多模态物联网系统的混合联邦学习复现

摘要

多模态联邦学习在物联网场景下具有广泛的应用，然而现有的多模态学习方法常将物联网设备作为数据的收集器而忽略了物联网设备本身具有的计算资源，同时传统的多模态方法仅将多模态数据作为具有更高维度信息的单模态数据进行处理，并未真正解决多模态联邦学习的挑战，本文通过提出在边缘设备进行本地单模态训练，在边缘服务器进行多模态纵向联邦学习，在全局服务器进行横向联邦学习的混合联邦学习架构 HFM 来解决上述问题。通过复现混合联邦学习、纵向联邦学习、横向联邦学习和本地多模态学习在对 Cifar-100 数据集的图像分类任务并评估其效果可以得出结论，即 HFM 性能更好且更适用于多模态物联网的下游推理任务。

关键词：联邦学习；多模态物联网；边缘计算

1 引言

物联网将大量设备和传感器连接在一起，促进了各个英语的无缝数据交换和自动化。在多模态物联网领域，这些物联网设备具有捕获与同一样本 [1] 对应的各种数据类型的能力。在边缘服务器中，每个物联网设备可能包含一个或多个传感器，负责收集不同的数据类型。这些物联网设备将多模态数据上传到边缘服务器中，进行特征提取和下游推理任务。与单模态数据相比，多模态数据通常具有更全面的互补特征，从而提高下游推理性能。

联邦学习能够在保证数据隐私的同时扩大模型训练的样本空间 [2]。这种分布式学习范式使得设备能够协同训练模型，而无需在中央服务器聚合敏感数据。在以数据异构和隐私保护问题为特征的多模态物联网领域中联邦学习具有巨大的应用前景 [3]。但是在现有的大多数多模态物联网场景中，由于边缘服务器有限的计算资源有限，尽管大多数物联网设备具有计算能力，实际应用中仍然仅将它们视为数据收集的传感器。此外，目前现有的多模态物联网方法都是将多模态输入视为具有更丰富特征和更高维度的“单模态”输入，无法解决多模态联邦学习 [4] 的根本挑战。

为了使分布式物联网设备不仅可以作为手机信息的传感器，还可以作为边缘计算设备进行特征提取和下游推理，从而减轻边缘服务器的计算负担。这篇论文 [5] 提出了 HFM 混合联邦学习算法。HFM 在特征空间上利用纵向联邦学习分配计算资源，在样本空间上利用横向联邦学习分配计算资源。通过将两种联邦学习范式混合的方式将边缘计算设备计算资源解放出来，同时避免了将多模态输入简单的融合应用于多模态学习的问题。

本文主要工作概述如下：

- 混合联邦学习（HFM）：在边缘计算设备上进行本地单模态训练，在边缘服务器上做多模态纵向联邦学习，在全局服务器上做多模态联邦学习。通过分层的方式将联邦学习进行混合，释放边缘设备计算潜力。
- 设计算法对比试验：分别对混合联邦学习（HFM）、纵向联邦学习（VFL）、横向联邦学习（HFL）和本地多模态学习（Local）对同一任务的效果进行评估。

2 相关工作

多模态学习在物联网场景下常有如图 1(a) 所示的工作场景，在这个工作场景下模型利用物联网设备收集的多模态数据通过一定的编码方式进行聚合，以此完成多模态学习。但这样的多模态学习并没有考虑到隐私保护，因此当多模态学习同联邦学习整合在一便有了如图 1(b) 和图 1(c) 所示的多模态横向联邦学习和多模态纵向联邦学习，然而这样的融合尽管考虑到了隐私保护，仍有一些其他方面的问题。

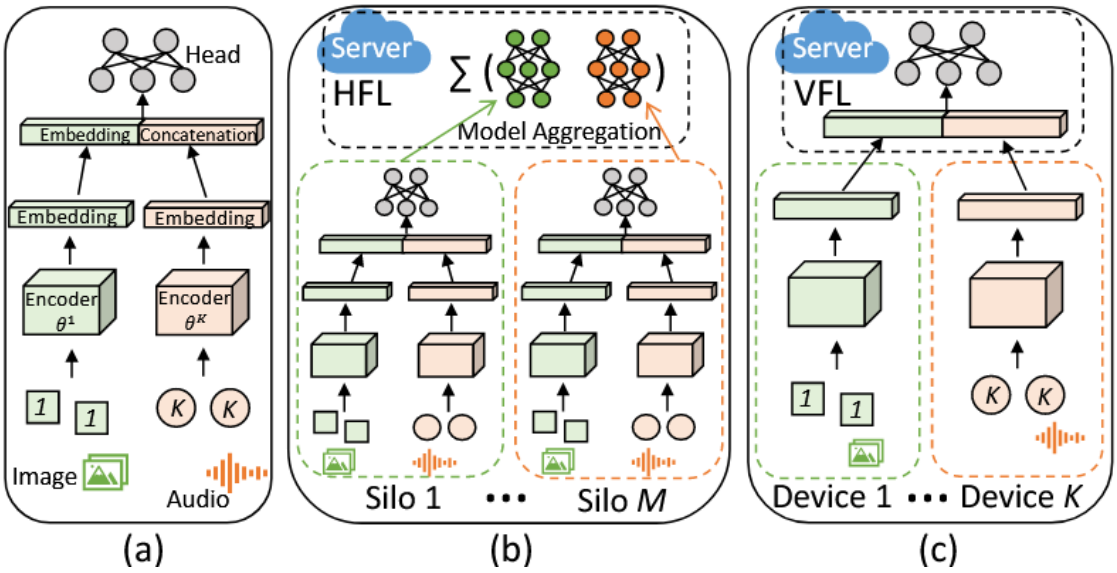


图 1. (a) 本地多模态学习。(b) 横向联邦多模态学习。(c) 纵向联邦多模态学习。

2.1 横向联邦学习

横向联邦学习（HFL）适用于各参与方特征空间重叠但样本空间不重叠的应用场景。如两家不同地区的银行，它们的用户群体来源于不同地区的群众，但他们的业务很相似，因此记录的用户特征也是相似的。在物联网场景中，各水平数据孤岛样本数量有限但对样本采集的数据是相同模态的，那么可以使用横向联邦学习提高下游推理性能 [6]。当多模态引入物联网横向联邦学习，那么其具体的工作架构如图 1(b) 所示。这样的工作架构实际上是将多模态输入视为具有更丰富特征和更高维度的单模态输入，无法完全利用多模态数据之间互补的特性。

2.2 纵向联邦学习

纵向联邦学习（VFL）适用于各参与方样本空间重叠但特征空间不重叠的应用场景。如同一地区的两种机构，它们的用户群体来源于同一地区的群众，但他们的业务不相似，因此各自记录的用户特征也不同。在物联网场景中，各垂直数据孤岛可能对同一批样本采集数据，不同数据孤岛持有同一批样本的不同模态数据 [7]。物联网场景下的纵向联邦学习具体的工作架构如图 1(c) 所示，这样的工作架构尽管考虑到了多模态数据互补的特性，然而由于垂直数据孤岛是对同一批样本采集数据，因此可能会由于数据样本过少而无法得到理想的神经网络模型。在多模态物联网场景下，样本空间和特征空间往往是同时被划分开的，单独使用横向联邦学习和纵向联邦学习都不足以解决多模态物联网所带来的挑战。

3 本文方法

3.1 本文方法概述

论文 [8] 提出了混合联邦学习框架（HFM）来解决多模态物联网场景下样本空间和特征空间同时被划分开的问题。HFM 框架包含由 M 个筒仓组成的多模态物联网系统，各筒仓对不同的样本进行数据采集，单个筒仓内包含 K 个物联网设备，这些物联网设备对同一批样本采集不同模态的数据。筒仓间通过全局服务器交流，全局服务器上部署有全局横向联邦模型 Θ，筒仓内物联网设备通过边缘服务器交流，第 m 筒仓边缘服务器上部署有筒仓纵向联邦模型 θ_m^0，且第 k 个物联网设备持有数据 x_m^k 和标签 y_m 并部署有本地模型 θ_m^k。HFM 框架是在筒仓内每经过 Q 轮次对物联网设备采集的多模态数据进行纵向联邦学习并在选择小批量数据集 $\mathcal{B}_m$ 作为纵向联邦学习训练数据集，在各筒仓间每经过 RQ 轮次进行横向联邦学习。具体工作过程如图 2所示：

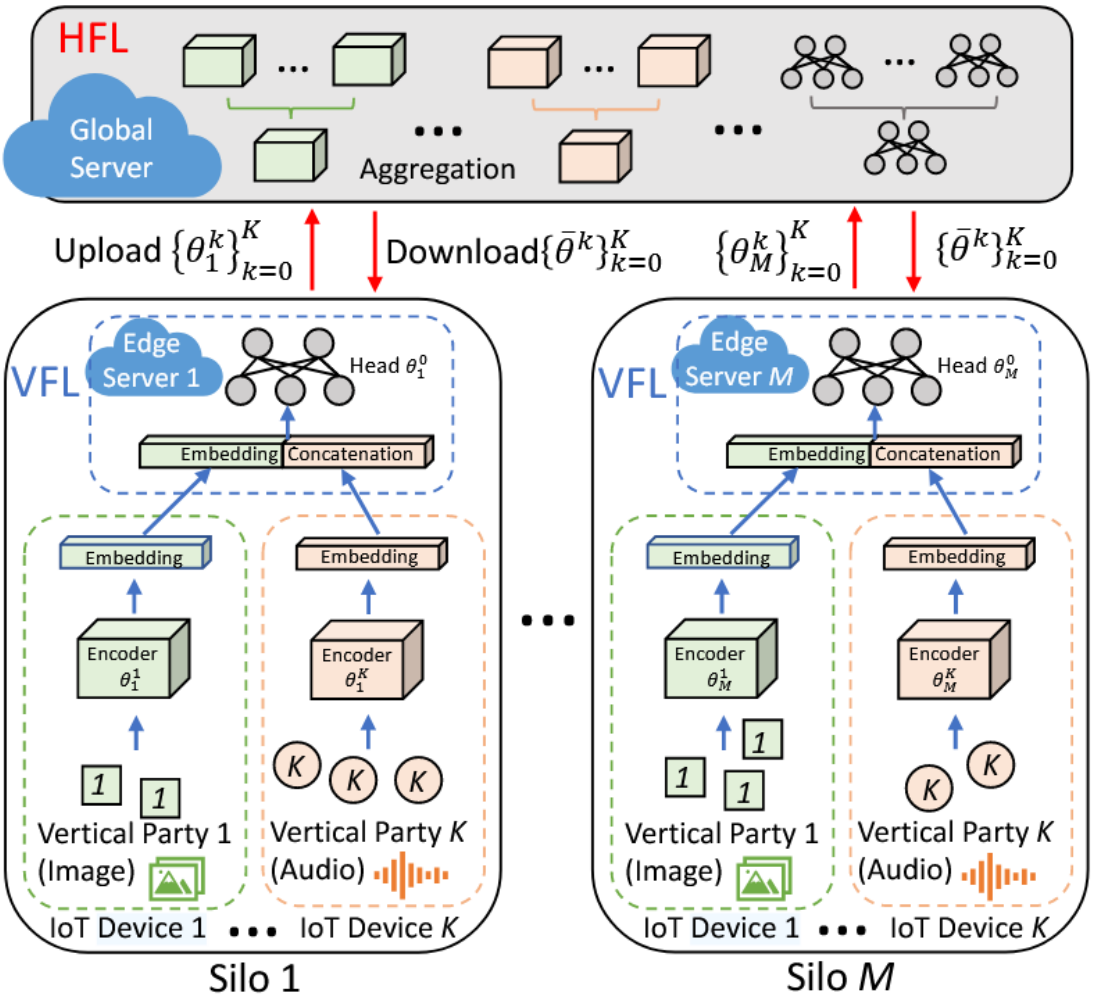


图 2. 混合联邦学习工作流程

3.2 纵向联邦学习

在第 m 筒仓内的每个 VFL 轮次开始时，从数据集 x_m 内随机采样小批量数据集 $\mathcal{B}_m$ 作为该 VFL 轮筒仓内数据集，筒仓内各物联网设备以此进行训练。在 VFL 轮次开始时，边缘服务器接受来自各物联网设备对小批量数据集 $\mathcal{B}_m$ 生成的嵌入集 $h_m^k(\theta_m^{k,t}; x_m^{k,\mathcal{B}_m^{t_0}})$ 并将嵌入集连同边缘服务器纵向联邦模型组合起来作为 $\Phi_m^{t_0}$ 下发到各物联网设备。物联网设备更新纵向联邦学习模块并在训练中根据自身模型生成嵌入以及其余模型的嵌入集组合训练本地网络模型。梯度更新算法如下：

$$\theta_m^{k,t+1} = \theta_m^{k,t} - \eta \nabla_k f_{\mathcal{B}^{t_0}}(\Phi_m^{k,t}, Y_m^k), \tag{1}$$

其中 $\Phi_m^{k,t}$ 为更新了第 k 个物联网设备本地模型嵌入的嵌入集，这样的更新会包含上一轮其余物联网设备嵌入的陈旧信息，但最终并不会影响模型收敛。因此在纵向联邦学习中，物联网设备本地模型梯度计算算法可以为：

$$\nabla_k f_{\mathcal{B}^{t_0}} \Phi_m^{k,t}, Y_m^k := \frac{1}{\mathcal{B}_m} \sum_{i \in \mathcal{B}_m} \nabla_{\theta_m^k} \mathcal{L}[\theta_m^0, h_m^1(\theta_m^1; x_m^{1,i}), \ldots, h_m^K(\theta_m^K; x_m^{K,i}); y_m^i] \tag{2}$$

3.3 横向联邦学习

各筒仓所持有的纵向联邦学习全局模型结构是一致的，因此可以在此采用横向联邦学习的方式将各筒仓模型联合在一起。在第 HFL 轮次开始时全局服务器接受各筒仓纵向联邦学习模型，将各个使用联邦平均的方式聚合在一起并下发到各筒仓边缘服务器。边缘服务器同样将聚合后的纵向联邦学习模型下发到物联网设备中，并进行各自的纵向联邦学习。全局服务器聚合边缘服务器模型算法如下：

$$\theta^{k,t} = \frac{1}{N} \sum_{m=1}^{M} N_m \theta_m^{k,t} \tag{3}$$

其中 N 为整个系统中样本总量，N_M 为第 m 个筒仓包含的样本数量，因此边缘服务器模型聚合权重同边缘服务器持有的样本数量挂钩。HFL 轮次中包含了多个 VFL 轮次，具体根据 Q 和 RQ 的设置而定，根据原论文可知，Q 和 RQ 的设定对 HFM 架构效果同样存在一定的影响。

4 复现细节

4.1 与已有开源代码对比

本次复现实现的代码未参考任何其他人发布的代码,在复现中原生了一套包括对 MNIST、MNIST-M、ModelNet10、Cifar-100 等数据集进行处理的数据集处理类,并在模型方面将 Resnet18 进行简化，同时对本地机器学习训练过程（Local）、横向联邦学习训练过程（HFL）、纵向联邦学习训练过程（VFL）以及混合联邦学习训练过程（HFM）原生出一套可复用的代码架构。

4.2 实验环境搭建

代码使用 pytorch==2.4.1、torchvision=0.19.1 的 pytorch 版本，运行在单张 Nvidia 3090 服务器显卡上，Python 版本为 3.10。代码目录结构最高级有 src、data、result，最高级目录下存在运行批命令文件 run.sh，运行 run.sh 将进行对 Cifar-100 数据集进行图像分类的 HFM 框架实验以及三种对比试验：本地框架、横向联邦学习框架、纵向联邦学习框架。

4.3 创新点

在文章中对 ModelNet40 数据集进行处理，获得同一 3d 云图不同视图的图像并以此作为多模态数据进行图像分类，由此得到启发，对同样一批数据的不同处理在一定程度上可以认为是多模态数据并由此避免了多模态数据对齐的问题。因此在复现时选择对 Cifar-100 数据集进行处理，将数据集分为 RGB 图像和灰度图像两种模态，并以此进行图像分类处理，同时由于 Cifar-100 数据集中图像较小，使用常规的 Resnet18 模型对齐进行数据处理很容易出现过拟合的情况，因此对 Resnet18 模型进行简化，将其转换为能够提取 64 长度特征向量的特征提取器。

文章并没有给出在筒仓中纵向联邦学习是如何实现的。根据对原论文算法的研究，在纵向联邦学习中，边缘服务器收集各设备对各设备各自模态数据的嵌入，并将这种嵌入汇总连同边缘服务器持有的模型下发到设备中，设备借助本地模型和边缘服务器持有的模型利用其

余模型的嵌入进行训练，因此在图像分类任务中，可以认为边缘服务器持有能够接受多模态嵌入的分类器头、设备持有特征提取器，在训练中设备将本身产生嵌入同其余设备嵌入对齐输入边缘服务器下发的分类器头并以此更新分类器头和本地模型，边缘服务器将分类器头进行聚合以此完成纵向联邦学习。

在具体实验中发现混合联邦学习框架下，测试集损失值在后期不降反升，出现一定的过拟合现象。考虑到混合联邦学习在要求纵向联邦学习和横向联邦学习的情况下，其本地训练总轮次比其余对比试验高出一个量级，因此需要采用一定的方法抑制这种本地训练轮次过高导致的过拟合，因此引入 FedProx [9] 作为改良。

FedProx 在本地训练阶段引入了一个近端项到客户端的损失函数中，这个近端项用于衡量本地模型更新同全局模型之间的距离，约束本地模型不至于向极端的方向更新。本地训练阶段 FedProx 损失函数定义为

$$\mathcal{L}(\omega) = \sum_{k=1}^{K} \frac{n_k}{n} (\mathcal{F}_k(\omega) + \frac{\mu}{2} |\omega - \omega^t|^2) \tag{4}$$

其中 $\mathcal{F}_k(\omega)$ 为客户端本地损失函数，ω^t 为全局模型参数。

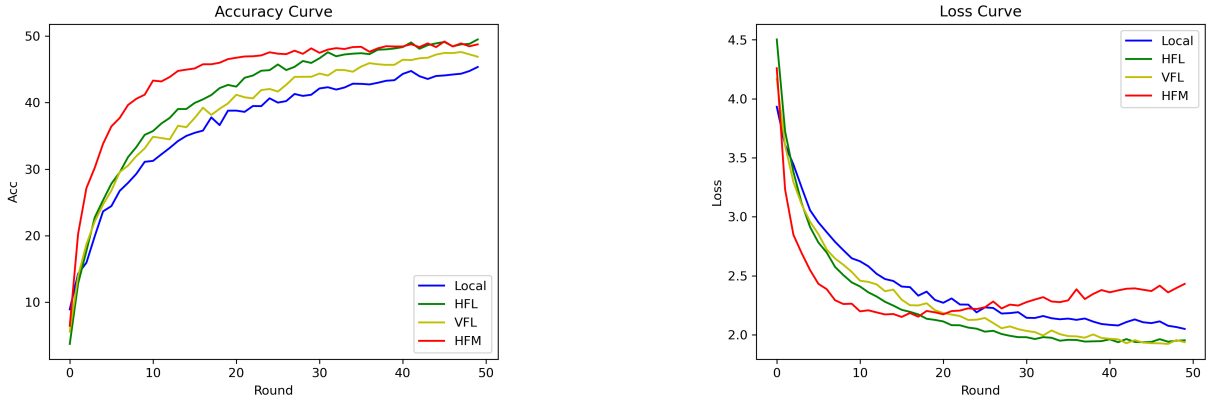
5 实验结果分析

5.1 实验设置

本文实验设置 HFM 实验组和 HFL、VFL、Local 对照组。实验对 Cifar-100 数据集进行图像分类训练，共训练 50 轮次，观察准确率和损失值。四组实验使用同样的 Resnet18 模型，损失函数统一为交叉熵损失函数，Cifar-100 数据集则将其分成彩色图像和灰度图像作为两组天然对齐的多模态数据。

5.2 评估 HFM

在 Local 和 HFL 框架中，本地模型都是将将多模态的输入同时输入模型作为一种单模态的数据，在 HFL 中利用 FedAvg 在边缘服务器进行聚合。在 VFL 中将多模态的数据分开，在客户端训练各自模态的特征编码器，并在服务器利用 FedAvg 聚合分类器头。在 HFM 框架中将小批次的数据集大小固定为整个数据集的十分之一，客户端每进行 5 轮本地训练则利用纵向学习在边缘服务器聚合一次，边缘服务器每进行两次聚合则利用横向学习在全局服务器利用横向学习聚合一次，即 $RQ = 10$ 且 $Q = 5$。



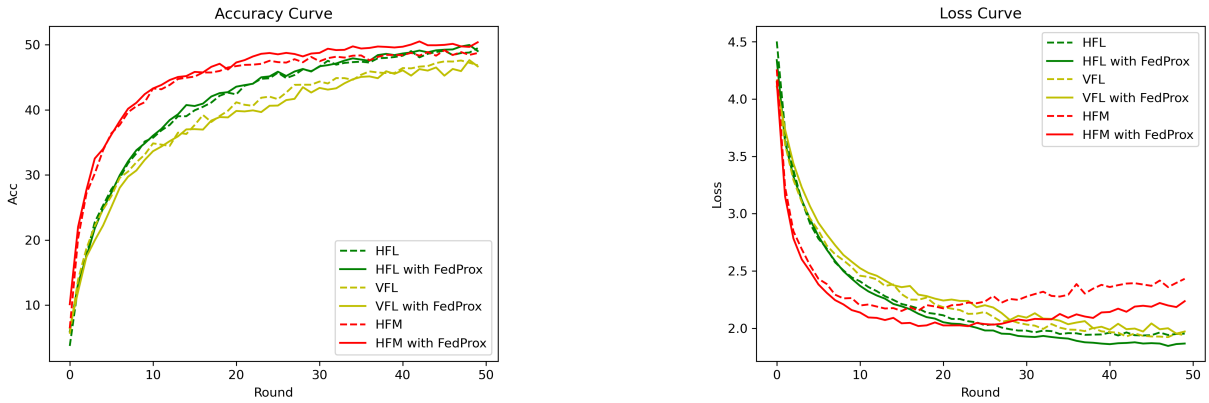
(a) 准确率　　(b) 损失值

图 3. HFM 与其他基线的准确率和损失值

如图3(a) 所示，可以看到 HFM 框架中尽管使用了陈旧信息，但仍不影响模型收敛。HFM 同 HFL 框架相比，在解放了物联网设备计算资源且没有将多模态数据进行简单拼接的同时没有明显的性能损失。在同样保留多模态各自特性的 VFL 框架相比较，HFM 又有着很高的性能提升。然而如3(b) 所示，HFM 框架出现过拟合现象，在训练后期测试集损失值随训练的深入逐渐上升，这可能是由于本地训练轮次过高所导致的，然而这种本地训练轮次过高是 HFM 框架所注定的，因此需要寻找别的方式来尽量减少过拟合现象的发生。

5.3 引入 FedProx

在复现中，选择引入 FedProx 来抑制过拟合现象。由于 FedProx 是针对联邦学习框架设计的，对本地机器学习无法应用，因此在这里仅设置 HFL、VFL 和 HFM 三组实验并同原先未引入 FedProx 的实验结果进行比较。如图4(a) 所示，引入 FedProx 后 HFL 和 HFM 相较原先有一定的性能提升。然而，如图4(b) 所示，引入 FedProx 后 HFM 过拟合现象虽然相较原先被抑制了，但训练后期测试集损失值仍有一定上升趋势。



(a) 准确率　　(b) 损失值

图 4. 引入 FedProx 后 HFM 与其他基线的准确率和损失值

6 总结与展望

本文介绍了一种在特征空间使用纵向联邦学习，在样本空间使用横向联邦学习的混合联邦学习架构（HFM）。其旨在解放物联网场景中物联网设备的计算资源并在根本上完成多模态信息的分割和合作。同时本文对 HFM 框架进行了复现并在其之上融入 FedProx 进行优化。从复现结果上来看，本文得出了同原文一致的结论，即 HFM 框架性能优于 Local、HFL 和 VFL，且更适用于需要快速准确的下游推理任务的多模态物联网系统。

本次论文复现过程让我对联邦学习和多模态学习有个更加全面的了解。不仅了解了横向联邦学习和纵向联邦学习各自的代码框架及实践特点，也让我了解到了如何对模型超参数进行调优，训练过程中各种常见问题如何解决。在复现中，为了解决过拟合现象尽管尝试了 FedProx 算法，但虽然有性能提升，但过拟合现象仍未完全解决，因此后续将对此做出更多的探索。

参考文献

[1] Amit Kumar Singh, Deepa Kundur, and Mauro Conti. Introduction to the special issue on integrity of multimedia and multimodal data in internet of things. *ACM Trans. Multimedia Comput. Commun. Appl.*, 20(6), March 2024.

[2] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. 10(2), January 2019.

[3] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M. Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2022.

[4] Baochen Xiong, Xiaoshan Yang, Fan Qi, and Changsheng Xu. A unified framework for multi-modal federated learning. *Neurocomputing*, 480:110–118, 2022.

[5] Yuanzhe Peng, Yusen Wu, Jieming Bian, and Jie Xu. Hybrid federated learning for multimodal iot systems. *IEEE Internet of Things Journal*, 11(21):34055–34064, 2024.

[6] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M. Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2022.

[7] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M. Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2022.

[8] Yuanzhe Peng, Yusen Wu, Jieming Bian, and Jie Xu. Hybrid federated learning for multimodal iot systems. *IEEE Internet of Things Journal*, 11(21):34055–34064, 2024.

[9] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.