAvg的聚合，对针对输入无关模态缺失模式的客户端内提示进行聚合（第3.3节）——见图2。

### 3.1. 问题表述

在多模态联邦学习（MMFL）中，我们假设有 $n$ 个客户端和 $r$ 种数据模态（例如文本、图像）。

每个客户端 $t \in [n]$[1] 都可以访问一个私有数据集 $D_t \triangleq \{\boldsymbol{x}(M_{t,\ s})z_{t,\ s}\}_{s=1}^{m}$，其中 $\boldsymbol{x}(M_{t,\ s})$表示该模态集合的观测特征，$M_{t,\ s} \subseteq [r]$ 记录的是第 $s$个客户端的第 $t$个数据点的特征，而$z_{t,\ s}$表示对应的目标标签。我们的目标是利用所有私有数据集$\{D_t\}_{t=1}^{n}$ 学习一个多模态模型 $\boldsymbol{w}_*$，同时不对这些数据集进行集中化处理。也就是说，

$$\boldsymbol{w}_* = \arg\min_{\boldsymbol{w}} \left\{ L(\boldsymbol{w}) \triangleq \frac{1}{n}\sum_{t=1}^{n} L_t(\boldsymbol{w}) \right\} \text{ with}$$
$$L_t(\boldsymbol{w}) \triangleq \frac{1}{n}\sum_{s=1}^{m} \ell\Big(F\big(\boldsymbol{x}\big(M_{t,s}\big);\boldsymbol{w}\big), z_{t,s}\Big) \tag{1}$$

其中，$F(\boldsymbol{x}\big(M_{t,\ s}\big);\ \boldsymbol{w})$ 表示参数为 $\boldsymbol{w}$ 的多模态预测模型，而 $\ell(F(\boldsymbol{x}\big(M_{t,\ s}\big);\ \boldsymbol{w})\, z_{t,\ s})$ 表示 $\boldsymbol{w}$ 在第 $s$ 个数据点上的单样本损失。此前，MMFL 领域的研究 [8, 33, 37, 52] 主要致力于设计 $F(\boldsymbol{x}\big(M_{t,\ s}\big);\ \boldsymbol{w})$，而没有利用预训练基础模型中已有的知识。相较之下，我们将 $F(\boldsymbol{x}\big(M_{t,\ s}\big);\ \boldsymbol{w})$ 设为经过提示调优的预训练基于 Transformer 的模型，

$$F\Big(\boldsymbol{x}(M);\boldsymbol{w}\Big) \triangleq F_c\Big(F_p\Big(F_e\big(\boldsymbol{x}(M)\big) \circ \boldsymbol{w}_p\Big);\boldsymbol{w}_c\Big) \tag{2}$$

其中，$F_c(.;\boldsymbol{w}_c)$ 是由 $\boldsymbol{w}_c$ 参数化的预测头，$F_e(\boldsymbol{x}(M))$ $= F_e(\boldsymbol{x}_1) \circ \ldots \circ F_e(\boldsymbol{x}_{|M|})$，其中 $F_e(\boldsymbol{x}_i)$ 表示每个单模态特征 $\boldsymbol{x}_i$ 在 $\boldsymbol{x}(M) \triangleq (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{|M|})$ 中对应的分词后的输入序列，$F_p(.)$ 表示冻结的多模态预训练模型，例如 ViLT [21]，而 $\boldsymbol{w}_p$ 表示（可学习的）提示集合。此处，$\circ$ 表示集合拼接运算符，$\boldsymbol{w} = (\boldsymbol{w}_c, \boldsymbol{w}_p)$ 表示完整的调优参数集合。由于本地数据集 $\{D_t\}_{t=1}^{n}$ 具有去中心化和隐私属性，无法直接求解式（1）。因此，遵循联邦学习 [31] 范式，通过客户端和服务器端交替执行若干轮迭代来近似其最优解 $\boldsymbol{w}_*$：

$$\boldsymbol{w}_t^{(h+1)} = \text{update}\Big(L_t, \boldsymbol{w}_*^{(h)}\Big)\ \forall t \in [n]\,, \tag{3}$$
$$\boldsymbol{w}_*^{(h+1)} = \text{aggregate}\Big(\boldsymbol{w}_1^{(h+1)}, \ldots, \boldsymbol{w}_n^{(h+1)}\Big), \tag{4}$$

其中 $\boldsymbol{w}_t^{(h)}$ 表示客户端 $\boldsymbol{w}_t$ 在 $t$ 经过 $h$ 次迭代后本地调优参数的估计值。例如，更新步骤可以采用对提示 $\boldsymbol{\omega}$ 进行普通梯度下降（GD）优化，而聚合步骤则表示对经过本地优化的提示取平均值。然而，这种普通方法没有考虑本地提示之间的客户端间异质性和客户端内异质性（见第1节），最终可能聚合出不兼容的提示，并降低整体性能（见第4节）。为避免这种情况，我们将引入新颖的客户端提示和服务器端提示设计，具体如下。

[1] $[n] \triangleq \{1, 2, \ldots, n\}$。

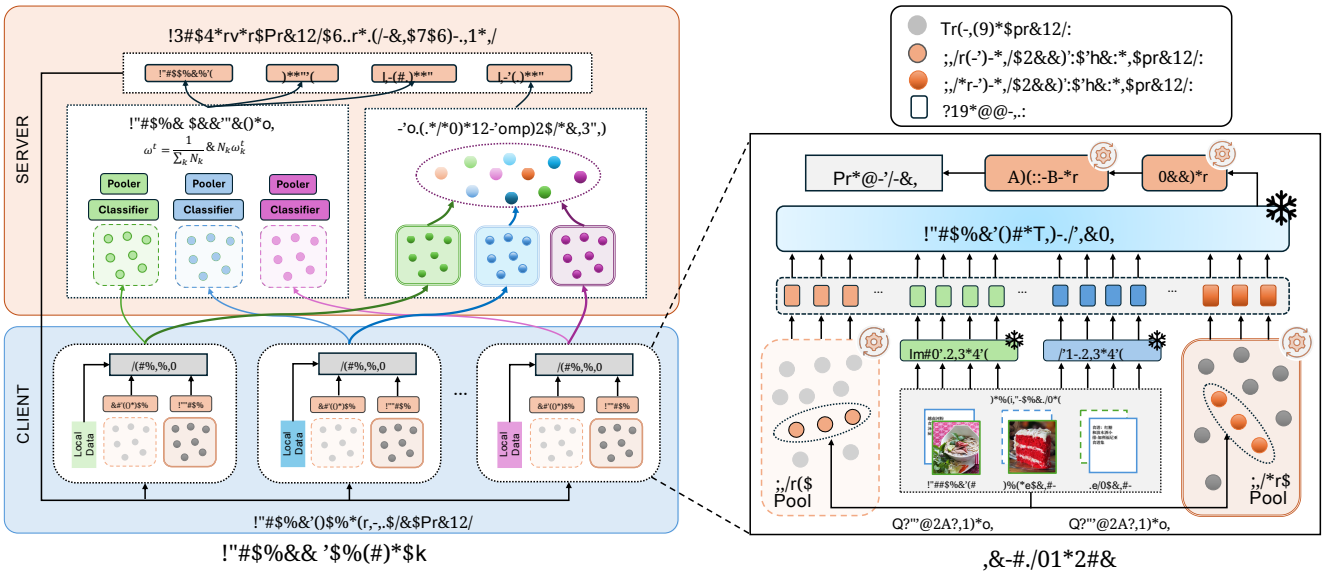


图 2. 所提出的多模态联邦提示调优框架概览——参见算法 3。每个客户端维护两组（可学习的）客户端内提示和客户端间提示。在每次迭代开始时，每个客户端通过公式 (7) 和公式 (8) 进行本地训练。随后，本地的客户端内提示和客户端间提示集将发送至服务器进行聚合——参见公式 (10)–公式 (14)。

### 3.2. 客户端设计

为了实现异质客户端之间有效的提示聚合，我们需要将输入级缺失数据模式和与输入无关的模态缺失模式的微调处理分离开来。通过进一步使用客户端间提示和客户端内提示这两个不同子集对 $\boldsymbol{w}_p = (\boldsymbol{w}_p^{inter}, \boldsymbol{w}_p^{intra})$ 进行参数化，可以实现这一点：

$$\boldsymbol{w}_p^{inter} \triangleq \left\{\boldsymbol{p}_1^{inter}, \boldsymbol{p}_2^{inter}, \ldots, \boldsymbol{p}_\tau^{inter}\right\}, \tag{5}$$

$$\boldsymbol{w}_p^{intra} \triangleq \left\{\boldsymbol{p}_1^{intra}, \boldsymbol{p}_2^{intra}, \ldots, \boldsymbol{p}_\tau^{intra}\right\}, \tag{6}$$

这些提示分别进行学习。跨客户端学习到的客户端间提示 $\boldsymbol{w}_p^{inter}$ 将发送至服务器，由服务器根据其针对相似输入级缺失数据模式的专门性，将其聚类为不同组——见第 3.3 节。客户端内提示 $\boldsymbol{w}_p^{intra}$ 的本地集合可通过联邦平均 [31] 进行平均，因为它们专门用于与输入无关的模态缺失模式。直观而言，聚合机制将决定微调知识的不同方面如何编码到各提示集合中。例如，如果有关不同输入级缺失数据模式的知识被（错误地）纳入 $\{\boldsymbol{p}_i^{intra}\}_{i=1}^{\tau}$，由于 $\{\boldsymbol{p}_i^{intra}\}_{i=1}^{\tau}$ 所采用的非聚类聚合机制，这些知识将被平均并合并为信息量较低的提示。

由于这种潜在的信息损失会降低整体性能，因此将更新（可学习的）提示参数以防止这种情况发生。否则，如果将有关缺失模态的输入无关模式的（常见）知识模式（错误地）纳入 $\{\boldsymbol{p}_i^{inter}\}_{i=1}^{\tau}$，它们对客户端间异质性的建模能力就会降低，从而也会影响性能。反过来，这也会产生提示更新信号，以避免这种情况。此外，在每组客户端间提示和客户端内提示中，也可能存在需要区分并内化到不同提示子集中的不同模式。

为实现这一点，我们针对两组提示开发了一种输入自适应学习机制，用于识别并更新与每个（多模态）输入样本最相关的提示子集。该机制被形式化为信息检索任务：学习一对键函数和查询函数 $k(\boldsymbol{p})$ 与 $q(\boldsymbol{x}(M))$，分别将提示 $\boldsymbol{p}$ 以及具有已观测模态集合 $M$ 的输入 $\boldsymbol{x}$ 映射到共享度量空间中。随后，我们可以定义几何输入-提示距离，例如 $d(\boldsymbol{x}(M), \boldsymbol{p}) = \cos(q(\boldsymbol{x}(M)), k(\boldsymbol{p}))$，用于衡量两者的上下文相关性。为避免将多样的微调知识加载到单个提示中，我们对（本地）损失函数进行正则化：当针对输入 $\boldsymbol{x}(M)$ 选择相关性较低的提示作为微调指令时，对其施加惩罚：

$$\boldsymbol{w}_t \triangleq \arg\min_{\boldsymbol{w}} L_t'(\boldsymbol{w}) \quad \text{where} \tag{7}$$

$$L_t'(\boldsymbol{w}) \triangleq \sum_{s=1}^{m}\left\{\ell\left(F\left(\boldsymbol{x}(M_{t,s}); \boldsymbol{w}'\right), z_{t,s}\right)\right\} + \sum_{s=1}^{m}\left\{r\left(\boldsymbol{x}(M_{t,s}), \boldsymbol{w}_p'\right)\right\}, \tag{8}$$

其中 $\ell(\boldsymbol{w}')$ 定义见公式（1），而 $\boldsymbol{w}' = (\boldsymbol{w}_c, \boldsymbol{w}_p')$表示为每个对象定制的调优指令集

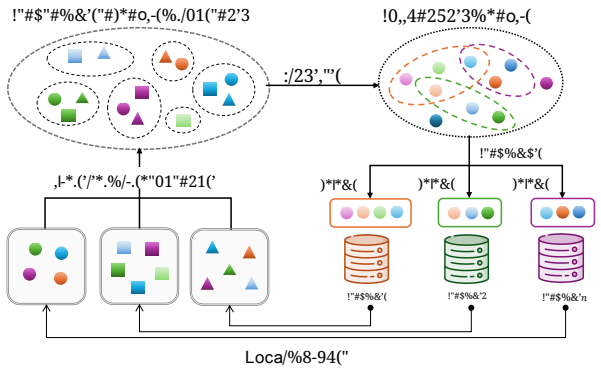


图 3. 提示对齐算法的工作流程。在每次迭代中，每个客户端使用其查询和键函数对摘要提示进行采样。随后，客户端执行本地更新，生成异构提示集，并将其发送至服务器，由服务器进行聚类并将其摘要为下一次迭代所需的新摘要提示集。

输入 $\boldsymbol{x}(M_{t,\ s})$，其中 $\boldsymbol{w}'_p = (\boldsymbol{w}_p^{inter'},\ \boldsymbol{w}_p^{intra'})$，满足$\boldsymbol{w}_p^{inter'} \subseteq \boldsymbol{w}_p^{inter}$和$\boldsymbol{w}_p^{intra'} \subseteq \boldsymbol{w}_p^{intra}$分别是 $\kappa$ 中与 $\boldsymbol{x}(M_{t,\ s})$最相关的提示集。正则化项 $r(\boldsymbol{x}(M)\ \boldsymbol{w}'_p)$ 定义如下：

$$r\Big(\boldsymbol{x}\big(M\big), \boldsymbol{w}'_p\Big) \ \triangleq \ \sum_{\boldsymbol{p}\in\boldsymbol{w}'_p} \Big\{ d\Big(\boldsymbol{x}\big(M\big),\ \boldsymbol{p}\Big)\Big\}. \quad (9)$$

直观地说，正则化项会惩罚（多模态）输入 $\boldsymbol{x}$（$M$）与根据前述距离度量确定的 $\kappa$ 最近提示之间的总距离 $d$（$\boldsymbol{x}$（$M$） $\boldsymbol{p}$）。同时最小化原始局部损失和该正则化项，可以使局部模型同时学会：（1）内化跨客户端异质性和客户端内异质性的微调模式，以最小化原始损失 $L_t$（$\boldsymbol{w}$）；以及（2）确保针对给定输入所选的提示与其保持接近，从而减少惩罚 $r$（$\boldsymbol{x}$（$M$） $\boldsymbol{w}'_p$）。该客户端设计的伪代码详见补充材料A。

### 3.3. 服务器聚合

现在，我们将建立一种有原则的算法，用于对齐和聚合专门针对相似输入级缺失数据模式的客户端间提示[2]。这并非易事，因为不同客户端提示集合中处于相同位置的客户端间提示，可能并不专门针对相似的缺失数据模式，这是由客户端异质性造成的：某些客户端并未观察到某些缺失数据模式。

为了找到正确的对齐方式，我们将其表述为一个聚类优化任务。假设每个客户端 $t$ 都会交换一组客户端间提示$\boldsymbol{w}_t^{inter} = (\boldsymbol{p}_i^t)_{i=1}^{\tau}$。每个

[2]本节仅介绍客户端间提示的聚合算法，因为客户端内提示专门针对输入无关的模态缺失模式，可以通过联邦平均进行聚合 [31]。

这些提示的潜在聚类由此定义了一种候选对齐方式，即同一簇中的提示被视为已对齐，并可进行聚合。随后可为聚合提示关联一个（可学习的）聚类中心——见图 3。通过求解以下优化任务，可以找到聚类分配和中心：每个簇的离散对齐变量 $\alpha_t^{pq} \in \{0,1\}$ 以及连续中心变量 $\boldsymbol{\theta}_q \ \ q \in [n \times \tau]$:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\theta},\gamma} \ G\Big(\boldsymbol{\alpha},\boldsymbol{\theta},\gamma\Big) \ \triangleq \ \sum_{t=1}^{n}\sum_{p=1}^{\tau}\sum_{q=1}^{n\times\tau} \alpha_t^{pq} \cdot c\Big(\boldsymbol{p}_t^p, \boldsymbol{\theta}_q, ;\gamma\Big) \quad (10)$$

$$\text{s.t.}\ \alpha_t^{p,1} + \ldots + \alpha_t^{p,n\times\tau} \ = \ 1\,, \quad (11)$$

其中 $\boldsymbol{\alpha} \triangleq (\alpha^{p,\ _t\ q})$ 和 $\alpha^{p,\ _t\ q} \in \{0,1\}\ \forall(t,\ p,\ q)$ 表示客户端 $t$ 的第 $p$个客户端间提示是否与第 $q$个聚类匹配。随后，根据（可学习的）代价函数 $c(\boldsymbol{p}_t^p,\ \boldsymbol{\theta}_q)$ 优化相应的聚类中心，使其与所有聚类成员之间的总距离最小化。优化后的聚类中心即可用作聚合后的提示或总结性提示。因此，最小化式（10）使我们能够同时对齐并总结/聚合各客户端的客户端间提示。请注意，式（11）施加了一项约束，其直观含义是，同一客户端的两个提示不应匹配到同一聚类并被聚合。这是因为来自同一客户端的本地提示已经学会编码不同的微调方面。因此，聚合这些提示会使其坍缩为信息量较低的提示。

此外，有些总结性／聚合提示可能比其他提示更通用，可用于为更广泛的（多模态）输入提供微调指令。在训练过程中，应优先更新这类提示。为此，我们需要一种机制来衡量其提示流行度。为此，我们在上述式（10）中的优化任务中加入一个正则化项，用以惩罚与较不流行的提示的对齐，

$$R\Big(\boldsymbol{\alpha},\zeta\Big) \ \triangleq \ \sum_{t=1}^{n}\sum_{p=1}^{\tau}\sum_{q=1}^{n\times\tau} \Big(\alpha_t^{p,q} \cdot \log U(\boldsymbol{\theta}_q;\zeta)\Big) \quad (12)$$

$$+ \sum_{t=1}^{n}\sum_{p=1}^{\tau}\sum_{q=1}^{n\times\tau} \Big(1-\alpha_t^{p,q}\Big)\cdot\log\Big(1-U(\boldsymbol{\theta}_q;\zeta)\Big),$$

其中，$U(\boldsymbol{\theta}_q;\ \zeta)$ 是 $\boldsymbol{\theta}_q$ 的流行度函数，由可学习参数 $\zeta$ 参数化。例如，$U(\boldsymbol{\theta}_q;\ \zeta) \ = \ \sigma(g(\boldsymbol{\theta}_q;\ \zeta)) \ \in (0,1)$[3] 其中，$g(\boldsymbol{\theta}_q;\ \zeta)$ 是由 $\zeta$ 参数化的深度神经网络。结合式（13）和式（10），我们得到最终的损失函数，

$$\underset{\boldsymbol{\alpha},\boldsymbol{\theta},\gamma,\zeta}{\text{minimize}} \ \Big\{G\Big(\boldsymbol{\alpha},\boldsymbol{\theta},\gamma\Big) \ + \ R\Big(\boldsymbol{\alpha},\zeta\Big)\Big\}, \quad (13)$$

[3]$\sigma$（。）表示 Sigmoid 函数。

该函数可通过交替执行以下步骤进行优化：（1）优化（$\boldsymbol{\theta}$，$\zeta$，$\gamma$），同时固定 $\boldsymbol{\alpha}$；（2）优化 $\boldsymbol{\alpha}$，同时固定（$\boldsymbol{\theta}$，$\zeta$，$\gamma$）。第一个优化子任务较为直接，而第二个则因约束条件以及优化变量的离散性质而较为复杂。为避开在固定（$\boldsymbol{\theta}$，$\zeta$，$\gamma$）时求解 $\boldsymbol{\alpha}$ 的技术难题，我们通过优化$\boldsymbol{\alpha}_t = \{\alpha_t^{p,q}\}_{p,\ q}$ 来近似求解，同时也固定 $\boldsymbol{\alpha}_{-t} = \{\alpha_{-t}^{p,q}\}_{p,\ q}$。由此得到一个最小加权线性和任务，

$$\min_{\boldsymbol{\alpha}_t} \ L(\boldsymbol{\alpha}_t) \triangleq \sum_{p=1}^{\tau}\sum_{q=1}^{n\times\tau}\Big(\alpha_t^{p,q}\cdot v(p,q)\Big) \ \text{ where} \tag{14}$$

$$v(p,q) \triangleq \left(\log\left\{\frac{U\Big(\boldsymbol{\theta}_q;\zeta\Big)}{1-U\Big(\boldsymbol{\theta}_q;\zeta\Big)}\right\} + c\Big(\boldsymbol{p}_t^p,\boldsymbol{\theta}_q;\gamma\Big)\right),$$

可使用匈牙利算法在多项式时间内有效求解 [23]。服务器端聚合算法的概述和伪代码见图3及补充材料A。下一节将详细评估该框架4。

## 4. 实证评估

本节将对我们提出的框架FED-PRIME进行详细的实证评估。[4] 我们的评估采用两个数据集：UPMC Food-101 [46] 和 MM-IMDB [1]，并采用若干基线方法。这些方法将联邦微调（针对单模态数据）和多模态提示调优（针对集中式数据）中的现有工作，适配到我们针对存在模态缺失和特征缺失的去中心化多模态数据所提出的新设置中。数据集和基线方法的详细介绍见第4.1节。主要结果和消融研究见第4.2节。

### 4.1. 实验设置

**A. 数据集**。遵循 [24], 中的实验方案，我们在两个多模态数据集上评估 FED-PRIME:

**1.**UPMC Food-101 [46]是一个图文分类数据集，由 Google 图片搜索收集的带噪图文对组成。其类别与规模更大、公开可用的 ETHZ Food-101 数据集 [4]中的类别相对应。在联邦设置中，我们仅从原始 UPMC Food-101 中选取出现频率最高的八个类别。

**2.** MM-IMDB [1]是一个包含图像和文本两种模态的电影类型分类数据集。由于每部电影可能属于多个类型，因此该任务最初是一个多标签分类问题，使用图像、文本或两者来预测电影类型。在联邦学习设置中，我们仅使用单标签实例，并且只从原始 MM-IMDB 中选择出现频率最高的八个类别。

[4]我们的实现可在https://github.com/ hangpt01/FedPrime上获取。

首先，按照 80:20 的比例将每个数据集划分为训练集和测试集。训练集/测试集数据中的（异构）缺失模式通过以下协议进行模拟。

**B. 训练数据模拟**。遵循 [24], ，我们使用缺失率 $\eta \in (0, 1)$ 在数据集中模拟数据缺失现象，具体如下所述的 3 种模拟情况:

**1. 图像缺失**。我们移除 $\eta$ 百分比数据中的图像特征，并保留其余 $(1-\eta)$ 百分比的完整数据。

**2. 文本缺失**。我们移除 $\eta$ 百分比数据中的文本特征，并保留其余 $(1-\eta)$ 百分比的完整数据。

**3. 图文均缺失**。我们移除 $\eta/2$ 百分比数据中的图像组件，并移除另外 $\eta/2$百分比数据中的文本组件，剩余 $(1-\eta)$ 百分比的数据保持完整。

随后，通过随机采样将模拟缺失数据后的训练集划分为 $n$个子集，分配给 $n$ 个客户端。因此，生成的客户端数据集在缺失数据模式上的分布各不相同。

**C. 测试数据模拟**。共有 5 种测试场景，包括：**(1)全模态**（无数据缺失）；**(2)仅图像**（删除所有文本特征）；**(3) 仅文本**（删除所有图像特征）；**(4) 两者均缺失**（在测试数据上运行**图文均缺失**）；以及**(5) ∼训练**（与训练数据采用相同的模拟方式）。

**D. 评估指标**。遵循 [24], 中的既有协议，我们使用宏平均 F1 值和分类准确率，评估 MM-IMDB 和 UPMCFood-101 上的以下基线方法。

**E. 基线方法**。遵循 [24], ，我们采用预训练多模态 Transformer ViLT [21] 作为客户端设计的骨干网络。每个客户端模型都是基于其自身本地数据集生成的该骨干网络的微调版本。由于此前尚无研究探讨联邦学习场景下多模态微调中的模态缺失问题，我们的对比基线主要采用了现有联邦学习和多模态提示调优方法的改编版本。这包括将最新的缺失模态提示调优最先进方法 [24]融入经典的 FedAvg 方法以及较新的联邦学习方法 FedMSplit [8]。这两种方法分别称为 FEDAVG-P 和 FEDMSPLIT-P。此外，为了强调在客户端内提示集与客户端间提示集之间分离微调知识的重要性，我们还纳入了 FED-PRIME的两个简化变体，分别采用客户端间提示池化或客户端内提示池化，并使用各自对应的聚合方式。这两种变体分别称为 FED-INTER 和FED-INTRA（更多详情见补充材料 B）。

### 4.2. 实验结果

本节对FED-PRIME 与其他基线方法在训练集和测试集中各种缺失数据设置下的性能进行了广泛比较，正如第4.1节所述。此外，我们还提供了

表1. 基线方法在 Food-101（左）和 MM-IMDB（右）数据集上，于训练集和测试集的各种缺失数据设置下所取得的性能。其中包括每个数据集中 3 种训练场景与 5 种测试场景的全部组合（即 15 项实验），如第 4.1 节所述。最佳结果和次佳结果分别以**粗体红色** 和 蓝色 标出。 请注意，在 **两者均缺失** 训练场景中， **测试（ ∼ 训练）** 与 **测试（双缺失）** 根据 **∼ 训练** 的定义表示同一测试场景。此情况下 **测试（双缺失）** 所报告的结果与 **测试（∼训练）** 相同，以短横线（-）表示。

| | 数据集 | UPMC Food-101 | | | | | MM-IMDB | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 训练 | 方法 | Test (∼ 训练) | Test (两者均缺失) | Test (全模态) | Test (仅文本) | Test (仅图像) | Test (∼ 训练) | Test (两者均缺失) | Test (全模态) | Test (仅文本) | Test (仅图像) |
| Miss Text | FEDAVG-P | 15.71 ± 2.27 | 14.90 ± 1.57 | 21.56 ± 7.81 | 16.91 ± 0.69 | 15.36 ± 0.43 | 22.42 ± 2.27 | 21.89 ± 1.34 | 30.78 ± 1.65 | 18.40 ± 0.06 | 14.53 ± 4.56 |
| | FEDMSPLIT-P | 15.62 ± 1.51 | 17.50 ± 1.97 | 25.27 ± 8.22 | 18.78 ± 0.64 | 17.50 ± 1.97 | 21.02 ± 1.89 | 19.97 ± 0.74 | 24.39 ± 6.08 | 14.38 ± 2.76 | 18.09 ± 6.13 |
| | FED-INTER | 54.82 ± 19.01 | 48.87 ± 24.64 | 59.17 ± 27.06 | 35.13 ± 26.78 | 56.59 ± 15.12 | 18.25 ± 3.50 | 16.95 ± 3.57 | 18.67 ± 7.63 | 15.03 ± 4.66 | 18.01 ± 1.95 |
| | FED-INTRA | 61.71 ± 17.22 | 48.09 ± 19.12 | 62.06 ± 26.98 | 22.51 ± 5.92 | 62.64 ± 11.83 | 13.38 ± 1.73 | 12.77 ± 0.85 | 12.55 ± 1.67 | 11.31 ± 0.38 | 14.33 ± 1.80 |
| | **FED-PRIME** | **78.88 ± 0.90** | **80.38 ± 0.65** | **92.12 ± 0.40** | **73.01 ± 4.25** | **76.83 ± 1.22** | **31.92 ± 0.20** | **31.48 ± 0.30** | **37.67 ± 0.04** | **30.60 ± 1.41** | **30.69 ± 1.41** |
| | **改进（%）** | **27.82** ↑ | **64.48** ↑ | **48.44** ↑ | **107.83** ↑ | **22.65** ↑ | **42.37** ↑ | **43.81** ↑ | **22.35** ↑ | **66.30** ↑ | **69.65** ↑ |
| Miss 图像 | FEDAVG-P | 17.35 ± 4.77 | 15.12 ± 1.48 | 16.84 ± 2.37 | 18.12 ± 6.49 | 14.81 ± 0.24 | 27.69 ± 5.97 | 22.55 ± 3.06 | 31.94 ± 0.98 | 23.76 ± 11.72 | 12.29 ± 0.47 |
| | FEDMSPLIT-P | 74.16 ± 10.56 | 48.88 ± 10.26 | 45.64 ± 32.43 | 88.65 ± 2.17 | 14.81 ± 0.90 | 19.11 ± 11.33 | 16.61 ± 7.22 | 18.19 ± 12.55 | 18.12 ± 12.30 | 12.81 ± 1.25 |
| | FED-INTER | 77.96 ± 11.62 | 64.62 ± 10.22 | 82.08 ± 7.75 | 77.69 ± 12.35 | 37.56 ± 6.49 | 18.79 ± 5.23 | 17.93 ± 3.60 | 20.56 ± 3.56 | 17.67 ± 6.79 | 15.47 ± 2.51 |
| | FED-INTRA | 22.84 ± 3.52 | 20.13 ± 1.72 | 23.48 ± 1.86 | 24.46 ± 2.99 | 16.66 ± 1.32 | 15.75 ± 4.34 | 14.06 ± 3.16 | 15.68 ± 4.77 | 14.53 ± 3.65 | 11.71 ± 0.42 |
| | **FED-PRIME** | **90.55 ± 0.22** | **79.12 ± 0.49** | **92.89 ± 0.21** | **90.18 ± 0.29** | **54.14 ± 2.50** | **36.08 ± 0.35** | **31.35 ± 0.61** | **38.49 ± 0.56** | **36.91 ± 0.59** | **18.15 ± 0.66** |
| | **改进（%）** | **16.15** ↑ | **22.44** ↑ | **13.17** ↑ | **1.73** ↑ | **44.14** ↑ | **30.30** ↑ | **39.02** ↑ | **20.51** ↑ | **55.35** ↑ | **17.32** ↑ |
| Miss Both | FEDAVG-P | 14.57 ± 1.50 | - | 17.17 ± 4.37 | 16.40 ± 4.05 | 13.24 ± 0.32 | 26.45 ± 2.63 | - | 33.03 ± 2.56 | 24.12 ± 11.30 | 20.21 ± 1.98 |
| | FEDMSPLIT-P | 49.15 ± 24.76 | - | 64.78 ± 36.62 | 64.62 ± 36.51 | 21.49 ± 7.19 | 24.25 ± 5.02 | - | 26.05 ± 11.17 | 26.02 ± 9.64 | 19.79 ± 6.20 |
| | FED-INTER | 56.32 ± 21.77 | - | 69.57 ± 19.41 | 45.15 ± 34.09 | 59.30 ± 10.84 | 26.53 ± 0.90 | - | 31.97 ± 2.22 | 29.69 ± 2.21 | 21.63 ± 0.77 |
| | FED-INTRA | 49.28 ± 32.87 | - | 56.70 ± 37.90 | 43.24 ± 34.19 | 49.85 ± 25.44 | 11.90 ± 0.37 | - | 12.47 ± 0.45 | 11.46 ± 0.33 | 12.83 ± 0.92 |
| | **FED-PRIME** | **84.44 ± 2.65** | - | **93.64 ± 0.58** | **87.95 ± 0.91** | **72.41 ± 3.88** | **32.01 ± 2.51** | - | **38.68 ± 0.65** | **31.00 ± 2.97** | **26.01 ± 0.12** |
| | **改进 (%)** | **49.93** ↑ | - | **34.60** ↑ | **36.10** ↑ | **22.11** ↑ | **20.66** ↑ | - | **17.11** ↑ | **4.41** ↑ | **20.25** ↑ |

**(*) 改进** 表示我们的方法与次优方法之间的相对性能提升（百分比）。

关于以下方面的消融研究：(1) 对各种缺失率 $\eta$的性能鲁棒性；(2) 设置客户端间和客户端内提示集的消融影响；以及 (3) 训练收敛性。

#### 4.2.1. 主要结果

我们在所有数据集上的大量实验中，将 FED-PRIME 的性能与其他基线方法进行比较。这包括训练数据上的 3 种缺失模拟场景与 5 种测试场景（同时包含缺失数据与无缺失数据）的全部组合，如第4.1节 B 部分和 C 部分先前所述。所有结果均报告于表 1，表明 FED-PRIME 在具有各种缺失数据模式的全部训练/测试场景中始终优于所有基线方法。UPMC Food-101 与 MM-IMDB 数据集上的性能分别以分类准确率和宏平均F1值衡量。由于该数据集具有类别不平衡特性，MM-IMDB 上的宏平均F1值在所有设置中都相对较低。在每项实验中，均报告了 FED-PRIME 相对于次优基线方法的相对性能提升，其在 UPMC Food-101 数据集上介于 1.73% 与 107.83% 之间，在 MM-IMDB 数据集上介于 4.41% 与 69.65% 之间。值得注意的是，FED-PRIME 在所有设置中均显著优于其两个简化变体 FED-INTRA 和 FED-INTER，二者分别仅处理内部异质性或相互异质性。这凸显了在存在缺失数据的多模态联邦学习中同时处理两类异质性所带来的显著协同效应及其必要性。此外，还可以观察到，在超过一半的实验设置中，次优基-线方法要么是FED-INTRA 或FED-INTER，其显著优于两者FEDAVG-P 和FEDMSPLIT-P 在这些情况下。这些观察结果补充了我们在消融研究中关于这些组件各自独立影响的发现，见第4.2.4节。

#### 4.2.2. 模态缺失的影响。

在不同的测试场景中，正如预期的那样，我们观察到，当测试集中没有缺失数据时，测试性能始终最高。另一个有趣的发现是，在**文本缺失**训练场景中，训练数据中约有 70% 的文本特征丢失，但所提出的模型在**仅文本** 测试场景中仍然表现出色。这表明，多模态提示对齐能够有效恢复文本信息，凸显了所提出设计的适应性。然而，在**图像缺失**训练场景中，**仅图像** 情况下的测试性能却显著下降。经调查，我们认为这很可能是由于 VILT 骨干网络的图像处理模块较为轻量，该模块经过优化，可与文本协同工作以实现有竞争力的性能，这也暗示了 VILT 在预训练期间可能以文本为中心。事实上，这与已有观察结果十分一致：与文本进行协同训练通常会提升性能。

#### 4.2.3. 对不同缺失率的鲁棒性。

为展示 FED-PRIME 对不同缺失率 $\eta$ 的鲁棒性，我们对 FED-PRIME、FEDAVG-P 及其集中式变体 [24] 进行了多组比较

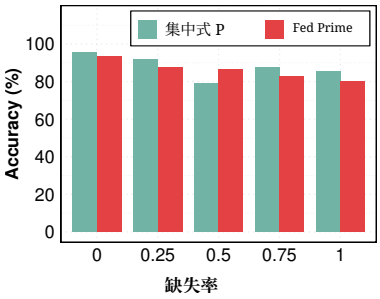


图 4在不同缺失率的两者均缺失场景下，UMPC Food-101 上的性能比较。

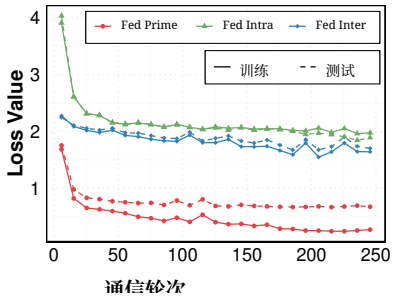


图 5Food-101 在文本缺失情况下的训练/测试损失收敛曲线。FED-PRIME 收敛速度最快。

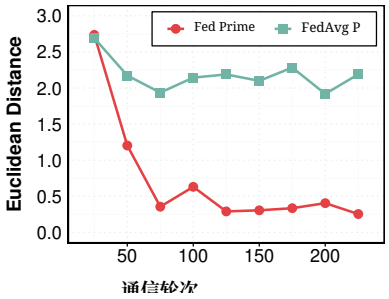


图 6 文本缺失场景下，UPMC Food-101 上聚合提示质心之间欧氏距离的收敛情况。

在广泛的 $\eta \in \{0.00, 0.25, 0.50, 0.75, 1.00\}$ **两者均缺失**训练场景下。图1和图4 中的结果表明，我们的模型显著优于 FEDAVG-P及其集中式对应方法 Centralized-P。随着缺失率增加，我们的方法始终保持着超过80%的竞争力，准确率接近集中式设置下 FEDAVG-P的性能上限，这得益于集中式数据集中的模态对齐程度更高。这些结果证明了我们的模型在不同缺失率下具有鲁棒性，并凸显了联邦提示调优方法的有效性：尽管只能访问具有异构缺失模式的去中心化本地数据集，其性能仍可媲美集中式提示调优模型。

#### 4.2.4. 消融研究

如第4.1节所述，FED-INTER和FED-INTRA 分别是从客户端设计中移除客户端内和客户端间提示集后，FED-PRIME 的变体。表1 显示，FED-PRIME 在所有测试条件下的表现均始终优于两者，且优势显著。缺少客户端内提示会导致FED-INTER 表现逊于 FED-PRIME ，这表明，设置专用提示集来捕捉微调中共有且高度可泛化、与输入无关的方面，对于提升性能至关重要。同样，FED-INTRA由于省略了客户端间提示集，性能也显著下降。这凸显了通过提示对齐捕捉并正确聚合输入级缺失数据模式的重要性。

#### 4.2.5. 收敛分析

**模型收敛**。 图5 显示，FED-PRIME 表现出更优的性能，相较于FED-INTER 和 FED-INTRA，在训练和测试中均实现了更快的收敛速度以及显著更低的最终损失值。这表明其具有更高的学习效率和更强的鲁棒性。两者FED-INTER 和 FED-INTRA收敛速度更慢，且损失值始终较高，尤其是在早期轮次。此外，我们的方法在服务器聚合轮次之间更加稳定，波动极小。相比之下，FED-INTER 和 FED-INTRA 表现出明显更大的变化，尤其是在后期。这凸显了 FED-PRIME 的有效性，其在各轮次中具有更快的收敛速度和更高的稳定性。

**提示收敛**。图6 显示，FED-PRIME缩短聚合提示质心之间距离的速度远快于FEDAVG-P，凸显出 FED-PRIME相较于FEDAVG-P具有更快的收敛速度和更好的提示对齐。相比之下，FEDAVG-P 收敛较慢且不稳定，这可能是默认提示对齐不正确所致。这再次强调，有必要采用可学习的提示聚类机制，以避免提示对齐错误。

## 5. 结论

针对特定任务微调大型预训练模型，已成为机器学习中日益有效的解决方案范式。然而，在实际的多模态场景中，数据分布于多个私有数据孤岛，各处的模态缺失和特征缺失模式各不相同，现有微调框架难以应对。为解决这些挑战，本文提出一种新颖的联邦提示微调方法，通过优化、对齐和聚合提示指令，弥补多个异构私有本地数据集在数据模态和分布模式方面的差异。在多模态基准数据集上进行的严格评估表明，面对本地数据集中各种缺失数据模式，我们的方法依然有效且稳健。我们认为，这项工作是迈向开发稳健、适应性强的联邦模型的初步探索，这类模型能够在真实世界环境中充分利用多模态数据的多样性。

## 致谢

本研究由河内科技大学（HUST）通过资助编号 T2024-TD-002 提供经费。此外，本研究使用了圣地亚哥超级计算中心（SDSC）的 GPU 计算资源，并通过高级网络基础设施协调生态系统：服务与支持（ACCESS）计划的 CIS230391 资源分配获得 ACES 的支持 [2]，该计划由美国国家科学基金会资助，资助编号为 #2138259,#2138286, #2138307, #2137603, and#2138296。

## 参考文献

[1] John Arevalo、Thamar Solorio、Manuel Montes y G´omez 和 Fabio A. Gonz´alez。《用于信息融合的门控多模态单元》。*arXiv*，abs/1702.01992，2017年。2，6，3[2] Timothy J. Boerner、Stephen Deems、Thomas R. Furlani、Shelley L. Knuth 和 John Towns。《ACCESS：推动创新：美国国家科学基金会高级网络基础设施协调生态系统：服务与支持》。载于《高级研究计算实践与经验 *2023*：为共同利益而计算》，第 173–176 页，美国纽约州纽约，2023年。美国计算机协会。9[3] Rishi Bommasani、Drew A. Hudson、Ehsan Adeli、Russ Altman、Simran Arora、Sydney von Arx、Michael S. Bernstein、Jeannette Bohg、Antoine Bosselut、Emma Brunskill、Erik Brynjolfsson、S. Buch、Dallas Card、Rodrigo Castellon、Niladri S. Chatterji、Annie S. Chen、Kathleen A. Creel、Jared Davis、Dora Demszky、Chris Donahue、Moussa Doumbouya、Esin Durmus、Stefano Ermon、John Etchemendy、Kawin Ethayarajh、Li Fei-Fei、Chelsea Finn、Trevor Gale、Lauren E. Gillespie、Karan Goel、Noah D. Goodman、Shelby Grossman、Neel Guha、Tatsunori Hashimoto、Peter Henderson、John Hewitt、Daniel E. Ho、Jenny Hong、Kyle Hsu、Jing Huang、Thomas F. Icard、Saahil Jain、Dan Jurafsky、Pratyusha Kalluri、Siddharth Karamcheti、Geoff Keeling、Fereshte Khani、O. Khattab、Pang Wei Koh、Mark S. Krass、Ranjay Krishna、Rohith Kuditipudi、Ananya Kumar、Faisal Ladhak、Mina Lee、Tony Lee、Jure Leskovec、Isabelle Levent、Xiang Lisa Li、Xuechen Li、Tengyu Ma、Ali Malik、Christopher D. Manning、Suvir P. Mirchandani、Eric Mitchell、Zanele Munyikwa、Suraj Nair、Avanika Narayan、Deepak Narayanan、Benjamin Newman、Allen Nie、Juan Carlos Niebles、Hamed Nilforoshan、J. F. Nyarko、GirΓay Ogut、Laurel Orr、Isabel Papadimitriou、Joon Sung Park、Chris Piech、Eva Portelance、Christopher Potts、Aditi Raghunathan、Robert Reich、Hongyu Ren、Frieda Rong、Yusuf H. Roohani、Camilo Ruiz、Jack Ryan、Christopher R'e、Dorsa Sadigh、Shiori Sagawa、Keshav Santhanam、Andy Shih、Krishna Parasuram Srinivasan、Alex Tamkin、Rohan Taori、Armin W. Thomas、Florian Tram`er、Rose E. Wang、William Wang、Bohan Wu、Jiajun Wu、Yuhuai Wu、Sang Michael Xie、Michihiro Yasunaga、Jiaxuan You、Matei A. Zaharia、Michael Zhang、Tianyi Zhang、Xikun Zhang、Yuhui Zhang、Lucia Zheng、Kaitlyn Zhou 和 Percy Liang。《基础模型的机遇与风险》。*arXiv*，2021年。1

[4] Lukas Bossard、Matthieu Guillaumin 和 Luc Van Gool。Food-101——利用随机森林挖掘判别性成分。载于欧洲计算机视觉会议，2014年。6

[5] Tom Brọwn、Benjamin Mann、Nick Ryder、Melanie Sub- biah、Jared Kaplan、Prafulla Dhariwal、Arvind Neelakan- tan、Pranav Shyam、Girish Sastry、Amanda Askell、Sand- hini Agarwal、Ariel Herbert-Voss、Gretchen Krueger、Tom Henighan、Rewon Child、Aditya Ramesh、Daniel M. Ziegler、Jeffrey Wu、Clemens Winter、Christopher Hesse、Mark Chen、Eric Sigler、Mateusz Litwin、Scott Gray、Benjamin Chess、Jack Clark、Christopher Berner、Sam McCandlish、Alec Radford、Ilya Sutskever 和 Dario Amodei。《语言模型是少样本学习者》。*CoRR*，abs/2005.14165，2020 年。1

[6] Alberto Brunete、Ernesto Gambao、Miguel Hernando 和 Raquel Cedazo。面向医疗环境中物联网设备、机器人系统和多模态界面的集成的智能辅助架构。*Sensors*，21(6):2212，2021 年。1，2

[7] Jiayi Chen 和 Aidong Zhang。HGMF：面向不完整多模态数据的异构图融合。在第*26*届*ACM*知识发现与数据挖掘国际会议论文集，第1295–1305页，2020年3– 。[8] Jiayi Chen 和 Aidong Zhang。FedMSplit：面向多模态拆分网络的相关性感知联邦多任务学习。在第*28*届*ACM SIGKDD*知识发现与数据挖掘会议论文集，第87–96页，2022年。1，3，6[9] Jinyu Chen、Wenchao Xu、Song Guo、Junxiao Wang、Jie Zhang 和 Haozhao Wang。FedTune：深入探究使用预训练变换器进行高效联邦微调，2022年。1[10] Yae Jee Cho、Andre Manoel、Gauri Joshi、Robert Sim 和 Dimitrios Dimitriadis。用于在联邦学习中训练大型模型的异构集成知识迁移。计算研究知识库 *arXiv* 预印本，2204.12703，2022年。2[11] Jia Deng、Wei Dong、Richard Socher、Li-Jia Li、Kai Li 和 Li Fei-Fei。ImageNet：大型层次化图像数据库。在*2009*年*IEEE*计算机视觉与模式识别会议，第248–255页，2009年。1[12] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。BERT：用于语言理解的深度双向变换器预训练。在*2019*年北美计算语言学协会年会：人类语言技术，第*1*卷（长短篇论文）论文集，第4171–4186页，明尼阿波利斯，明尼苏达州，2019年。计算语言学协会。1[13] Alexey Dosovitskiy、Lucas Beyer、Alexander Kolesnikov、Dirk Weissenborn、Xiaohua Zhai、Thomas Unterthiner、Mostafa Dehghani、Matthias Minderer、Georg Heigold、Sylvain Gelly、Jakob Uszkoreit 和 Neil Houlsby。一张图像胜过16×16个词：面向大规模图像识别的变换器。*ICLR*，2021年。1[14] Alireza Fallah、Aryan Mokhtari 和 Asuman Ozdaglar。个性化联邦学习：一种元学习方法

方法。计算研究知识库 arXiv 预印本，2002.07948，2020年。2[15] Ziwei Fan、Hao Ding、Anoop Deoras 和 Trong Nghia Hoang。面向逐项推荐的个性化联邦域自适应。在第三十九届人工智能不确定性会议论文集，第560–570页。PMLR，2023年。2[16] Tao Guo、Song Guo、Junxiao Wang、Xueyang Tang 和 Wenchao Xu。PromptFL：让联邦参与方协作学习提示，而非模型——基础模型时代的联邦学习。《IEEE移动计算汇刊》，2023年。1[17] Minh Hoang、Nghia Hoang、Bryan Kian Hsiang Low 和 Carleton Kingsford。多个黑盒专家的集体模型融合。在第36届国际机器学习会议论文集，第2742–2750页。PMLR，2019年。2[18] Trong Nghia Hoang、Quang Minh Hoang、Kian Hsiang Low 和 Jonathan P. How。大规模多智能体系统中的高斯过程集体在线学习。在AAAI人工智能会议论文集，第7850–7857页，2019年。2[19] Par minder Kaur、Husanbir Singh Pannu 和 Avleen Kaur Malhi。跨模态信息检索的比较分析：综述。《计算机科学评论》，39，100336，2021年 2：。[20] Muhammad Uzair khattak、Hanoona Rasheed、Muhammad Maaz、Salman Khan 和 Fahad Shahbaz Khan。Maple：多模态提示学习。在IEEE/CVF计算机视觉与模式识别会议，2023年。3[21] Wonjae Kim、Bokyung Son 和 Ildoo Kim。ViLT：无需卷积或区域监督的视觉-语言变换器。在第38届国际机器学习会议论文集，第5583–5594页。PMLR，2021年。1，3，6[22] Ranjay Krishna、Yuke Zhu、Oliver Groth、Justin Johnson、Kenji Hata、Joshua Kravitz、Stephanie Chen、Yannis Kalanitidis、Li-Jia Li、David A. Shamma、Michael S. Bernstein 和 Li Fei-Fei。Visual Genome：利用众包密集图像标注连接语言与视觉。《国际计算机视觉杂志》，123：32–73，2016年。1[23] Harold W Kuhn。用于指派问题的匈牙利算法。《海军研究后勤季刊》，2（97，1955年1）：83–。[24] Yi-Lun Lee、Yi-Hsuan Tsai、Wei-Chen Chiu 和 Chen-Yu Lee。面向视觉识别的缺失模态多模态提示。在IEEE计算机视觉与模式识别会议（CVPR），2023年。2，3，6，7，1[25] Brian Lester、Rami Al-Rfou 和 Noah Constant。规模对参数高效提示调优的作用。在2021年自然语言处理经验方法会议论文集，第3045–3059页，线上及多米尼加共和国蓬塔卡纳，2021年。计算语言学协会。1[26] Tian Li、Anit Kumar Sahu、Manzil Zaheer、Maziar Sanjabi、Ameet Talwalkar 和 Virginia Smith。异构网络中的联邦优化。《机器学习与系统论文集》，2：429–450，2020年。2，4

[27] Xiang Li、Kaixuan Huang、Wenhao Yang、S husen Wang 和 Zhihua Zhang。FedAvg 在非独立同分布数据上的收敛性。计算研究知识库 arXiv 预印本，1907.02189t，2019年。2[28] Tsung-Yi Lin、Michael Maire、Serge J. Belongie、James Hays、Pietro Perona、Deva Ramanan、Piotr Doll´ar 和 C. Lawrence Zitnick。Microsoft COCO：上下文中的常见物体。载于欧洲计算机视觉会议，2014年。1[29] Wang Lu、Xixu Hu、Jindong Wang 和 Xing Xie。FedCLIP：面向联邦学习中 CLIP 的快速泛化与个性化，2023 1 。[30] Tengfei Ma、Trong Nghia Hoang 和 Jie Chen。利用共识图对基于不同特征预训练的模型进行联邦学习。载于人工智能中的不确定性，页码1336–1346，2023年。2[31] Brendan McMahan、Eider Moore、Daniel Ramage、Seth Hampson 和 Blaise Aguera y Arcas。基于去中心化数据进行深度网络通信高效学习。载于人工智能与统计学，页码1273 1282，2017年 2、3、4、5 – 。[32] John Nguyen、Jianyu Wang、Kshitiz Malik、Maziar Sanjabi 和 Michael Rabbat。从何处着手？预训练和初始化对联邦学习的影响，2023年。1[33]Manh Duong Nguyen、Trung Thanh Nguyen、Huy-Hieu Pham、Trong Nghia Hoang、Phi Le Nguyen 和 Thanh Trung Huynh。Fedmac：通过跨模态聚合和对比正则化解决联邦学习中的部分模态缺失问题。2024年。1、3[34] Nang Hung Nguyen、Phi Le Nguyen、Thuy Dung Nguyen、Trung Thanh Nguyen、Duc Long Nguyen、Thanh Hung Nguyen、Huy Hieu Pham 和 Thao Nguyen Truong。Fed-DRL：基于深度强化学习的自适应聚合，用于联邦学习中的非独立同分布数据。载于第51届国际并行处理会议论文集，页码1–11，2022年。2[35] Nang Hung Nguyen、Duc Long Nguyen、Trong Bang Nguyen、Thanh-Hung Nguyen、Huy Hieu Pham、Truong Thao Nguyen 和 Phi Le Nguyen。CADIS：通过聚类聚合和知识蒸馏正则化处理联邦学习中的簇偏斜非独立同分布数据。载于第23届IEEE/ACM集群、云与互联网计算国际研讨会论文集，页码249–261，2023年。[36] Quang Ha Pham、Nang Hung Nguyen、Thanh Hung Nguyen、Huy Hieu Pham、Phi Le Nguyen 和 Truong Thao Nguyen。SEM：一种简单而高效的模型无关本地训练机制，用于解决联邦学习中的数据稀疏和数据不足问题。载于第11届计算与网络研讨会论文集，页码120–126，2023年。2[37] Thu Hang Phung、Binh P Nguyen、Thanh Hung Nguyen、Quoc Viet Hung Nguyen、Phi Le Nguyen 和 Thanh Trung Huynh。一种针对多模态联邦学习中模态缺失问题的对比学习与基于图的方法。载于2024年国际神经网络联合会议论文集，页码1–8，2024年。1、3

[38] Petra Poklukar、Miguel Vasco、Hang Yin、Francisco S Melo、Ana Paiva 和 Danica Kragic。几何多模态对比表征学习。载于2022年国际机器学习会议论文集，页码17782–17800，2022年。3[39]Alec Radford、Jong Wook Kim、Chris Hallacy、Aditya Ramesh、Gabriel Goh、Sandhini Agarwal、Girish Sastry、Amanda Askell、Pamela Mishkin、Jack Clark、Gretchen Krueger 和 Ilya Sutskever。通过自然语言监督学习可迁移的视觉模型。CoRR，abs/2103.00020，2021年。1[40] Colin Raffel、Noam Shazeer、Adam Roberts、Katherine Lee、Sharan Narang、Michael Matena、Yanqi Zhou、Wei Li 和 Peter J. Liu。探索统一的文本到文本变换器的迁移学习极限。《机器学习研究期刊》，21(140)：1–67，2020年。1[41] Kuniaki Saito、Kihyuk Sohn、Xiang Zhang、Chun-Liang Li、Chen-Yu Lee、Kate Saenko 和 Tomas Pfister。前缀条件化统一语言与标签监督。载于IEEE/CVF计算机视觉与模式识别会议（CVPR）论文集，页码2861–2870，2023年。1、3[42]Rachael Hwee Ling Sim、Yehong Zhang、Trong Nghia Hoang、Xinyi Xu、Bryan Kian Hsiang Low 和 Patrick Jaillet。私有协作机器学习中的激励机制。载于第三十七届神经信息处理系统会议，2023年。2[43] Canh T Dinh、Nguyen Tran 和 Josh Nguyen。基于Moreau包络的个性化联邦学习。《神经信息处理系统进展》，33：21394–21405，2020年。2[44] Maria Tsimpoukelli、Jacob L Menick、Serkan Cabi、S. M. Ali Eslami、Oriol Vinyals 和 Felix Hill。利用冻结语言模型进行多模态少样本学习。载于《神经信息处理系统进展》，页码200–212。Curran Associates, Inc.，2021年。1、3[45]Feng Wang、M Cenk Gursoy 和 Senem Velipasalar。通信高效且保护隐私的基于特征的联邦迁移学习。载于2022年IEEE全球通信会议论文集，页码3875–3880，2022年。2[46] Xin Wang、Devinder Kumar、Nicolas Thome、Matthieu Cord 和 Fr´ed´eric Precioso。基于大型多模态食物数据集的食谱识别。载于2015年IEEE国际多媒体与博览会研讨会（ICMEW）论文集，页码1–6，2015年。2、6、3[47] Pei-Yau Weng、Minh Hoang、Lam M. Nguyen、My T. Thai、Tsui-Wei Weng 和 Trong Nghia Hoang。针对非独立同分布和不平衡数据的概率式联邦提示调优。载于第三十八届神经信息处理系统年会，2024年。1[48] Baochen Xiong、Xiaoshan Yang、Fan Qi 和 Changsheng Xu。多模态联邦学习的统一框架。神经计算，480：110–118，2022年。2[49] Baochen Xiong、Xiaoshan Yang、Yaguang Song、Yaowei Wang 和 Changsheng Xu。面向模态不完整的多模态联邦学习的客户端自适应跨模型重建网络。载于第31届ACM国际届多媒体国际会议，页码 1241–1249，2023年。3[50] Fu-En Yang、Chien-Yi Wang 和 Yu-Chiang Frank Wang。通过客户端特定提示生成实现联邦学习中的高效模型个性化。载于 IEEE/CVF国际计算机视觉会议（ICCV）论文集，页码 19159–19168，2023年。1[51] Guan Yu、Quefeng Li、Dinggang Shen 和 Yufeng Liu。无需插补的块缺失多模态数据最优稀疏线性预测。美国统计学会会刊，115(531):1406–1419，2020年。3[52] Songcan Yu、Junbo Wang、Walid Hussein 和 Patrick C.K. Hung。面向不完整模态的鲁棒多模态联邦学习。计算机通信，214:234–243，2024年。1, 3[53]Mikhail Yurochkin、Mayank Agarwal、Soumya Ghosh、Kristjan Greenewald、Trong Nghia Hoang 和 Yasaman Khazaeni。神经网络的贝叶斯非参数联邦学习。载于国际机器学习会议，2019年。2[54] Mikhail Yurochkin、Mayank Argawal、Soumya Ghosh、Kristjan Greenewald 和 Trong Nghia Hoang。通过参数匹配实现统计模型聚合。载于神经信息处理系统进展，页码 10954–10964，2019年。2[55] Changqing Zhang、Yajie Cui、Zongbo Han、Joey Tianyi Zhou、Huazhu Fu 和 Qinghua Hu。深度部分多视图学习。IEEE模式分析与机器智能汇刊，44(5):2402–2415，2020年。3[56] Jianyi Zhang、Hao Frank Yang、Ang Li、Xin Guo、Pu Wang、Haiming Wang、Yiran Chen 和 Hai Li。MLLM-FL：多模态大语言模型辅助的异构长尾数据联邦学习。arXiv预印本 arXiv:2409.06067，2024年。1, 2[57] Wei Zhang 和 Xiang Li。利用先验分布实现机械故障诊断中的数据隐私保护联邦迁移学习。结构健康监测，21(4):1329–1344，2022年。2[58] Haodong Zhao、Wei Du、Fangqi Li、Peixuan Li 和 Gongshen Liu。FedPrompt：联邦学习中的通信高效且保护隐私的提示调优。载于 ICASSP 会议论文集，2023年。1[59] Kaiyang Zhou、Jingkang Yang、Chen Change Loy 和 Ziwei Liu。面向视觉语言模型的条件提示学习。载于 IEEE/CVF计算机视觉与模式识别会议（CVPR），2022年。1, 3[60] Kaiyang Zhou、Jingkang Yang、Chen Change Loy 和 Ziwei Liu。学习如何为视觉语言模型设计提示。国际计算机视觉期刊（IJCV），2022年。1, 3[61] Tongxue Zhou、Pierre Vera、St´ephane Canu 和 Su Ruan。通过条件生成器和相关性学习实现多模态脑肿瘤分割中的缺失数据插补。模式识别快报，158:125–132，2022年。3

# 面向异构且不完整的多模态客户端数据的联邦提示调优

补充材料

## A. FED-PRIME 的伪代码

**算法 1** 客户端设计

**输入**：预训练模型 $F$、数据集 $D_t$，以及 $\boldsymbol{w}_g^{inter}$、$\boldsymbol{w}_g^{intra}$

**输出**：客户端内提示与客户端间提示 $\boldsymbol{w}_p^{intra}$、$\boldsymbol{w}_p^{inter}$

1：初始化预测头 $\boldsymbol{w}_c$

2：初始化 $\boldsymbol{w}_p^{intra}, \boldsymbol{w}_p^{inter} \leftarrow \boldsymbol{w}_g^{intra}, \boldsymbol{w}_g^{inter}$

3: 优化 $\boldsymbol{w}_c$，$\boldsymbol{w}_p^{intra}$，$\boldsymbol{w}_p^{inter}$使用公式 (7) 和公式 (8)

4: **返回** $\boldsymbol{w}_p^{intra}$ , $\boldsymbol{w}_p^{inter}$

**算法 2**服务器聚合

输入：一组客户端间提示$(\boldsymbol{w}_t^{inter} \triangleq (\boldsymbol{p}_t^i)_{i=1}^{\tau})_{t=1}^{n}$输出：优化后的聚合提示集 $\boldsymbol{w}_g^{inter}$1：初始化对齐模型参数 $\boldsymbol{\theta}$ 2：对于 $e = 1$至最大迭代次数，执行以下操作：3：固定 $\boldsymbol{\alpha}$，通过公式（13）优化 $\zeta$、$\gamma$、$\boldsymbol{\theta}$ 4：冻结 $\zeta$、$\gamma$、$\boldsymbol{\theta}$ 5：对于 $t = 1$至 $n$，执行以下操作：6：冻结 $\boldsymbol{\alpha}_{-t}$ 7：通过公式（14）优化 $\boldsymbol{\alpha}_t$ 8：结束循环 9：结束循环 10：$\boldsymbol{w}_g^{inter} \leftarrow \boldsymbol{\theta}$11：对于 $q = 1$ 至 $n \times \tau$，执行以下操作：12：$\boldsymbol{w}_g^{inter} \leftarrow \boldsymbol{w}_g^{inter} \setminus \boldsymbol{\theta}_q$ 如果 $\nexists(t, p)$：$\alpha^{p}{}_{,t}{}^{q} > 0$13：结束循环 14：返回聚合的客户端间提示集 $\boldsymbol{w}_g^{inter}$

**算法 3** 多模态联邦提示调优

**输入**：预训练模型 $F$ 和迭代次数 $T$

**输出**：优化后的聚合提示集 $\boldsymbol{\theta}$

1：初始化全局提示集 $\boldsymbol{\theta}$

2: **对于** $e = 1$**到** $T$ **执行**

3: **对于** $t = 1$ **到** $n$**执行**

4: 将 $\boldsymbol{\theta}$ 发送至客户端 $t$

5: 请求 $\boldsymbol{w}_t^{inter}$，$\boldsymbol{w}_t^{intra}$从客户端 $t$通过算法1

6: **结束循环**

7: 计算 $\boldsymbol{w}_g^{intra}$在（$\boldsymbol{w}_t^{intra}$）上运行联邦平均$_{t=1}^{n}$

8: 更新 $\boldsymbol{w}_g^{inter}$通过在算法 2 上 $(\boldsymbol{w}_t^{inter})_{t=1}^{n}$

9: **结束循环**

10: **返回** 聚合提示$\boldsymbol{w}_g^{inter}, \boldsymbol{w}_g^{intra}$

## B. 实现细节

**输入**。所有基线方法均使用来自 UPMC Food-101 数据集的输入，该数据集包含 6，728 个多模态样本；同时使用 MM-IMDB 数据集，该数据集经过预处理后包含 5，778 对图文数据。对于文本模态，输入使用 BERT-base-uncased 分词器进行分词，如 [24], 所述；UPMC Food-101 的最大序列长度为 40，MM-IMDB 的最大序列长度为 128。文本缺失时，我们使用空字符串作为虚拟输入。对于图像模态，我们遵循 [21, 24] ，将输入图像较短的一边调整为 384 像素，同时将较长的一边限制在 640 像素以内，以保持纵横比。如 [21], 所示，我们将图像分解为 $32 \times 32$ 个图像块。如果图像缺失，则按照 [24]中的描述，创建一个所有像素值均设为 1 的虚拟图像。

**多模态骨干网络**。 遵循 [24], ，我们采用预训练的多模态 Transformer ViLT [21]作为骨干网络，因为它广泛用于各种基于 Transformer 的多模态任务学习方法。ViLT 源自 Vi- sion Transformer [13]，并进一步发展为能够处理由词元化文本和图像块组成的多模态输入。ViLT 不使用特定于模态的特征提取器，而是在多个大型视觉语言数据集（例如 MS- COCO [28] 和 Visual Genome [22]）上通过图文匹配（ITM）和掩码语言建模（MTM）等目标进行预训练。

**模型训练细节**。 为减少对大量微调的需求，我们冻结 ViLT 骨干网络的参数，仅训练可学习提示和特定于任务的层（池化器和分类器）。每个池包含 20 个提示，每次输入从每个池（客户端间池和客户端内池）中选取 5 个提示。随后将这些提示拼接起来，并添加到初始的多头自注意力（MSA）层，因此每次输入总共包含 10 个提示。对于 FED-INTRA 和 FED-INTER，由于仅使用单个池，该池仍包含 20 个提示。从该池中直接选取 10 个提示，确保每次输入的提示总数与 FED-PRIME 保持一致。相比之下，FEDAVG-P 和 FEDMSPLIT-P 将提示附加到前 6 个 MSA 层，每个提示的长度为 16，因此每次输入包含的提示数量更多。

**基线方法聚合细节**。在 FEDAVG-P 中，服务器聚合流程会为每种模态组合更新新训练的组件——即提示、池化器和分类器——随后将其分发给相应的客户端。在 FEDMSPLIT-P 中，原始工作假

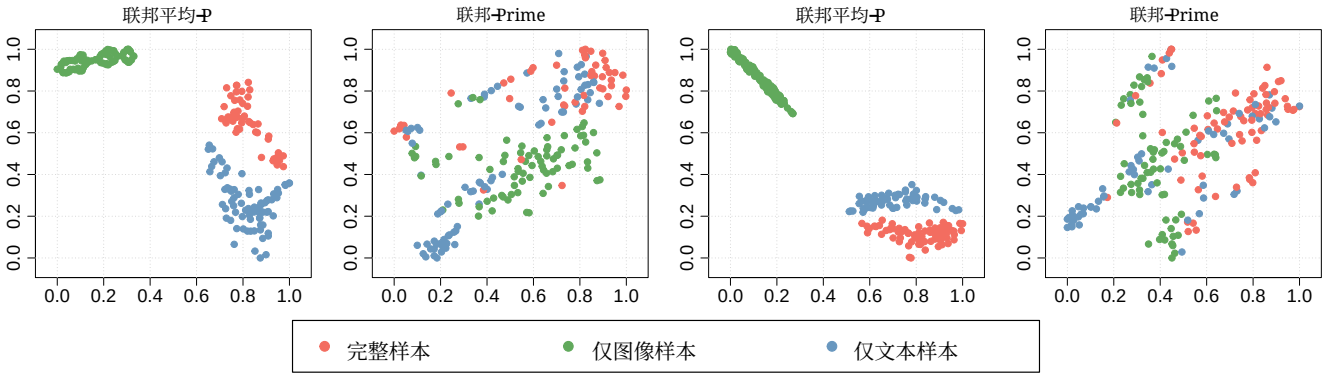


图7. 在 MM-IMDB 的 **两者均缺失** 训练场景下，分类前嵌入的 t-SNE 图：客户端 #4（左）和客户端 #14（右），每个客户端包含两个子图。

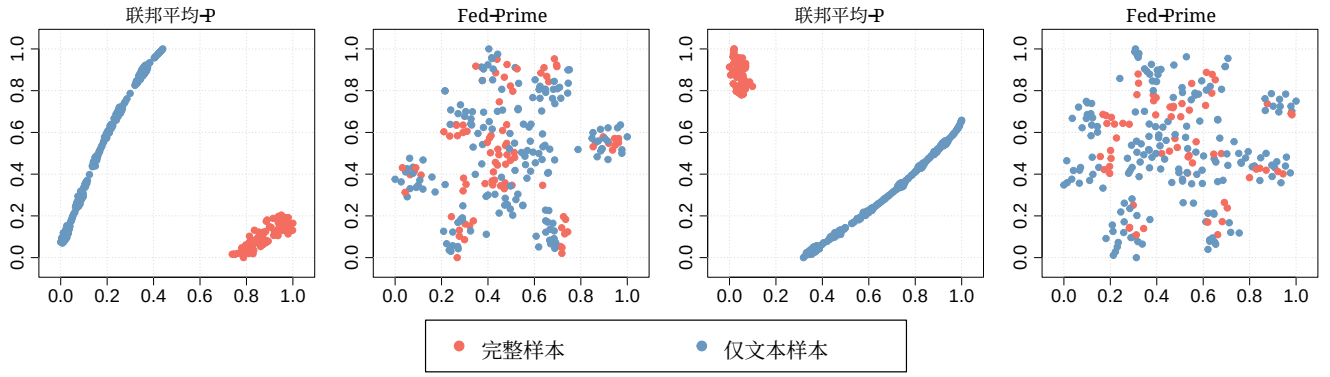


图8。t-SNE 嵌入的 t-SNE 图우우우우우 分类前嵌入的 t-SNE 图，展示了 UMPC Food-101 在 **图像缺失** 训练场景下客户端 #4（左 t) 以及客户端 #14（右侧），每个包含两个 客户端子图。

表2。在 UPMC Food-101 数据集上报告的不同池大小下 FED-PRIME 的 Top-1 准确率。

| **池大小** | 10 | 15 | 20 | 25 | 30 |
|---|---|---|---|---|---|
| **Top-1 测试准确率** | 73.94 | 82.50 | 82.61 | 82.72 | 83.21 |

设每个客户端拥有一组独特的模态，并且给定客户端内所有样本的模态分布均匀。这一假设与我们的设定不同：客户端呈现出异质且不完整的多模态数据分布。此外，原始框架会构建模态特定编码器块，并依据量化客户端间模态关系的相似度系数，对相似的编码器块进行聚合。为使该方法适用于我们的设定，我们基于客户端相似度聚合所有模块化组件——包括提示、池化器和分类器——同时假设所有客户端都保有一套共享的此类模块化组件，涵盖所有可能的模态配置。

**超参数设置**。所有评估的基线方法在 UPMC Food-101 数据集上均使用 512 的批量大小，在 MM-IMDB 上均使用 256 的批量大小。我们将 UPMC Food-101 和 MM-IMDB 的训练数据集随机划分为 20 个客户端，所有客户端均参与每一通信轮次。我们使用随机梯度下降（SGD）优化器，Food-101 的基础学习率为 0.05，IMDB 的基础学习率为 0.01，并设置 250 个通信轮次，以比全量微调更快地收敛。每轮中，客户端在本地训练全局模型 1 个训练轮次。所有实验缺失场景均一致采用这些超参数配置。

**代码可用性与可复现性**。 我们在 https://github.com/hangpt01/FedPrime 发布了完整的实现和配置。代码仓库包括源代码、配置文件（例如提示维度、提示数量、聚类阈值、正则化系数），以及简要指南和脚本，用于运行实验或将框架应用于其他数据集。

**自动超参数调优**。 除上述默认设置外，该代码仓库还包含用于自动调优关键超参数的工具与脚本（例如聚类迭代阈值、池大小以及

表3. 每轮不同基线方法所需的资源

| 方法 | UPMC Food-101 [46] | | MM-IMDB [1] | |
|---|---|---|---|---|
| | GPU（GB） | 时间（秒） | GPU（GB） | 时间（秒） |
| FEDAVG-P | 31.70 | 245.53 | 39.07 | 649.58 |
| FEDMSPLIT-P | 34.30 | 265.19 | 39.13 | 755.62 |
| FED-INTER | 16.83 | 315.67 | 23.73 | 685.27 |
| FED-INTRA | 16.65 | 253.60 | 23.31 | 625.83 |
| **FED-PRIME** | 16.80 | 240.54 | 23.68 | 653.57 |

活跃提示）。我们还提供常见调优方法的实现，即网格搜索、随机搜索和贝叶斯优化，使用户能够基于验证损失或启发式规则高效探索配置。这些工具减少了大量手动搜索的需求，并简化了对新问题的适配。

inst

## C. 计算资源分析。

表 3总结了评估方法所需的计算资源。值得注意的是，FEDAVG-P 和FEDMSPLIT-P的 GPU 内存使用量显著更高，例如在 UPMC Food-101 数据集上分别达到 31.70 GB 和 34.30 GB，几乎是FED-PRIME的两倍。这种差异可归因于它们在总共 12 层的网络中，选择在前六个多头自注意力（MSA）层中添加提示，从而扩展序列长度，并增加中间激活、梯度和计算所需的内存。此外，FEDMSPLIT-P 进一步增加了计算成本，因为它需要估算客户端之间的相似度，并为每个客户端维护一个个性化模型。相反，FED-PRIME仅在第一个 MSA 层中添加提示，从而将扩展后的序列长度及相关计算限制在单个层内。这种设计显著降低了总体内存开销。其他变体的 GPU 内存使用量相对接近。总体而言，MM-IMDB 数据集的 GPU 使用量高于 UPMC Food-101 数据集，这主要是因为实验中采用了更长的文本序列。

就每轮执行时间而言，FED-INTER 和 FEDMSPLIT-P 交替表现出最长的运行时间。其他方法的运行时间性能相近，FEDAVG-P 与 FED-PRIME 之间约相差 5 秒。值得注意的是，FED-PRIME 在实现高性能的同时，所需 GPU 内存更少，且执行时间相当，充分展现了其效率。

## D. 更多消融研究

表2 清楚地表明，池大小与测试准确率呈正相关，池越大，准确率就越高。当池大小从 15降至10时，准确率显著下降，凸显了维持性能所需的关键阈值。将池大小从15 增至 30 虽能整体提升准确率，但随着池大小从 15 增至25，提升幅度逐渐减小。值得注意的是，池大小为 30 时准确率最高，提升幅度大于池大小从 15 增至 25时观察到的较小增益。总的来说，尽管增大池大小能够提高准确率，但其边际收益可能不足以抵消额外的计算成本。因此，本研究后续所有实验均选用池大小 20 ，以在准确率和计算效率之间实现实用的平衡。

## E. 更多提示分析

图7 展示了在 MM-IMDB 数据集的两者均缺失场景下，FEDAVG-P和**FED-PRIME** 在最终轮次（第 250 轮）的嵌入，并对两个随机客户端（客户端 #4 和客户端 #14）进行了可视化。如图所示，FEDAVG-P采用针对每种缺失类型分别为嵌入添加提示的设计，因此在嵌入空间中为每种样本类型形成了不同的簇。相比之下，FED-PRIME 生成的嵌入更加分散，其中完整样本的嵌入位置更接近模态缺失样本的嵌入。这表明，无论样本中具体缺失的是哪种模态，FED-PRIME都更擅长学习有意义且全面的表征。值得注意的是，FEDAVG-P 嵌入中的纯图像样本与纯文本样本和完整样本相距甚远。这种分离凸显了FEDAVG-P 在有效捕捉跨模态关系方面的局限性，这可能是因为该方法使用了针对特定缺失模态类型的提示。这进一步支持了以下假设：FED-PRIME 中的 ViLT 骨干网络具有更强的文本表征和整合能力。

为了进一步探索这些模型，图 8展示了相同客户端在 UMPC Food-101 数据集的 **图像缺失** 训练场景下的嵌入。观察到类似趋势： FED-PRIME生成的嵌入能够良好融合不同样本类型的表示，而FEDAVG-P 则产生了两个明显分离的簇。与此前观察到的情况相同，当数据中包含仅含图像的样本时，FEDAVG-P 中的文本样本和完整样本的嵌入可能会聚集得更近。然而，即使不存在仅含图像的样本，文本样本和完整样本的嵌入仍然过于分离，这凸显出

不同模态之间缺乏连贯的表示。这种分离可能会损害 FEDAVG-P 的性能，导致后续将分类器应用于这些嵌入时，其结果表现平平。

## F. 额外实验结果

**池间收敛**。 图 9和图 10展示了两个数据集在三种预定义训练场景下，客户端间提示池的池大小变化。结果表明，提示相对于其质心的分散程度先增加，随后又逐渐降低。这一观察说明，所提出的方法能够优化学到的提示，从而有效捕捉参与客户端之间的数据多样性。在模型充分学会聚类和对齐后，它开始整合客户端特定信息，进而使整体池大小增加。

**大规模联邦学习的可扩展性**。 我们在 Food-101 数据集上进行了额外实验，模拟包含 100 个客户端、客户端参与率在 10% 至 50% 之间变化的场景。我们将结果与该数据集上表现最强的基线方法 FED MSPLIT-P 进行了比较。如表 **??**所示，我们的双提示机制展现出良好的可扩展性，性能随着参与客户端数量的增加而提升。此外，在测试准确率方面，我们的方法始终优于基线方法。在计算效率方面，FED-PRIME 的 GPU 内存消耗显著更低，且不受客户端数量影响，始终保持稳定；相比之下，随着客户端数量增加，FEDMSPLIT-P 的内存使用量也会增加。这一特性使 FED-PRIME 尤其适合部署在资源受限的环境中。

**高度不平衡数据下的性能**。除了模态缺失异质性外，我们还在极端数据不平衡情况下评估了我们的方法。FedProx [26],是一种专为解决非独立同分布问题而设计的流行基线方法，因此被选作该设置下的比较对象。具体而言，我们将我们的方法与一种变体进行比较，该变体使用 FedProx 修改分类器和客户端内提示的平均方式，记为 FEDPROX-P。我们通过模拟狄利克雷分布，在极端非独立同分布设置下对 Food-101 数据集进行了额外评估，参数为$\alpha = 0.1$。表 4 中的结果证实，我们的方法始终优于 FEDPROX-P，进一步表明聚合客户端间提示能够有效应对极端数据不平衡。尽管 FEDPROX-P旨在解决数据异质性，但我们的方法通过使用客户端间提示有效缓解了这一问题，凸显了我们的提示对齐算法以及针对特定模型组件采用的选择性平均策略的优势。我们还评估了以下情形：客户端拥有全模态或仅拥有单一模态，我们的方法在准确率上均优于 FEDAVG-P及其集中式对应方法（见图1 和图4）。

表 4. UPMC Food-101 数据集上的非独立同分布联邦学习设置。用短横线 (-) 表示的结果代表**测试（双缺失）**与**测试（∼训练）**相同的情况。

| 训练 | 方法 | Test (∼ 训练) | Test (两者均缺失) | Test (全模态) | Test (仅文本) | Test (仅图像) |
|---|---|---|---|---|---|---|
| Miss Text | FEDPROX-P | 67.42 | 64.34 | 77.29 | 56.83 | 68.24 |
| | **FED-PRIME** | **71.20** | **71.15** | **85.08** | **63.91** | **69.56** |
| Miss 图像 | FEDPROX-P | 82.56 | 71.26 | 85.24 | 83.05 | 45.31 |
| | **FED-PRIME** | **87.38** | **75.59** | **89.25** | **87.05** | **48.47** |
| Miss Both | FEDPROX-P | 75.75 | - | 89.36 | 83.98 | 69.61 |
| | **FED-PRIME** | **79.98** | - | **91.00** | **86.70** | **70.38** |

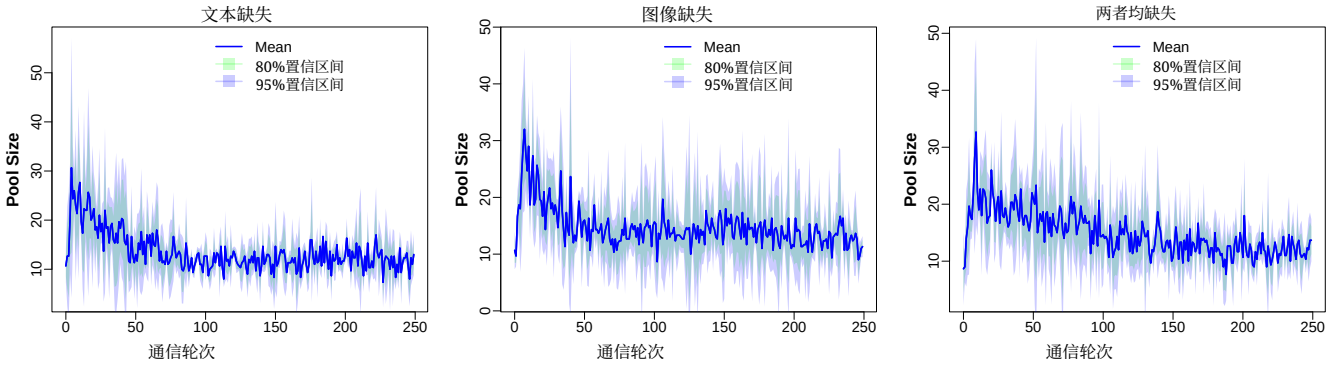


图9. UMPC F 的变化　　在不同训练场景下，ood-101 的客户端间提示池大小在 250 个通信轮次中的变化　　s.

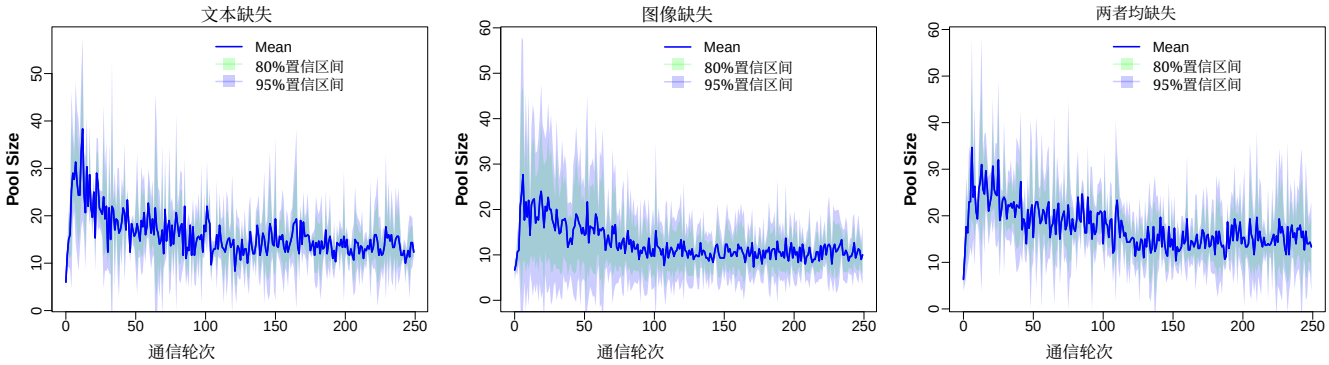


图10. MM-I 的变化　　不同训练场景下，MDB 客户端间提示池大小在 250 个通信轮次中的变化。