ColNet: Collaborative Optimization in Decentralized Federated Multi-task Learning Systems

Chao Feng[1], Nicolas Fazli Kohler[1], Zhi Wang[1], Weijie Niu[1], Alberto Huertas Celdrán[1,2], Gérôme Bovet[3], Burkhard Stiller[1]

[1]Communication Systems Group, Department of Informatics, University of Zurich, 8050 Zürich, Switzerland
[cfeng, niu, huertas, stiller]@ifi.uzh.ch, [nicolasfazli.kohler, zhi.wang]@uzh.ch

[2]Department of Information and Communications Engineering, University of Murcia, 30100–Murcia, Spain

[3]Cyber-Defence Campus, armasuisse Science & Technology, 3602 Thun, Switzerland gerome.bovet@armasuisse.ch

***Abstract*—The integration of Federated Learning (FL) and Multi-Task Learning (MTL) has been explored to address client heterogeneity, with Federated Multi-Task Learning (FMTL) treating each client as a distinct task. However, most existing research focuses on data heterogeneity (e.g., addressing non-IID data) rather than task heterogeneity, where clients solve fundamentally different tasks. Additionally, much of the work relies on centralized settings with a server managing the federation, leaving the more challenging domain of decentralized FMTL largely unexplored. Thus, this work bridges this gap by proposing *ColNet*, a framework designed for heterogeneous tasks in decentralized federated environments.**

***ColNet* partitions models into a backbone and task-specific heads, and uses adaptive clustering based on model and data sensitivity to form task-coherent client groups. Backbones are averaged within groups, and group leaders perform hyper-conflict-averse cross-group aggregation. Across datasets and federations, *ColNet* outperforms competing schemes under label and task heterogeneity and shows robustness to poisoning attacks.**

I. Introduction

The integration of Federated Learning (FL) and Multi-Task Learning (MTL) has gained significant attention for its ability to tackle diverse but related tasks in a collaborative and privacy-preserving manner. FL trains models across clients by exchanging model updates rather than raw data, preserving data privacy [1]. Decentralized FL (DFL) eliminates the need for a central server, enhancing system robustness by avoiding single points of failure [2]. MTL, traditionally performed on a single machine, leverages task interdependence to improve generalization, reduce overfitting, and address data sparsity [3], [4]. Extending MTL to federated settings results in Federated Multi-Task Learning (FMTL) [5], combining FL's privacy preservation with MTL's collaborative task-sharing benefits.

In both centralized and decentralized FMTL, existing research primarily addresses the challenge of non-independent and identically distributed (non-IID) data across clients, a core issue in FL [6]. Non-IID data arises in various forms, including attribute skew (differences in feature distributions), label skew (differences in label distributions), and temporal skew (time-based changes in data distributions), each complicating model training [7]. However, while much progress has been made in handling non-IID data, most studies focus on label heterogeneity rather than deeper task heterogeneity. Many approaches assume that all clients access the same set of class labels, even in non-IID settings. In reality, clients may encounter only subsets of labels relevant to their tasks [8], creating disparities in outputs and limiting collaborative learning. Addressing task heterogeneity, where clients solve fundamentally different problems, remains a significant challenge, especially in decentralized settings.

Aggregation algorithms, like FedAvg, struggle in multi-task learning scenarios due to their reliance on model homogeneity [9]. Task heterogeneity introduces complexities, as clients may require distinct model architectures for tasks (*e.g.,* classification versus point detection), leading to conflicting and dominant gradients during training [10], [11]. While recent methods such as FedBone [12] and FedHCA^2 [10] mitigate these challenges, they focus on centralized settings (CFMTL), leaving decentralized FMTL (DFMTL) largely unexplored. Decentralization amplifies these challenges due to the absence of a coordinating server, demanding robust methods for managing tasks and model diversity.

This work proposes *ColNet*, a decentralized framework for training multi-task models in heterogeneous environments. *ColNet* partitions models into a backbone and task-specific heads. *ColNet* uses adaptive clustering that leverages model sensitivity and data sensitivity to automatically partition nodes into task-coherent groups. Within each group, backbone layers are aggregated using FedAvg. Group leaders then exchange the group-averaged backbones across groups through a hyper-conflict-averse (HCA) aggregation scheme, which reduces gradient conflict and strengthens cross-task collaboration.

Experiments on the CIFAR-10 and CelebA datasets demonstrate the effectiveness of *ColNet*. Firstly, the framework's performance is analyzed with varying aggregation round frequencies, showing that more frequent aggregations improve model performance. Secondly, different levels of layer privatization are examined, revealing that sharing most layers benefits class label heterogeneity setups, while increasing privatization slightly enhances performance in task heterogeneity scenarios. Compared with other schemes, *ColNet* consistently outperforms baselines in both class label and task heterogeneity scenarios. Additional robustness evaluation indicates that *ColNet* is resilient to diverse poisoning attacks, including model poisoning and data poisoning, highlighting its capability to address diverse challenges in DFL.

TABLE I: Overview of Research in Federated Multi-Task Learning

Federation Schema	Techniques	Research	Approach used	Heterogeneity Addressed
Centralized	Convex Optimization	**MOCHA** [5]	Convex optimization, dynamic client inclusion	Label heterogeneity
		OFMTL [13]	Convex optimization, dynamic client inclusion	Label heterogeneity
	Regularization	**Ditto** [14]	Bi-level optimization, local regularization	Label heterogeneity
		PMTL [15]	Differential privacy, noise-injected global aggregation	Label heterogeneity
	Privatization	**FedPer** [16]	Shared base layers, private personalization layers	Label heterogeneity
		FedRep [17]	Shared base layers, head freezing	Label heterogeneity
	Grouping	**CFL** [18]	Client grouping, cosine similarity	Label heterogeneity
		HeurFedAMP [19]	Dynamic grouping, cosine similarity	Label heterogeneity
	Knowledge Distillation	**EFDLS** [20]	Knowledge distillation, local teacher-student models	Label heterogeneity
		FedICT [21]	Two-way knowledge distillation, reduced model updates	Label heterogeneity
	Hybrid	**FedBone** [12]	Split learning, gradient projection, task adaptation	Task heterogeneity
		FedHCA^2 [10]	Hyper conflict-averse aggregation, encoder-decoder architecture	Task heterogeneity
Decentralized	Regularization	**DCLPMN** [22]	Graph-based regularization, neighbor collaboration	Label heterogeneity
	Hybrid	**SpreadGNN** [23]	Decentralized aggregation, molecular data focus	Label heterogeneity
	Hybrid	***ColNet*** (This work)	Adaptive task clustering, shared backbone, hyper conflict-averse aggregation	Task heterogeneity

II. Related Work

This section provides an overview of FMTL research, covering both centralized and decentralized approaches and comparing them based on the techniques employed and types of heterogeneity addressed, as presented in TABLE I.

A. Centralized Approaches

FMTL was first introduced in 2017 with the MOCHA framework [5], and since then, a variety of approaches have emerged to tackle different aspects of multi-task learning in centralized federated settings. Early work primarily focused on solving convex optimization problems, such as MOCHA and OFMTL [13], which were robust but unsuitable for modern non-convex deep learning methods. Subsequent research addressed this limitation, extending CFMTL to handle non-convex problems and introducing techniques like regularization, privatization, grouping, and knowledge distillation.

Ditto [14] introduced bi-level optimization techniques that allow each client to maintain a personalized model while regularizing it toward a global model. This approach ensures uniform performance across devices and enhances resistance to data and model poisoning attacks. Similarly, PMTL [15] leverages relaxed differential privacy to learn personalized models while safeguarding client data. By adding noise during global aggregation, PMTL enables effective collaboration without compromising privacy. FedPer [16] tackled statistical heterogeneity by dividing models into shared base layers and private personalization layers. Extensions like FedRep [17] introduced mechanisms such as freezing the shared layers during early training phases to improve adaptability and achieve higher test accuracy.

Frameworks like CFL [18] introduced grouping strategies to cluster clients with similar data distributions, reducing the impact of conflicting gradients during aggregation. HeurFedAMP [19] further refined this concept by using cosine similarity to dynamically determine grouping, allowing more granular adaptation to client data heterogeneity. EFDLS [20] utilized knowledge distillation models to transfer knowledge efficiently within individual clients. This method enables collaborative learning by sharing only essential updates, reducing communication overhead. FedICT [21] extended this idea by employing a two-way distillation process that customizes local models for specific tasks while maintaining a flexible and adaptive global model, making it suitable for diverse client environments.

Most existing studies primarily focus on class label heterogeneity, restricting their adaptability in dynamic environments. Frameworks like FedBone [12] introduced split learning to separate general models on a central server from task-specific models on clients. Similarly, FedHCA^2 [10] proposed innovative aggregation techniques, such as HCA Aggregation, to address gradient conflicts, but it was designed for centralized environments. These advancements have predominantly relied on centralized coordination, simplifying aggregation but introducing challenges like single points of failure, limited scalability, and privacy risks.

B. Decentralized Approaches

DFMTL eliminates the need for a central server, allowing clients to communicate directly in a peer-to-peer manner. Despite the advantages, DFMTL introduces added challenges, such as managing gradient conflicts, asynchronous updates, and efficient client communication. Early research, such as DCLPMN [22], addressed these challenges by leveraging graph-based regularization to model task relationships, enabling collaboration between neighboring clients. SpreadGNN [23] applied decentralized aggregation techniques tailored for tasks like molecular data analysis to improve scalability and address domain-specific requirements.

As presented in TABLE I, while research on FMTL has made significant progress, most works address label heterogeneity and non-IID data rather than task heterogeneity, where clients solve different tasks. Additionally, decentralized settings remain underexplored compared to centralized frameworks, despite their potential for improved privacy and scalability. Existing frameworks rarely address task heterogeneity in decentralized environments, leaving a clear gap for Heterogeneous DFMTL. This work addresses this gap by proposing a novel framework, *ColNet*, designed to handle both label and task heterogeneity in a decentralized setting.

III. The *ColNet* Solution

This work introduces *ColNet*, a collaborative optimization approach designed for multi-task learning in DFL. This section defines the DFMTL problem and outlines its optimization objectives. It then presents the proposed *ColNet* framework, describing each step in turn.

A. Problem Statement

In a DFMTL system with N nodes, the learning objective is to collaboratively learning task-specific models while preserving data privacy and operating under a fully decentralized communication pattern.

Let $\mathcal{V} = \{1, 2, \ldots, N\}$ denote the set of nodes. Each node $i \in \mathcal{V}$ is associated with a task τ_i and holds a local dataset $\mathcal{D}_i$. The number of distinct tasks is $M \leq N$, accommodating the possibility that multiple nodes may share the same task or tasks may be heterogeneous across nodes.

The nodes form a connected, undirected graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{E}$ specifies the peer-to-peer (P2P) links. Communication is entirely decentralized, with no central server. Each node exchanges model updates or partial information solely with its neighbors $\mathcal{N}(i)$.

Each node i maintains a parameter vector $\mathbf{w}_i \in \mathbb{R}^d$ that addresses its local task τ_i. Heterogeneous local data render a single global model suboptimal, making personalized node-specific models more appropriate. Any common structure or correlation among tasks can be leveraged to improve overall performance.

Thus, the learning goal is framed as a joint optimization problem capturing both local performance and inter-task relationships:

$$\min_{\{\mathbf{w}_i\}_{i=1}^N} \sum_{i=1}^{N} f_i\big(\mathbf{w}_i; \mathcal{D}_i\big) \;+\; R\big(\{\mathbf{w}_i\}_{i=1}^N\big), \tag{1}$$

where $f_i(\mathbf{w}_i; \mathcal{D}_i)$ is the local loss function at node i, and $R(\{\mathbf{w}_i\})$ is a regularization or coupling term designed to exploit shared structure among tasks.

B. ColNet Architecture

To tackle the challenges of multi-task learning in DFL, *ColNet* uses collaborative optimization so that nodes benefit from both task-coherent peers and cross-task exchanges. Fig. 1 outlines a five-stage recurrent workflow.

1) Local Learning: Each node trains on its local dataset and updates both the shared *backbone* and its *task-specific layers*.
2) Model Exchanging: Nodes exchange their current *backbone* parameters with peers in the DFL overlay. Task-specific layers are not shared.
3) Adaptive Clustering and Intra-Group Aggregation: Using model-sensitivity and data-sensitivity indicators derived from Stage 2, nodes are automatically grouped into task-coherent clusters. Within each cluster, nodes perform *intra-group* aggregation on the backbone (FedAvg), while task-specific layers remain private to preserve task diversity. A cluster leader is designated to coordinate cross-group exchange.
4) Cross-Group Aggregation: Cluster leaders exchange their group-averaged backbone. Each leader applies a simplified HCA aggregation to reduce conflicting gradients across tasks, producing an updated backbone that integrates complementary information from other groups.
5) Model Redistribution: Leaders redistribute the updated backbone to all members of the corresponding clusters. Nodes synchronize their backbone parameters and resume Stage 1 with refreshed initialization.

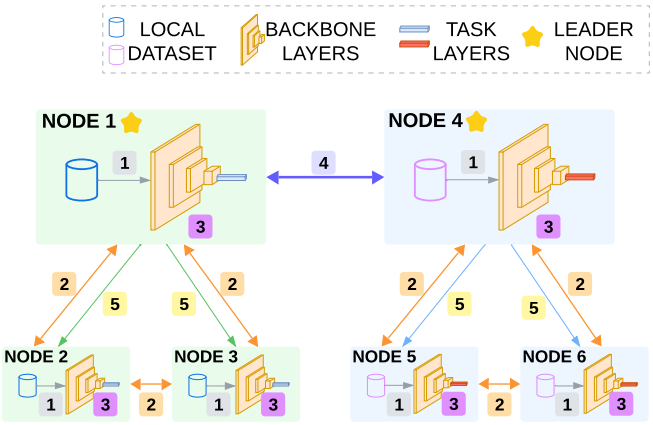

Fig. 1: Overview of the *ColNet* Learning Process

Algorithm 1 *ColNet* Five-Stage Client Workflow (Node i)

Require: Graph $G = (\mathcal{V}, \mathcal{E})$, neighbors $\mathcal{N}(i)$, data $\mathcal{D}_i$, loss ℓ, rounds R, local epochs E, clusters K, weights α, β, γ, temps $\tau_{\text{loss}}, \tau_J$, leader set $\mathcal{L}_{\text{lead}}$

Initialize $\mathbf{w}_i = (B_i, H_i)$; $B_i^{\text{prev}} \leftarrow B_i$
for $r = 1$ to R **do**
  Stage 1: Local Learning ▷ optimize f_i on $\mathcal{D}_i$
  Update (B_i, H_i) by E-epoch local training with loss ℓ
  Stage 2: Model Exchanging ▷ backbone only
  Exchange B_i with all $j \in \mathcal{N}(i)$ to obtain $\{B_j\}_{j \in \mathcal{N}(i)}$
  Stage 3: Adaptive Clustering & Intra-Group Aggregation
  $(g_i, \mathcal{K}(i)) \leftarrow \mathit{TaskAwareClustering}(B_i, H_i, \{B_j\}, \mathcal{D}_i)$
  $B_i \leftarrow \mathit{FedAvg}(\{B_k\}_{k \in \mathcal{K}(i)} \cup \{B_i\})$ ▷ heads H stay local
  Stage 4: Cross-Group Aggregation ▷ leaders only
  if $i \in \mathcal{L}_{\text{lead}}$ **then**
    $\Delta \bar{B}_i \leftarrow B_i - B_i^{\text{prev}}$
    $B_i \leftarrow \mathit{LeaderHCA}(B_i, \Delta \bar{B}_i, \mathcal{L}_{\text{lead}})$
  end if
  Stage 5: Redistribution
  if $i \in \mathcal{L}_{\text{lead}}$ **then**
    Broadcast updated B_i to all $k \in \mathcal{K}(i)$
  else
    Receive B_{leader} from group leader; $B_i \leftarrow B_{\text{leader}}$
  end if
  $B_i^{\text{prev}} \leftarrow B_i$
end for

C. Learning Process

Compared to standard DFL methods, the main novelty of *ColNet* during training lies in introducing *adaptive clustering*, *intra-task*, and *cross-task* aggregation. The learning process in each client is presented in Algorithm 1.

a) ***Local Learning****:* Client i solves the local objective on $\mathcal{D}_i$ for E epochs (cf. Eq. (1)), updating both B_i and H_i.

b) ***Model Exchanging****:* Clients exchange *backbone* parameters with neighbors in G, i.e., each node broadcasts B_i and receives $\{B_j\}_{j\in\mathcal{N}(i)}$. Task heads H are not shared.

c) ***Adaptive Clustering and Intra-Group Aggregation****:* Given received backbones $\{B_j\}_{j\in\mathcal{N}(i)}$, client i forms temporary models $M_0 = (B_i, H_i)$ and $M_j = (B_j, H_i)$ for all $j \in \mathcal{N}(i)$ so that all models share the *same* task head H_i. This aligns the evaluation to client i's task and avoids confounding from heterogeneous heads. The adaptive clustering is presented in Algorithm 2.

(1) Three signals on local data. On $\mathcal{D}_i$, client i computes three pairwise signals among the index set $\mathcal{I} = \{0\} \cup \mathcal{N}(i)$:

$$\text{Cosine similarity:} \quad C[p,q] \;=\; \frac{\langle \theta(M_p),\, \theta(M_q)\rangle}{\|\theta(M_p)\|\,\|\theta(M_q)\|}, \tag{2}$$

$$\text{Empirical loss:} \quad \Lambda[p,q] \;=\; \mathbb{E}_{(x,y)\sim\mathcal{D}_i}\big[\ell\big(f(x;M_q),y\big)\big], \tag{3}$$

$$\text{Avg. Jacobian norm:} \quad J[p,q] \;=\; \mathbb{E}_{x\sim\mathcal{D}_i}\left[\left\| \frac{\partial f(x;M_q)}{\partial z(x;M_q)} \right\|_F\right]. \tag{4}$$

Here $\theta(M)$ stacks the (flattened) backbone parameters of M (heads are identical and omitted), f is the task output (e.g., logits), and $z(\cdot; M_q)$ denotes the *backbone feature* right before the head. The Jacobian norm measures the model's local sensitivity to backbone features, serving as a proxy for model stability.

Together, these three signals capture both model similarity and data-dependent sensitivity on $\mathcal{D}_i$, providing a balanced view that is less biased than any single metric.

(2) Symmetrization and temperature scaling. Λ and J are asymmetric because they evaluate M_q on $\mathcal{D}_i$, thus, they are symmetrized via:

$$\begin{aligned} \Lambda &\leftarrow \tfrac{1}{2}(\Lambda + \Lambda^\top) \\ J &\leftarrow \tfrac{1}{2}(J + J^\top). \end{aligned} \tag{5}$$

Then, they are converted into similarities via temperatures $\tau_{\text{loss}}, \tau_J > 0$:

$$\begin{aligned} \Lambda_{\text{sim}}[p,q] &= \exp\big(-\Lambda[p,q]/\tau_{\text{loss}}\big) \\ J_{\text{sim}}[p,q] &= \exp\big(-J[p,q]/\tau_J\big). \end{aligned} \tag{6}$$

In practice, this work *standardizes* each matrix before exponentiation (z-score per-matrix) to balance scale across signals.

(3) Affinity fusion and self-confidence. This work fuses the three channels into an affinity matrix:

$$\begin{aligned} S \;&=\; \alpha\, C \;+\; \beta\, \Lambda_{\text{sim}} \;+\; \gamma\, J_{\text{sim}} \\ &\alpha,\beta,\gamma \ge 0 \;\&\; \alpha{+}\beta{+}\gamma = 1, \end{aligned} \tag{7}$$

Algorithm 2 TASKAWARECLUSTERING at Node i

Require: B_i, H_i, neighbor backbones $\{B_j\}_{j\in\mathcal{N}(i)}$, data $\mathcal{D}_i$, loss ℓ, α, β, γ, $\tau_{\text{loss}}, \tau_J$, clusters K
Ensure: Cluster label g_i, peer set $\mathcal{K}(i)$
Build models $M_0 \leftarrow (B_i, H_i)$ and $M_j \leftarrow (B_j, H_i)$ for $j \in \mathcal{N}(i)$
Let index set $\mathcal{I} \leftarrow \{0\} \cup \mathcal{N}(i)$; init $C, \Lambda, J \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}$
for all $p, q \in \mathcal{I}$ **do**
  $C[p,q] \leftarrow \mathit{CosSim}(M_p, M_q)$
  $\Lambda[p,q] \leftarrow \mathit{AvgLoss}(M_q; \mathcal{D}_i, \ell)$
  $J[p,q] \leftarrow \mathit{AvgJacobianNorm}(M_q; \mathcal{D}_i)$
end for
$\Lambda \leftarrow (\Lambda + \Lambda^\top)/2$, $J \leftarrow (J + J^\top)/2$
$\Lambda_{\text{sim}}[p,q] \leftarrow \exp(-\Lambda[p,q]/\tau_{\text{loss}})$
$J_{\text{sim}}[p,q] \leftarrow \exp(-J[p,q]/\tau_J)$
$S \leftarrow \alpha C + \beta\Lambda_{\text{sim}} + \gamma J_{\text{sim}}$; set $S[p,p] \leftarrow 1$
Apply spectral clustering on S into K clusters; obtain g_i
$\mathcal{K}(i) \leftarrow \{k \in \mathcal{N}(i) \cup \{i\} \mid g_k = g_i\}$
return $(g_i, \mathcal{K}(i))$

(4) Spectral clustering. Apply normalized spectral clustering on the affinity matrix S to obtain K clusters. Let g_i be the label assigned to index 0 (client i), and define its in-group peers as $\mathcal{K}(i) = \{k \in \mathcal{N}(i) \cup \{i\} \mid g_k = g_i\}$.

(5) Intra-group aggregation (backbone only). Within $\mathcal{K}(i)$, clients average backbones while keeping heads local:

$$\bar{\mathbf{w}}^B \;=\; \frac{1}{|\mathcal{K}(i)|}\sum_{k\in\mathcal{K}(i)} \mathbf{w}_k^B, \qquad B_i \leftarrow \bar{\mathbf{w}}^B. \tag{8}$$

This produces a group-coherent backbone B_i tailored to i's task via H_i.

(6) Leader Selection. Once task groups are established, each group appoints a leader on a rotating basis to balance coordination overhead. In round r, one member serves as leader and coordinates inter-group aggregation; after this step, leadership rotates (round-robin) to the next member in the group. The outgoing leader announces the handover to its group and to peer leaders so that subsequent cross-group exchanges use the updated routing.

d) ***Cross-Group Aggregation****:* Cross-group aggregation is challenging due to potential conflicts between updates produced by heterogeneous tasks. Let $\mathcal{L}_{\text{lead}}$ denote the set of current group leaders and $G_\ell = |\mathcal{L}_{\text{lead}}|$ the number of groups. Each leader $i \in \mathcal{L}_{\text{lead}}$ computes a backbone delta to its previous round,

$$\Delta\mathbf{w}_i^B \;=\; \bar{\mathbf{w}}_i^B - \mathbf{w}_{i,\text{prev}}^B, \tag{9}$$

exchanges $\Delta\mathbf{w}_i^B$ with other leaders, and forms the mean update $\Delta\bar{\mathbf{w}}^B = \frac{1}{G_\ell}\sum_{\ell\in\mathcal{L}_{\text{lead}}} \Delta\mathbf{w}_\ell^B$.

Inspired by FedHCA^2 [10], *ColNet* adapts a gradient-alignment objective that seeks an aggregated update $\tilde{U}$ close to $\Delta\bar{\mathbf{w}}^B$ while improving its alignment with the worst-case task direction:

$$\begin{aligned} &\max_{\tilde{U}} \min_i \langle \Delta\mathbf{w}_i^B, \tilde{U}\rangle \\ &\text{s.t.} \quad \|\tilde{U} - \Delta\bar{\mathbf{w}}^B\| \;\le\; c\,\|\Delta\bar{\mathbf{w}}^B\| \end{aligned} \tag{10}$$

Algorithm 3 LEADERHCA (Cross-Group Agg. at Leader i)

Require: Current backbone B_i, local delta $\Delta\bar{B}_i$, leader set $\mathcal{L}_{\text{lead}}$
Ensure: Updated backbone B_i
1: **for all** $\ell \in \mathcal{L}_{\text{lead}} \setminus \{i\}$ **do**
2: Send $\Delta\bar{B}_i$ to ℓ; Receive $\Delta\bar{B}_\ell$ from ℓ
3: **end for**
4: $\tilde{U}_i \leftarrow HCA(\Delta\bar{B}_i, \{\Delta\bar{B}_\ell\}_{\ell\in\mathcal{L}_{\text{lead}}\setminus\{i\}})$ ▷ see Eq. (11)
5: $B_i \leftarrow B_i + \tilde{U}_i$
6: **return** B_i

where $c \in [0, 1)$ bounds the deviation from the mean update. A Lagrangian treatment yields the closed-form

$$\begin{aligned} \tilde{U} &= \Delta\bar{\mathbf{w}}^B + \frac{\sqrt{\phi}}{\|U_w\|} U_w \\ U_w &= \frac{1}{G_\ell} \sum_{\ell\in\mathcal{L}_{\text{lead}}} w_\ell \, \Delta\mathbf{w}_\ell^B \\ \phi &= c^2 \|\Delta\bar{\mathbf{w}}^B\|^2 \end{aligned} \tag{11}$$

with nonnegative weights w_ℓ determined by the chosen conflict metric. This update balances proximity to the mean and reduction of cross-task conflicts, as shown in Algorithm 3. Each leader applies $\tilde{U}$ to its backbone and proceeds to redistribution.

*e) **Model Redistribution**:* Group leaders broadcast the updated backbone to their members, who synchronize B and continue with the next round, while task heads remain private. The rotating-leader policy is then applied to select successors for the following cross-task aggregation.

Stages (2)–(3) adapt collaboration to both model- and data-driven signals, which improves stability under label and task heterogeneity. Stage (4) integrates complementary cross-task information while mitigating gradient conflicts. The five-stage loop repeats until a stopping criterion is met.

IV. DEPLOYMENT AND EVALUATION

This section details the deployment and experimental evaluation of *ColNet*, covering the targeted multi-task learning scenarios, the chosen evaluation metrics, and the resulting performance outcomes [1].

A. Scenarios, Datasets, and Models

Two heterogeneity scenarios are considered for evaluation: *label heterogeneity* and *task heterogeneity*.

Label heterogeneity arises when clients train on different subsets of class labels, which requires splitting the model into a shared backbone and task-specific layers to protect privacy and maintain performance. Clients with the same labels form a single task group, sharing only the backbone parameters. The **CIFAR-10** dataset [24] was selected to evaluate label heterogeneity, where each image belongs to one of ten classes. To simulate label heterogeneity, the dataset is partitioned into two super-classes: **animals** (bird, cat, deer, dog, frog, horse) and **objects** (airplane, automobile, ship, truck). Within each super-class, samples are split evenly and assigned to three clients in an IID manner.

[1]Code available at: https://github.com/Cyber-Tracer/asfdfmtl

The dataset is split into 50,000 images for training and 10,000 for testing, further subdivided into animal or object sets. Each client's training subset is then partitioned (80% train and 20% validation), and all clients use the same test subset in a task group. Data augmentation techniques, such as random cropping, flipping, and color jittering are applied to mitigate overfitting.

All clients train a ResNet-18 [25] backbone and separate task layers to accommodate different output dimensions (four classes for objects, six for animals). A Cross-Entropy loss and SGD optimizer (learning rate 0.01, momentum 0.9, weight decay 1e-3) are employed.

Task heterogeneity is more complex, as clients may work on entirely different tasks (e.g., image classification and object detection), each requiring distinct final layers, loss functions, and optimizers. Dividing the model into a shared backbone and task-specific layer can facilitate aggregation in this scenario. The **CelebA** dataset [26] is used to evaluate task heterogeneity. It contains over 200,000 face images with up to 40 binary attributes and five landmark annotations, making it suitable for multiple tasks, including face attribute recognition and landmark detection [27], [28], [29].

To simulate task heterogeneity, two distinct task groups are defined: **multi-label face attribute classification** and **face landmark detection**. A subset comprising 80% of the dataset is used for training/validation, created by first selecting 40% for each task group and assigning it evenly across that group's clients; the remaining 20% serves as a common test set. Each client then partitions its local portion into 80% training and 20% validation. Standard augmentations (random rotation and horizontal flips) are applied to mitigate overfitting.

Both tasks employ ResNet-18 backbone and distinct task layers: (a) **Face attribute classification** uses Binary Cross Entropy loss and an AdamW optimizer 5e-5 learning rate, 5e-3 weight decay; (b) **Face landmark detection** uses Mean Squared Error loss and an Adam optimizer 5e-4 learning rate.

In the experiments, the overlay network uses a fully connected topology, and nodes have no knowledge of other nodes' task types. After each local training round, each node disseminates only its backbone parameters to all other nodes in the network. Adaptive task clustering is then performed based on the received backbones and local data. Normalized spectral clustering is used with the number of clusters set to k=2, after which intra-group aggregation and subsequent steps proceed.

The experiments report four evaluation metrics: loss, precision, recall, and F1-Score. Since face landmark detection is a regression task, only the loss metric is used to evaluate this subtask. To handle class imbalance and multi-class scenarios, *micro* averaging strategy is used, aggregating metrics across all classes within each task group. Metrics are kept separate for different groups because their labels (in the label heterogeneity scenario) or tasks (in the task heterogeneity scenario) are distinct.

B. Federation Setup

For the federated setup, six nodes are split into two task groups, each fully connected internally, with inter-group communication handled by leader nodes. Fifteen training rounds are performed, each comprising two local epochs followed by intra-group and cross-group model aggregation. To investigate aggregation frequency, the experiment also compares 2, 3, and 5 local epochs per round.

Three baselines are implemented to assess the proposed *ColNet* framework:

1) No aggregation: Nodes train only on local data.
2) Intra-aggregation only: Nodes aggregate their backbone parameters within each group, with no cross-group exchange.
3) FedPre (intra- and cross-aggregation): Nodes use FedPre [16] for both intra-group and cross-group backbone aggregation.

By contrast, *ColNet* combines FedPre for intra-group aggregation with HCA-based cross-group aggregation, enabling more effective handling of diverse tasks.

C. Impact of Aggregation Frequency

To maximize multi-task learning performance, it is important to determine the optimal aggregation frequency, *i.e.,* how many local epochs to run per training round. Hence, the initial experiment compares *ColNet*'s performance with 2, 3, and 5 local epochs under two evaluation scenarios, thereby identifying the most suitable hyperparameter.

TABLE II: Average Client Test Metrics of *ColNet* for Two Evaluation Scenarios with Different Local Epoch Setups

Scenario	Subtask	Epochs	Loss	Precision	Recall	F1-Score
Label Heterogeneity (CIFAR-10)	Animal Classification	2	**0.673**	**0.779**	**0.768**	**0.769**
		3	0.786	0.764	0.732	0.729
		5	0.801	0.743	0.727	0.717
	Object Classification	2	0.370	0.888	0.878	0.877
		3	**0.333**	**0.895**	**0.891**	**0.891**
		5	0.383	0.884	0.874	0.874
Task Heterogeneity (CelebA)	Attribute Classification	2	**0.231**	**0.741**	**0.545**	**0.605**
		3	0.233	0.735	0.542	0.598
		5	0.237	0.733	0.522	0.579
	Landmark Detection	2	5.299	-	-	-
		3	**5.224**	-	-	-
		5	5.762	-	-	-

TABLE II presents the results for four subtasks: animal classification, object classification, face attribute classification, and face landmark detection. Since face landmark detection is a regression task, only the loss metric is reported for that subtask. Besides, Fig. 2 presents the epoch-wise validation loss for each subtask with different local training epoch settings.

In the label heterogeneity scenario (*i.e.,* CIFAR-10), using two epochs per aggregation achieves the best performance for the animal group, while the object group slightly trails the three-epoch scheme. However, the difference is negligible, and both precision and recall remain closely aligned. From a training perspective (see Figure 2), the two-epoch setup converges more smoothly and exhibits lower loss. Despite the minor drop for the object group, the overall benefits of two-epoch intervals prevail.

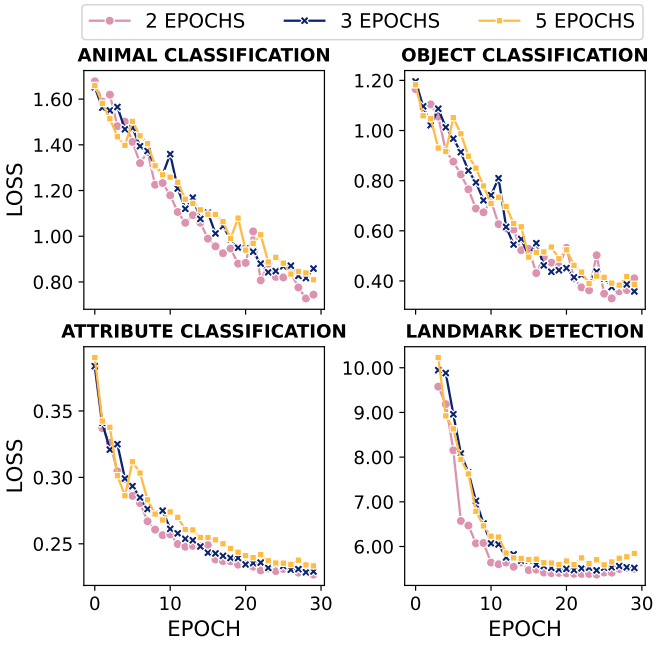

Fig. 2: Average Client Validation Loss in Each Epoch with Different Local Epoch Setups

Frequent aggregations benefit face attribute classification in the task heterogeneity scenario (*i.e.,* CelebA). While the landmark detection task achieves its best results with three epochs, the two-epoch setting converges more quickly and smoothly. Consequently, two local training epochs per round are adopted in all remaining experiments.

D. Impact of the Privatization Degree

When deploying the backbone and task layers, a crucial hyperparameter concerns how to allocate these two parts of the network architecture. Although both scenarios in this work rely on ResNet-18 as the base model, determining which layers form the backbone and which form the task layers still requires careful consideration. Retaining more of the original, lower-level ResNet-18 layers in the backbone increases the amount of shared information among nodes, which may be detrimental for highly heterogeneous tasks. Conversely, expanding the upper residual layers into the task layer offers a higher degree of privatization but may weaken the effectiveness of aggregations.

To evaluate how different levels of privatization impact multi-task learning, three configurations are compared: (i) no residual layers are privatized (BL-0), (ii) the last residual layer is part of the task layer (BL-1), and (iii) the last two residual layers are part of the task layer (BL-2). BL-0 represents the lowest degree of privatization, whereas BL-2 represents a high degree.

TABLE III compares the performance of *ColNet* client models under these three settings in both scenarios. Besides,

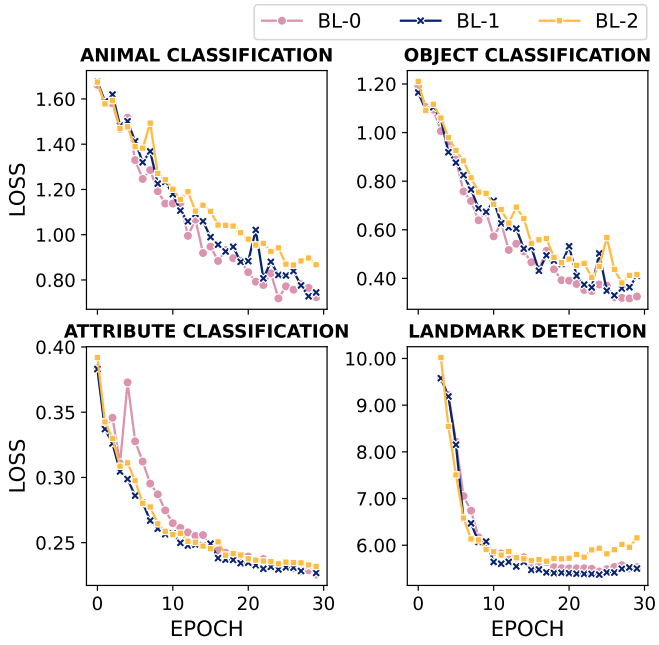

Fig. 3: Average Client Validation Loss in Each Epoch with Different Privatization Setups

Fig. 3 presents the validation loss in each epoch during the training process. Overall, BL-2 shows the weakest performance, implying that retaining more residual layers in the backbone benefits DFMTL by providing sufficient neurons to exchange knowledge across different tasks. At the same time, a suitable number of task layers enables proper personalization under varied conditions. Drawing on the results from both scenarios, a moderate privatization strategy—treating the last residual block and the output layer of ResNet-18 as task layers (*i.e.,* BL-1) proves optimal and is therefore adopted for subsequent experiments.

E. Comparison Among Aggregation Schemes

Having established an appropriate aggregation frequency and layer privatization strategy, the *ColNet* framework is now compared with other aggregation schemes. The main objective is to determine whether clients that leverage cross-task knowledge achieve better performance on their own tasks.

TABLE III: Average Client Test Metrics of *ColNet* for Two Evaluation Scenarios with Different Privatization Setups

Scenario	Subtask	Privatization	Loss	Precision	Recall	F1-Score
Label Heterogeneity (CIFAR-10)	Animal Classification	BL-0	0.706	**0.786**	0.759	0.758
		BL-1	**0.673**	0.779	**0.768**	**0.769**
		BL-2	0.810	0.739	0.720	0.716
	Object Classification	BL-0	**0.313**	**0.900**	**0.895**	**0.895**
		BL-1	0.370	0.888	0.878	0.877
		BL-2	0.371	0.881	0.876	0.875
Task Heterogeneity (CelebA)	Attribute Classification	BL-0	**0.230**	0.738	0.541	0.599
		BL-1	0.231	**0.741**	**0.545**	**0.605**
		BL-2	0.236	0.734	0.533	0.592
	Landmark Detection	BL-0	**5.285**	-	-	-
		BL-1	5.299	-	-	-
		BL-2	5.829	-	-	-

This experiment compares the performance of *ColNet* with (i) No aggregation (NO AGG.), (ii) Intra-aggregation (INTRA-AGG.), and (iii) FedPre with intra- and cross-aggregation (FEDPRE I&C), in label heterogeneity and task heterogeneity scenarios. TABLE IV presents the test metrics of different aggregation schemes in these two scenarios after 15 federated rounds with a total of 30 epochs.

TABLE IV: Average Client Test Metrics for Different Aggregation Schemes

Scenario	Subtask	Aggregation	Loss	Precision	Recall	F1-Score
Label Heterogeneity (CIFAR-10)	Animal Classification	NO AGG.	0.938	0.699	0.675	0.669
		INTRA-AGG.	0.953	0.723	0.680	0.679
		FEDPRE(I&C)	0.853	0.730	0.700	0.689
		ColNet	**0.673**	**0.779**	**0.768**	**0.769**
	Object Classification	NO AGG.	0.464	0.846	0.840	0.840
		INTRA-AGG.	0.409	0.867	0.855	0.856
		FEDPRE(I&C)	**0.346**	0.879	0.875	0.875
		ColNet	0.370	**0.888**	**0.878**	**0.877**
Task Heterogeneity (CelebA)	Attribute Classification	NO AGG.	0.244	0.722	0.511	0.570
		INTRA-AGG.	0.236	0.733	0.526	0.585
		FEDPRE(I&C)	0.239	0.726	0.513	0.571
		ColNet	**0.231**	**0.741**	**0.545**	**0.605**
	Landmark Detection	NO AGG.	6.054	-	-	-
		INTRA-AGG.	5.392	-	-	-
		FEDPRE(I&C)	5.325	-	-	-
		ColNet	**5.299**	-	-	-

[NO AGG.: No aggregation; INTRA-AGG.: Intra-aggregation; FEDPRE(I&C): FedPre with intra- and cross-aggregation]

TABLE IV compares *ColNet* with three reference aggregation schemes under both the label heterogeneity and task heterogeneity scenarios. The results demonstrate that *ColNet* consistently outperforms the other methods across multiple datasets in multi-task settings.

In the label heterogeneity scenario, *ColNet* achieves the highest performance in both the animal and object classification subtasks. Notably, the F1-Score for animal classification exceeds the reference aggregation schemes by more than 0.08, signifying a substantial improvement. The findings further highlight that collaborative learning in a federated environment produces clearly superior results compared to no aggregation at all. Aggregating only within task groups increases precision by about 0.02, while employing an intra- and inter-group FedPer approach improves performance even more. Besides, *ColNet*'s cross-group HCA method effectively resolves conflicting gradients among heterogeneous tasks, allowing nodes to absorb knowledge from tasks beyond their own and enhancing local task performance.

A similar pattern is presented in the task heterogeneity scenario. Models without aggregation perform worst, whereas introducing inter-group aggregation yields a 0.01 gain in precision and a notable reduction in loss for the landmark detection subtask. With *ColNet*, both subtasks attain the best results: the F1-Score rises by 0.03, precision increases by 0.02, and the landmark detection loss drops by 0.8. These outcomes underscore *ColNet*'s clear advantages over other aggregation strategies in DFMTL.

Fig. 4 presents the average client validation loss across epochs with using different aggregation schemes. Across all tasks, *ColNet* consistently achieves the lowest validation loss, highlighting its effectiveness in reducing errors during training.

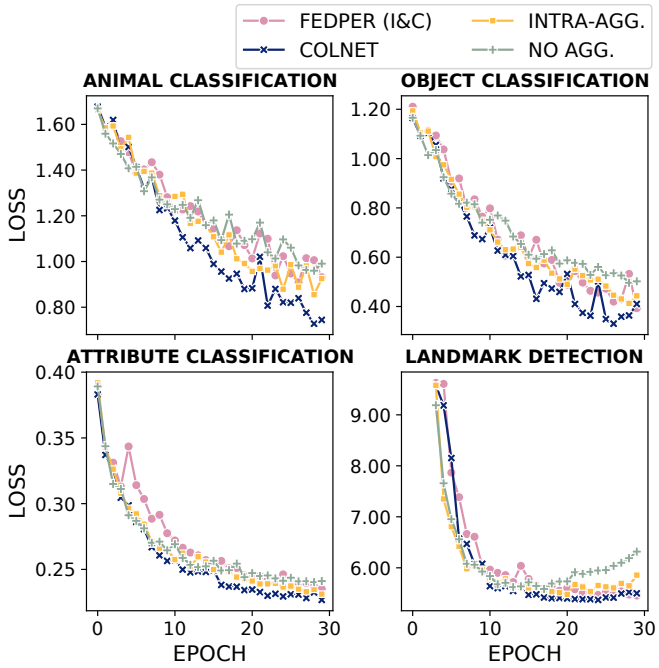

Fig. 4: Average Client Validation Loss in Each Epoch for Different Aggregation Schemes

This advantage is particularly pronounced in animal classification and landmark detection, where *ColNet*'s smoother and more stable convergence clearly outperforms the other methods. The results underscore *ColNet*'s ability to manage both inter-task and intra-task heterogeneity more effectively than simpler aggregation schemes.

In contrast, FedPre and intra-aggregation show higher loss values and less stable convergence trends, while no aggregation performs the worst in all scenarios, reinforcing the critical importance of aggregation in federated multi-task learning. Notably, *ColNet* demonstrates a significant advantage by mitigating gradient conflicts and enabling knowledge sharing. These findings affirm *ColNet*'s superiority in DFMTL environments.

F. Robustness Analysis of ColNet

In the preceding experiments, all participating clients were assumed benign (honest and non-adversarial). In decentralized learning, this assumption often breaks down: some clients may behave maliciously, most notably via poisoning attacks that corrupt model updates or local data, degrade performance, and undermine model usability [30]. Accordingly, this study evaluates *ColNet*'s resilience to multiple poisoning settings, covering both model-poisoning and data-poisoning variants, and analyzes its model robustness.

1) Poisoning Attack Strategies: This implements four poisoning attack strategies to evaluate the model robustness of the proposed *ColNet*, covering data poisoning attacks (untargeted label flipping), model poisoning attacks (scaled boost and AT2FL [31]), and aggregation manipulation.

Untargeted Label Flipping. Before local training, labels are corrupted to inject label noise: for multi-label tasks, each label is flipped with probability p (Bernoulli mask), and for single-label tasks, the true class is replaced with a uniformly sampled different class with probability p. The operation degrades overall generalization without steering predictions toward a specific alternative.

Scaled Boost. After local training but before upload, the client scales its update by a factor $s > 1$ and uploads $s \cdot \Delta_i$, optionally preserving non-trainable statistics and applying clipping or BN re-estimation to reduce detectability, thereby amplifying the client's influence on global updates and inducing overshoot or instability.

AT2FL (Inner-Loop Adversarial Data Poisoning). During local training, for a small number of mini-batches the attacker computes input gradients and crafts adversarial inputs $\mathbf{x}_{\text{adv}} = \text{Proj}_{\mathcal{X}}(\mathbf{x} + \epsilon\,\text{sign}(\nabla_{\mathbf{x}} L))$, performs additional optimization steps on these perturbed samples to steer local parameters toward directions that increase future loss on target nodes, then uploads the manipulated backbone.

Aggregation Manipulation (Malicious Aggregation Filter). At aggregation, the aggregator replaces or perturbs the conflict-averse aggregate $\Delta_{\text{HCA}} = \mathcal{A}_{\text{HCA}}(\{\Delta_i\})$ with $\tilde{\Delta}_{\text{agg}} = \mathcal{F}_{\text{mal}}(\Delta_{\text{HCA}}, \{\Delta_i\})$, thereby compromising cross-task reconciliation or implanting group-level backdoors.

2) Attack evaluation setup: The robustness analysis experiments follow the same base training and evaluation protocol as the model-performance experiments. The only change is that, in each attack trial, a single client is designated as the *attacker* while all other clients remain benign. Comparative runs are executed in two modes: (a) *clean baseline* (no attacker present) and (b) *attacked* (one attacker active). Robustness is measured by the impact of the attack on the performance of benign clients; that is, if the loss and F1-Score of benign clients degrade only mildly under attack relative to the clean baseline, the system is considered more robust.

The dataset-specific attacker placement used in the experiments is as follows: (i) **CIFAR-10 (label-heterogeneous setting).** The first node in the Animal Classification (AN_N0) task is set as the attacker; the remaining nodes are benign. (ii) **CelebA (task-heterogeneous setting).** The first node in the Attribute Classification (AC_N0) task is set as the attacker; the remaining nodes are benign.

3) Robustness Analysis Results:

a) Untargeted label flipping.: This attack flips labels in the attacker's local dataset before training. As shown in Fig. 5, its effect is highly concentrated on the poisoned node: e.g., on CIFAR-10 the poisoned AN_N0's loss increased from ≈ 0.65 to ≈ 4.41 and its F1 dropped from 0.77 to 0.052 (a ~ 0.72 absolute F1 loss). By contrast, same-group mates AN_N1/N2) experienced negligible aggregate harm (mean F1 stayed roughly the same: $\sim 0.734 \rightarrow \sim 0.736$), and cross-group object nodes (OB_N) were essentially unaffected. A similar pattern holds on CelebA dataset: the poisoned attribute node (AC_N0) suffered a large F1 drop ($0.591 \rightarrow 0.139$), while the other attribute nodes showed small or no degradation.

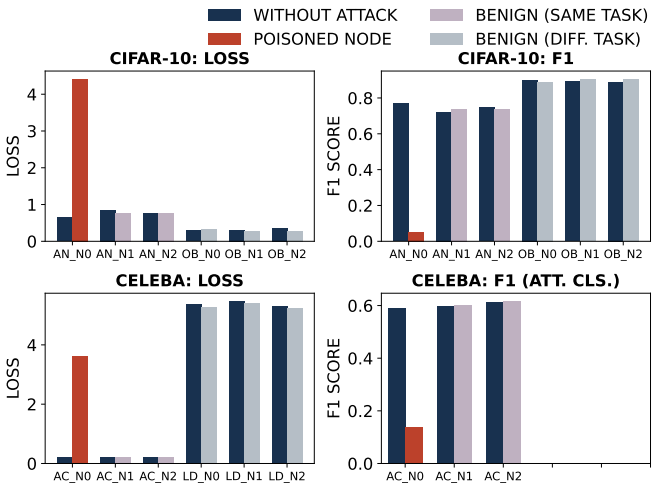

Fig. 5: Impact of Untargeted Label Flipping Attack on CIFAR-10 and CelebA Datasets

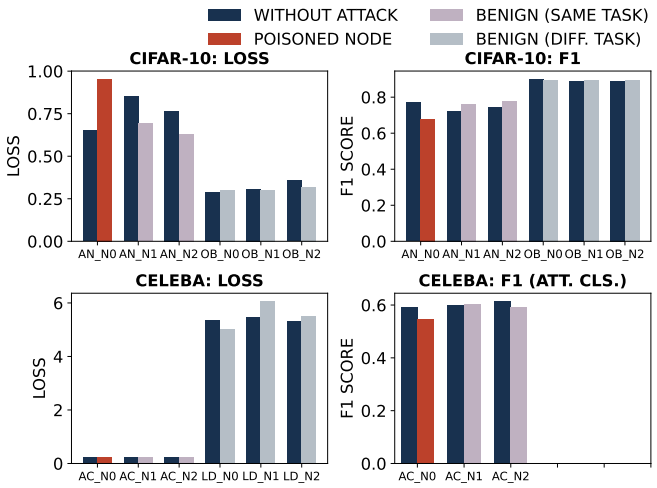

Fig. 7: Impact of AT2FL Attack on CIFAR-10 and CelebA Datasets

b) Scaled Boost (model poisoning).: Scaled Boost amplifies a client's update before upload (here $\times 5$ amplified). As shown in Fig. 6, the poisoned client incurs the largest absolute degradation, for CIFAR-10 AN_N0 F1 fell from 0.77 to ≈ 0.18, but unlike simple label flips there is measurable collateral damage to same-group benign clients: AN_N1/N2 saw modest F1 drops (each $\approx$ 0.04–0.05 absolute), while cross-group object nodes remained effectively stable (changes ≈ 0 or within noise). On CelebA, the attacker's attribute node (AC_N0) lost ~ 0.27 F1, and the other attribute nodes suffered smaller but non-negligible drops.

c) AT2FL (inner-loop adversarial poisoning).: AT2FL crafts adversarial inputs during a few local mini-batches to bias the local update towards directions that maximize future loss on target nodes. As shown in Fig. 7, for CIFAR-10, AN_N0's F1 decreased from 0.77 to ≈ 0.68, while same-group mates (AN_N1/N2) in some runs even improved slightly, indicating that adversarially nudging shared representations can have non-uniform, task-dependent side effects. Cross-group object nodes remained essentially stable (very small changes). On CelebA the AT2FL perturbations caused small-to-moderate F1 decreases on the attribute nodes (e.g., AC_N0: $0.591 \rightarrow 0.547$).

d) Aggregation manipulation.: Tampering with the aggregation routine (replacing or perturbing the conflict-averse solution) is the most system-wide attack. As shown in Fig.8, in CIFAR-10 the malicious aggregation caused sizeable F1 drops across both same-group and cross-group nodes: AN nodes' F1 decreased from ~ 0.77 to ~ 0.55 (same-group mean drop ≈ 0.18), and object nodes' F1 also fell substantially

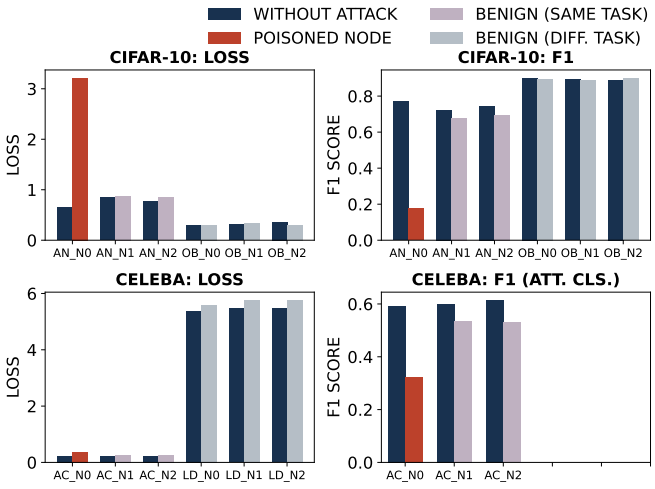

Fig. 6: Impact of Scaled Boost Attack on CIFAR-10 and CelebA Datasets

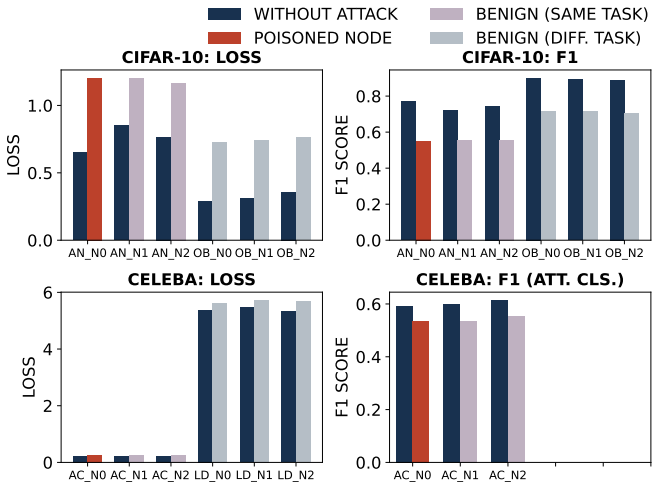

Fig. 8: Impact of Aggregation manipulation Attack on CIFAR-10 and CelebA Datasets

(cross-group mean drop ≈ 0.18). On CelebA the aggregation attack produced roughly uniform declines across attribute nodes (each AC_N lost ~ 0.06 F1) and raised landmark losses.

Untargeted label flipping primarily collapses the attacker's own model with minimal spillover to other clients; model-level manipulations (e.g., Scaled Boost) inflict stronger local damage and measurable collateral degradation within the attacker's task group; and attacks that subvert the aggregation stage break task isolation and cause the most severe, system-wide declines, affecting both same-task and cross-task nodes.

Overall, *ColNet* demonstrates model robustness: it strongly resists isolated label noise and modest manipulations, but remains vulnerable when an attacker either (i) amplifies their aggregate influence or (ii) directly compromises the aggregation routine, suggesting defense priorities of anomaly detection for client updates and hardening aggregation integrity.

V. Summary and Future Work

ColNet is a decentralized federated multi-task learning framework that groups clients by task, averages backbones within each group, and reconciles group updates via an HCA-style conflict-averse aggregator. An adaptive clustering module discovers and refines group membership over time. Empirical results show consistent gains over alternative aggregators while demonstrating improved robustness. The combination of a shared backbone, task-specific heads, HCA aggregation, and adaptive clustering substantially limits cross-task propagation of client-side attacks.

Future work focuses on fault-tolerant deployment, including leader election and rotation, straggler mitigation, and support for asynchronous rounds, together with stronger aggregation hardening through Byzantine-robust and secure aggregation to balance accuracy, privacy, and robustness. Large-scale distributed evaluations against stronger adaptive adversaries will quantify trade-offs and guide improvements toward comprehensive robustness guarantees.

References

[1] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[2] E. T. M. Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Communications Surveys & Tutorials*, 2023.

[3] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, pp. 41–75, 1997.

[4] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2021.

[5] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," *Advances in neural information processing systems*, vol. 30, 2017.

[6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv:1806.00582*, 2018.

[7] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," 2021. [Online]. Available: https://arxiv.org/abs/2106.06843

[8] R. Aoki, F. Tung, and G. L. Oliveira, "Heterogeneous multi-task learning with expert diversity," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 19, no. 6, pp. 3093–3102, 2022.

[9] W. Huang, M. Ye, and B. Du, "Learn from others and be yourself in heterogeneous federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 143–10 153.

[10] Y. Lu, S. Huang, Y. Yang, S. Sirejiding, Y. Ding, and H. Lu, "Fedhca2: Towards hetero-client federated multi-task learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5599–5609.

[11] J. Yu, Y. Dai, X. Liu, J. Huang, Y. Shen, K. Zhang, R. Zhou, E. Adhikarla, W. Ye, Y. Liu *et al.*, "Unleashing the power of multi-task learning: A comprehensive survey spanning traditional, deep, and pretrained foundation model eras," *arXiv:2404.18961*, 2024.

[12] Y. Chen, T. Zhang, X. Jiang, Q. Chen, C. Gao, and W. Huang, "Fedbone: Towards large-scale federated multi-task learning," *arXiv:2306.17465*, 2023.

[13] R. Li, F. Ma, W. Jiang, and J. Gao, "Online federated multitask learning," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 215–220.

[14] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International conference on machine learning*. PMLR, 2021, pp. 6357–6368.

[15] S. Hu, Z. S. Wu, and V. Smith, "Private multi-task learning: Formulation and applications to federated learning," *arXiv:2108.12978*, 2021.

[16] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv:1912.00818*, 2019.

[17] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *International conference on machine learning*. PMLR, 2021, pp. 2089–2099.

[18] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 8, pp. 3710–3722, 2020.

[19] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-iid data," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 9, 2021, pp. 7865–7873.

[20] H. Xing, Z. Xiao, R. Qu, Z. Zhu, and B. Zhao, "An efficient federated distillation learning system for multitask time series classification," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[21] Z. Wu, S. Sun, Y. Wang, M. Liu, Q. Pan, X. Jiang, and B. Gao, "Fedict: Federated multi-task distillation for multi-access edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 35, no. 6, pp. 1107–1121, 2023.

[22] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," *arXiv:1610.05202*, 2016.

[23] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr, "Spreadgnn: Decentralized multi-task federated learning for graph neural networks on molecular data," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 6, 2022, pp. 6865–6873.

[24] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[26] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[27] A. Woubie, E. Solomon, and J. Attieh, "Maintaining privacy in face recognition using federated learning method," *IEEE Access*, 2024.

[28] M. Rasouli, T. Sun, and R. Rajagopal, "Fedgan: Federated generative adversarial networks for distributed data," *arXiv:2006.07228*, 2020.

[29] M. Mortaheb, C. Vahapoglu, and S. Ulukus, "Fedgradnorm: Personalized federated gradient-normalized multi-task learning," in *2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*. IEEE, 2022, pp. 1–5.

[30] C. Feng, A. H. Celdrán, J. Von der Assen, E. T. M. Beltrán, G. Bovet, and B. Stiller, "Dart: A solution for decentralized federated learning model robustness analysis," *Array*, vol. 23, p. 100360, 2024.

[31] G. Sun, Y. Cong, J. Dong, Q. Wang, L. Lyu, and J. Liu, "Data poisoning attacks on federated machine learning," *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 11 365–11 375, 2022.