Evidential Trust-Aware Model Personalization in Decentralized Federated Learning for Wearable IoT

Murtaza Rangwala, Richard O. Sinnott and Rajkumar Buyya
Quantum Cloud Computing and Distributed Systems (qCLOUDS) Lab
School of Computing and Information Systems
The University of Melbourne, Australia
Email: {rangwalam, rsinnott, rbuyya}@unimelb.edu.au

***Abstract*—Decentralized federated learning (DFL) enables collaborative model training across edge devices without centralized coordination, offering resilience against single points of failure. However, statistical heterogeneity arising from non-identically distributed local data creates a fundamental challenge: nodes must learn personalized models adapted to their local distributions while selectively collaborating with compatible peers. Existing approaches either enforce a single global model that fits no one well, or rely on heuristic peer selection mechanisms that cannot distinguish between peers with genuinely incompatible data distributions and those with valuable complementary knowledge. We present MURMURA, a framework that leverages evidential deep learning to enable trust-aware model personalization in DFL. Our key insight is that epistemic uncertainty from Dirichlet-based evidential models directly indicates peer compatibility: high epistemic uncertainty when a peer's model evaluates local data reveals distributional mismatch, enabling nodes to exclude incompatible influence while maintaining personalized models through selective collaboration. MURMURA introduces a trust-aware aggregation mechanism that computes peer compatibility scores through cross-evaluation on local validation samples and personalizes model aggregation based on evidential trust with adaptive thresholds. Evaluation on three wearable IoT datasets (UCI HAR, PAMAP2, PPG-DaLiA) demonstrates that MURMURA reduces performance degradation from IID to non-IID conditions compared to baseline (0.9% vs. 19.3%), achieves 7.4× faster convergence, and maintains stable accuracy across hyperparameter choices. These results establish evidential uncertainty as a principled foundation for compatibility-aware personalization in decentralized heterogeneous environments.**

***Index Terms*—Decentralized federated learning, evidential deep learning, trust-aware personalization, model personalization**

I. INTRODUCTION

Wearable Internet of Things (IoT) devices such as smartwatches, fitness trackers, and medical monitors continuously collect sensor data for applications ranging from human activity recognition to early disease detection [1]. Training machine learning models on such data traditionally requires centralized collection, raising privacy concerns and imposing substantial communication and compute overheads on bandwidth-constrained edge networks.

Federated Learning (FL) addresses these concerns by enabling collaborative model training while keeping data on user devices [2]. However, standard centralized FL approaches create a single point of failure and scalability bottleneck problematic for IoT deployments spanning thousands of heterogeneous devices. Decentralized Federated Learning (DFL) eliminates this bottleneck through peer-to-peer architectures where devices exchange and aggregate model updates directly [3], improving fault tolerance and reducing communication costs.

A. The Personalization Challenge

Each device's data follows a different distribution due to user diversity, sensor placement variations, environmental contexts, and device heterogeneity. This *statistical heterogeneity* creates a fundamental tension: nodes need models personalized to their local distributions, yet naive personalization through isolated training wastes the potential benefits of collaboration.

The challenge is to enable each node to learn a model that performs well on its own data distribution while still benefiting from collaboration with peers that have related but not identical data. This requires each node to answer: *which peers should I collaborate with?* A peer with very different data may hurt rather than help local model performance, even if that peer has trained effectively on their own distribution.

B. The Peer Compatibility Problem

Existing personalized FL methods typically assume centralized coordination where a server decides how to cluster or weight clients [4]. In fully decentralized settings, each node must make local decisions about peer compatibility without central oversight.

The critical challenge is distinguishing between harmful dissimilarity and beneficial diversity. Peers may have genuinely incompatible data distributions, insufficient training quality, or complementary diversity that could enhance generalization. Simple similarity metrics based on model parameter distances or gradient norms cannot capture this distinction, as they conflate distributional differences with model quality. A peer with very different but high-quality data might be valuable for improving robustness, while a peer with similar but poorly-learned data might degrade performance.

C. Our Approach: From Uncertainty to Trust

To assess peer compatibility, nodes need more than just prediction accuracy; they need to understand *why* a peer model makes its predictions. Consider evaluating a peer's model on local data: if the model predicts correctly, does that mean the peer has compatible data, or is the model simply guessing? If predictions are incorrect, does that indicate incompatibility, or is the data genuinely challenging?

Standard neural networks cannot answer these questions because they output only final predictions without indicating confidence or the type of uncertainty involved. This limitation makes it impossible to distinguish between a model that has never seen similar data (incompatible peer) versus one that finds the data inherently difficult (potentially valuable peer).

We address this by leveraging *uncertainty quantification*. When a model evaluates data, we need to separate two fundamentally different types of uncertainty: uncertainty due to lack of knowledge about the data distribution (which signals incompatibility), and uncertainty due to inherent ambiguity in the data itself (which does not). Recent advances in evidential deep learning (EDL) [5] enable exactly this decomposition by having neural networks output not just predictions, but also quantified measures of these two uncertainty types.

We propose MURMURA, a framework enabling *trust-aware model personalization* in DFL through evidential uncertainty quantification. Each node evaluates candidate peer models by running them on local validation samples and examining both predictions and uncertainty characteristics. Peers whose models show low uncertainty from lack of knowledge receive high trust scores, indicating they were trained on compatible data distributions. Peers whose models show high uncertainty from lack of knowledge are excluded, as this reveals distributional mismatch. This approach naturally handles heterogeneity: each node personalizes by choosing which peers to aggregate based on compatibility, maintaining models adapted to local distributions while benefiting from selective collaboration.

D. Contributions

Our work makes the following contributions:

- We introduce the first framework combining evidential uncertainty quantification with trust-aware personalization in DFL, enabling nodes to maintain personalized models through selective peer collaboration in non-IID environments.
- We develop an adaptive trust-aware aggregation algorithm that evaluates peer compatibility through cross-evaluation on local data and applies adaptive trust thresholds that tighten as training progresses.
- We provide MURMURA as an open, extensible framework supporting multiple network topologies, baseline aggregation methods, and systematic evaluation under varying heterogeneity levels.
- Our approach computes uncertainty with a single forward pass per evaluation, avoiding the computational overhead of ensemble methods while providing richer information than point estimates.

The remainder of this paper is organized as follows. Section II surveys related work. Section III formalizes the system model and design objectives. Section IV presents the framework architecture. Section V details the trust-aware personalization mechanism. Section VI describes implementation. Section VII presents experimental evaluation. Section VIII concludes.

II. Background and Related Work

This section surveys related work in DFL, personalization methods, and EDL, positioning MURMURA's contributions.

A. Decentralized Federated Learning

FL was introduced to enable collaborative model training while keeping data on local devices [2]. The original approach uses a central server to coordinate training rounds: clients train locally, send updates to the server, which aggregates them and broadcasts the result. While this protects data privacy, it creates centralization challenges.

Decentralized FL removes the need for a central server entirely. Instead, nodes form a network and exchange model updates directly with their neighbors. This peer-to-peer approach improves fault tolerance and can reduce communication costs by leveraging local network structure. Researchers have established both theoretical convergence guarantees [3], [6] and practical benefits including faster training on bandwidth-limited networks [7]. For IoT deployments where devices may join and leave dynamically, gossip-based communication protocols provide particularly natural patterns for information exchange [8].

B. Personalized Federated Learning

When devices have heterogeneous data distributions, forcing everyone to share a single global model often produces mediocre results. Personalized FL addresses this by letting each device maintain a model adapted to its own data while still benefiting from collaboration.

Early approaches include simple strategies like fine-tuning a global model on local data after training [9], or treating each client as a related but distinct task in a multi-task learning framework [10]. More sophisticated methods have emerged with theoretical guarantees. For example, some approaches maintain both a global model (learned collaboratively) and a personalized model (adapted locally), balancing the two with careful regularization [11], [12]. Others apply meta-learning principles to learn models that can quickly adapt to new data distributions [13].

Most personalized FL methods assume centralized coordination where a server decides how to group similar clients or weight their contributions. In decentralized settings without a server, personalization becomes harder. Recent work has explored peer selection mechanisms where nodes choose which neighbors to learn from based on task similarity [14], [15]. However, these approaches rely on heuristics such as cosine similarity of gradients or accuracy on shared test samples to assess peer compatibility rather than principled uncertainty quantification. These heuristics cannot distinguish between a peer with incompatible data and one with compatible but challenging data, limiting their effectiveness in highly heterogeneous environments.

C. Evidential Deep Learning

Standard neural network classifiers output probabilities for each class but cannot express confidence in these predictions.

TABLE I
COMPARISON OF MURMURA WITH RELATED WORK

Approach	Decentral.	Personal.	Uncertainty	Compatibility
pFedMe [11]	✗	✓	✗	✗
Ditto [12]	✗	✓	✗	✗
DisPFL [14]	✓	✓	✗	✗
PFedDST [15]	✓	✓	✗	Heuristic
D-PSGD [7]	✓	✗	✗	✗
FedUAA [17]	✗	✗	✓	Central
TPFL [18]	✗	✗	✓	Prediction-only
MURMURA	✓	✓	✓	✓

EDL [5] addresses this by having networks output evidence values that parameterize a Dirichlet distribution over probability vectors. This enables explicit quantification of epistemic uncertainty (lack of knowledge due to insufficient training or distribution shift) and aleatoric uncertainty (inherent data ambiguity). Epistemic uncertainty directly indicates whether a model has learned relevant patterns for given data, making it particularly useful for identifying compatible peers in heterogeneous FL settings. While EDL has been used in various FL contexts such as selecting reliable models [16] or weighting aggregation in centralized settings [17], no prior work has leveraged it for compatibility-based peer selection in fully decentralized personalized FL under non-IID data conditions.

D. Positioning and Research Gap

Table I compares MURMURA with related approaches across key dimensions. As the table shows, no existing work combines all four capabilities needed for our setting: decentralized operation (no server), personalization (adapted models per node), principled uncertainty quantification for compatibility assessment, and trust-aware peer selection based on distributional compatibility.

III. PROBLEM FORMULATION

This section formalizes the system model and design objectives for trust-aware model personalization in DFL under statistical heterogeneity.

A. System Model

We consider N nodes connected through an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, N\}$ and $\mathcal{E}$ denote nodes and communication links respectively. Each node i maintains a neighborhood $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\}$ of directly reachable peers. Each node i also possesses a local dataset $\mathcal{D}_i$ drawn from distribution $P_i(\mathbf{x}, y)$. We explicitly model statistical heterogeneity: $P_i \neq P_j$ in general. The learning objective is personalized models $\{\boldsymbol{\theta}_i\}_{i=1}^N$ minimizing aggregate risk:

$$\min_{\{\boldsymbol{\theta}_i\}} \sum_{i=1}^{N} \mathbb{E}_{(\mathbf{x},y)\sim P_i}[\ell(f(\mathbf{x};\boldsymbol{\theta}_i), y)] \tag{1}$$

where $f(\cdot;\boldsymbol{\theta})$ is the model and ℓ is the loss. Note each node optimizes for its own distribution P_i, enabling personalization while benefiting from selective collaboration. Training proceeds in synchronous rounds. In each round, nodes perform local training, exchange parameters with neighbors, and aggregate received updates according to their aggregation strategy.

B. Statistical Heterogeneity Model

We characterize the degree of non-IID-ness through distributional divergence between nodes. For nodes i and j, we quantify their compatibility through:

$$\Delta_{ij} = D_{\text{KL}}(P_i \| P_j) \tag{2}$$

where D_{KL} denotes Kullback-Leibler divergence. Nodes with $\Delta_{ij} \approx 0$ have similar distributions and should benefit from mutual collaboration. Nodes with large Δ_{ij} have incompatible distributions where naive aggregation would degrade personalized performance.

In practice, we do not have access to true distributions P_i and cannot compute Δ_{ij} directly. Instead, nodes must infer compatibility from empirical observations—specifically, by evaluating how peer models perform on local data. This motivates our evidential approach where epistemic uncertainty serves as a proxy for distributional mismatch.

C. Heterogeneity Sources in Wearable IoT

For wearable sensor networks, statistical heterogeneity arises naturally from multiple sources:

- **User diversity:** Different users exhibit distinct behavioral patterns.
- **Sensor placement:** Even for the same activity, sensor orientation and body placement create distributional variations.
- **Environmental context:** Indoor versus outdoor environments, terrain variations, and ambient conditions affect sensor readings.
- **Device heterogeneity:** Different sensor hardware, sampling rates, and calibration introduce systematic biases.

These factors create the non-IID conditions that necessitate personalization while making naive federated averaging suboptimal.

D. Design Objectives

We design MURMURA to simultaneously satisfy four objectives:

Objective 1: Personalized Learning. Each node should learn a model that performs well on its own local data distribution. The model at node i should minimize prediction error specifically on that node's data distribution P_i, rather than optimizing for a global average across all distributions.

Objective 2: Selective Collaboration. Nodes should benefit from collaboration with compatible peers while excluding incompatible peers whose data distributions are too dissimilar. Collaboration should improve performance over isolated training when compatible peers exist, while gracefully degrading to local learning when no compatible peers are available.

Objective 3: Principled Compatibility Assessment. Peer compatibility should be assessed through uncertainty quantification rather than heuristics or arbitrary thresholds. High epistemic uncertainty should indicate distributional incompatibility regardless of the source. High aleatoric uncertainty alone should not necessarily indicate incompatibility as it may reflect legitimately challenging data.

Objective 4: Resource Efficiency. The framework must be practical for deployment on resource-constrained wearable devices. Compatibility assessment should impose minimal computational overhead compared to training itself, and communication costs should not exceed the cost of exchanging model parameters.

The central challenge is enabling each node to make autonomous compatibility decisions in a fully decentralized setting, balancing personalization against collaboration while maintaining computational efficiency.

IV. Murmura Framework

This section presents the Murmura framework design, emphasizing the architectural decisions that enable systematic evaluation of trust-aware personalization under controlled heterogeneity.

A. Design Principles and Architecture

Addressing the objectives defined in Section III requires an architecture that supports three critical capabilities: (1) isolation of aggregation strategy from network topology to enable controlled comparison, (2) reproducible heterogeneity simulation for systematic evaluation, and (3) extensibility for integrating new personalization approaches.

Murmura achieves these goals through modular separation of concerns, as illustrated in Figure 1. The architecture comprises three layers. The Core layer separates network orchestration (managing training rounds and communication) from node-level decision making (local training and autonomous aggregation). This separation is essential: it allows individual nodes to implement different aggregation strategies while maintaining identical communication patterns, isolating the impact of personalization mechanisms from topology effects.

The Pluggable Modules layer decouples topology generation from aggregation logic. This enables systematic study of how personalization strategies perform under different network structures (ring, fully-connected, Erdős-Rényi [19], k-regular) without conflating topology and algorithm effects—a common limitation in prior work where algorithms are evaluated on fixed topologies.

The Configuration layer addresses reproducibility through declarative experiment specification. Rather than hard-coding experimental parameters, researchers specify complete experimental configurations including topology, data partitioning, and hyperparameters in YAML files. This design choice supports two key research activities: systematic parameter sweeps

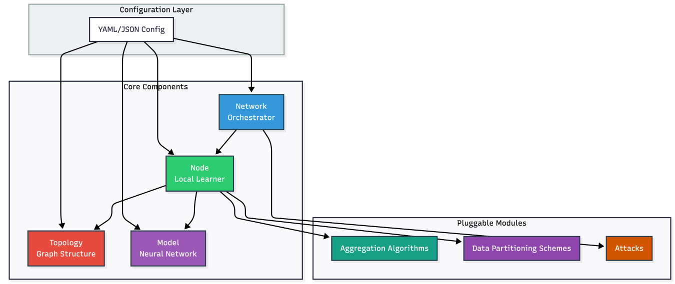

Fig. 1. Murmura framework architecture showing the three-layer design. The Configuration Layer provides YAML/JSON-based experiment specification. The Core Components layer contains the Network Orchestrator and Node (Local Learner), along with Topology and Model definitions. The Pluggable Modules layer enables flexible integration of aggregation strategies and data partitioning schemes.

(by generating configuration variants) and exact reproduction (by version-controlling complete experimental specifications).

B. Controlled Heterogeneity Simulation

A central challenge in evaluating personalized FL is creating realistic yet controlled non-IID data distributions. Murmura employs Dirichlet-based partitioning parameterized by concentration α to simulate statistical heterogeneity. Low α values create extreme heterogeneity where nodes receive samples predominantly from 1–2 classes, while higher values approach IID conditions. This parameterization enables systematic investigation of how personalization strategies degrade under increasing heterogeneity, addressing Objective 2 from Section III. Critically, the framework supports natural partitioning strategies (e.g., by subject ID for wearable datasets) alongside synthetic Dirichlet partitioning. This dual capability allows validation that findings from controlled synthetic heterogeneity generalize to realistic distribution patterns.

C. Decentralized Training Protocol

Algorithm 1 formalizes the training protocol. The key design decision is synchronous execution with parallel local training followed by parallel aggregation. While asynchronous protocols might better reflect real deployments, synchronous execution eliminates timing effects as a confounding variable, enabling cleaner isolation of personalization strategy impact.

Each round proceeds through local training, parameter exchange with neighbors, and trust-aware aggregation. The aggregation step is where personalization occurs: each node autonomously evaluates received models and selectively incorporates compatible updates. Upon completion, each node maintains a personalized model adapted to its local data distribution. The protocol's simplicity is deliberate: by standardizing communication and training patterns, we ensure that performance differences across aggregation strategies reflect algorithmic merit rather than implementation artifacts. This design directly supports Objective 3 (principled compatibility assessment) by providing a controlled environment for evaluation.

Algorithm 1 MURMURA Decentralized Training

Require: Nodes $\{1, \ldots, N\}$ with local data $\{\mathcal{D}_i\}$
Require: Topology $\mathcal{G}$ with neighborhoods $\{\mathcal{N}_i\}$
Require: Rounds T, local epochs E
for round $t = 1$ to T **do**
  for each node $i \in \mathcal{V}$ **in parallel do**
    $\boldsymbol{\theta}_i^t \leftarrow \text{LocalTrain}(\boldsymbol{\theta}_i^{t-1}, \mathcal{D}_i, E, t)$
  end for
  for each node $i \in \mathcal{V}$ **in parallel do**
    Collect $\{\boldsymbol{\theta}_j^t\}_{j \in \mathcal{N}_i}$ from neighbors
    $\boldsymbol{\theta}_i^t \leftarrow \text{TrustAwareAggregate}(i, \boldsymbol{\theta}_i^t, \{\boldsymbol{\theta}_j^t\}_{j \in \mathcal{N}_i}, t)$
  end for
end for
Output: Each node i maintains personalized model $\boldsymbol{\theta}_i^T$

V. EVIDENTIAL TRUST-AWARE PERSONALIZATION

This section explains how MURMURA uses evidential deep learning to assess peer compatibility and enable personalized model aggregation in heterogeneous data environments.

A. Evidential Uncertainty Quantification

For a K-class classification problem, evidential networks output evidence values $\mathbf{e} = [e_1, \ldots, e_K]$ via:

$$e_k = \exp(z_k), \quad \alpha_k = e_k + 1 \tag{3}$$

where z_k is the standard network logit for class k, and α_k are Dirichlet concentration parameters. The Dirichlet strength $S = \sum_{k=1}^{K} \alpha_k$ measures the total accumulated evidence and indicates overall prediction confidence. The evidential framework decomposes predictive uncertainty into two meaningful components.

Epistemic uncertainty (lack of knowledge) measures whether the model has learned patterns relevant to the data:

$$u = \frac{K}{S} \tag{4}$$

where K is the number of classes and S is the Dirichlet strength. This uncertainty ranges from 0 (high confidence) to 1 (complete ignorance). High epistemic uncertainty signals that the model has not seen similar data during training, indicating distributional mismatch, which is precisely the signal needed to identify incompatible peers.

Aleatoric uncertainty (data ambiguity) captures inherent difficulty via prediction entropy:

$$H[\hat{\mathbf{p}}] = -\sum_{k=1}^{K} \frac{\alpha_k}{S} \log \frac{\alpha_k}{S} \tag{5}$$

High aleatoric uncertainty indicates genuinely ambiguous inputs (e.g., transitional movements between activities) rather than distributional incompatibility.

Algorithm 2 Computing Compatibility via Cross-Validation

Require: Validation set $\mathcal{D}_i^{\text{val}}$, peer model $\boldsymbol{\theta}_j$
Require: Uncertainty threshold τ_u, accuracy weight w_a
Evaluate peer model on validation samples
Compute mean epistemic uncertainty $\bar{u}$ and accuracy
Base trust score: $\text{trust}_j = (1 - \bar{u}) \cdot (w_a \cdot \text{acc} + (1 - w_a))$
if $\bar{u} > \tau_u$ **then**
  Apply penalty: $\text{trust}_j \leftarrow \text{trust}_j \cdot \exp(-(\bar{u} - \tau_u))$
end if
return trust_j

This distinction is crucial for personalization: a well-trained peer model evaluated on incompatible data shows high epistemic uncertainty (revealing distributional mismatch), whereas the same model on compatible but challenging data shows low epistemic uncertainty despite high aleatoric uncertainty. This enables principled compatibility assessment.

B. Training Evidential Models

Evidential models require a specialized loss function that encourages accurate predictions while preventing the model from simply claiming high confidence everywhere. Following Sensoy et al. [5], we use:

$$\mathcal{L} = \sum_{k=1}^{K} (y_k - \hat{p}_k)^2 + \lambda_t \cdot \text{KL}[\text{Dir}(\tilde{\boldsymbol{\alpha}}) \| \text{Dir}(\mathbf{1})] \tag{6}$$

where $\hat{p}_k = \alpha_k / S$ is the expected probability for class k, and the KL term prevents accumulating evidence for incorrect classes. The coefficient λ_t gradually increases during training to allow initial learning before regularizing uncertainty.

C. Computing Compatibility from Uncertainty

When node i receives a model from peer j, it evaluates compatibility by running the peer's model on its own validation data. The evaluation examines both prediction accuracy and uncertainty characteristics, with epistemic uncertainty serving as the primary compatibility signal. Algorithm 2 formalizes this process. For each validation sample, the peer model produces both a prediction and an epistemic uncertainty score. High epistemic uncertainty indicates the peer's model was not trained on similar data, revealing distributional incompatibility regardless of whether predictions happen to be correct. The trust score naturally handles heterogeneity: a well-trained peer with compatible data shows low epistemic uncertainty (high trust) even if there are occasional prediction errors. A peer with incompatible data shows high epistemic uncertainty (low trust) regardless of prediction accuracy, as the peer's model was trained on different distributional patterns.

D. Adaptive Trust Thresholds

Trust thresholds should adapt as training progresses. Early in training, all models exhibit high uncertainty simply because they have not converged. Applying strict thresholds would reject all peers, preventing collaboration. As training proceeds, models accumulate evidence on their respective distributions

Algorithm 3 Trust-Aware Model Aggregation

Require: Node i, local model $\boldsymbol{\theta}_i$, neighbor models $\{\boldsymbol{\theta}_j\}_{j\in\mathcal{N}_i}$
Require: Validation set $\mathcal{D}_i^{\text{val}}$, self-weight α, round t
1: Compute adaptive threshold $\tau_{\min}^{(t)}$
2: Evaluate each neighbor, retaining only those with $\text{trust}_j \geq \tau_{\min}$
3: **if** no peers pass threshold **then**
4: **return** $\boldsymbol{\theta}_i$ {Keep local model}
5: **end if**
6: Normalize trust scores: $w_j = \text{trust}_j / \sum_k \text{trust}_k$
7: Aggregate trusted peers: $\boldsymbol{\theta}_{\text{peers}} = \sum_j w_j \cdot \boldsymbol{\theta}_j$
8: Personalize: $\boldsymbol{\theta}_i^{\text{new}} = \alpha \cdot \boldsymbol{\theta}_i + (1-\alpha) \cdot \boldsymbol{\theta}_{\text{peers}}$
9: **return** $\boldsymbol{\theta}_i^{\text{new}}$

and uncertainty decreases for in-distribution data. Following principles from our prior work [20], we implement adaptive tightening:

$$\tau_{\min}^{(t)} = \tau_{\min}^{(0)} \cdot \left(1 - \gamma_\tau \cdot e^{-\kappa \cdot t/T}\right) \tag{7}$$

where t is the current round and T is the total number of rounds. This starts lenient (accepting peers despite high uncertainty) and gradually tightens (requiring lower uncertainty), matching the expected reduction in epistemic uncertainty as models converge on their respective data distributions. Optional exponential moving average smoothing reduces noise in trust scores across rounds, providing temporal stability to compatibility assessments.

E. Trust-Aware Aggregation

Each node independently decides which peers to trust and how much weight to assign them. Algorithm 3 shows the complete procedure. If no peers pass the compatibility threshold, the node simply retains its local model, providing graceful degradation to isolated learning. This autonomous decision-making enables personalization through selective collaboration rather than forced consensus or complete isolation. Compatibility assessment requires one forward pass per validation sample per neighbor, yielding complexity $O(d \cdot |\mathcal{D}_i^{\text{val}}| \cdot C_{\text{forward}})$ for d neighbors. The evidential output layer adds minimal overhead (typically under 1% additional inference time compared to standard classification).

VI. Implementation

This section describes the MURMURA implementation, detailing the software architecture, key components, and practical considerations for deployment on resource-constrained wearable devices.

A. Software Architecture and Design Principles

MURMURA is implemented in Python 3.12 using PyTorch [21] as the deep learning backend. The framework adopts a modular, layered architecture that cleanly separates concerns and facilitates extensibility. The codebase organization reflects the conceptual model presented in Section IV, with each major component residing in its own package.

The core layer (murmura/core/) implements fundamental abstractions including the Network orchestrator, Node implementation, and type definitions for model states and data partitions. The topology layer (murmura/topology/) provides graph generators for ring, fully-connected, Erdős-Rényi, and k-regular topologies, all implementing a common Topology interface. The aggregation layer (murmura/aggregation/) contains implementations of baseline aggregators alongside our evidential trust aggregator, all adhering to a unified Aggregator protocol. The data layer (murmura/data/) supplies dataset adapters and partitioning utilities supporting both standard benchmarks and wearable sensor datasets. Finally, the configuration layer (murmura/config/) defines Pydantic schemas for type-safe, validated experiment specifications.

This modular design enables systematic comparison of aggregation strategies under identical conditions, facilitates integration of new datasets and topologies, and supports reproducible experimentation through declarative configuration.

B. Evidential Deep Learning Components

The evidential modeling capability requires two key components: an evidential output layer and a specialized loss function. The evidential output layer replaces the standard softmax head with a module that produces Dirichlet concentration parameters. This layer consists of a linear transformation from feature space to logit space, followed by an exponential activation ensuring non-negative evidence values, and finally the addition of one to obtain valid Dirichlet parameters greater than zero.

This modification is remarkably lightweight, adding only a single linear layer and activation function beyond a standard classifier. Evidential models compute uncertainty in a single forward pass, avoiding the computational overhead of ensemble methods (which require 5+ model evaluations) or Monte Carlo dropout (requiring dozens of stochastic passes) while providing richer information than point estimates. The computational overhead is empirically less than 1% additional inference time compared to softmax classification.

The evidential loss function combines two terms as described in Section V. The MSE component encourages accurate expected probability predictions, while the KL divergence component regularizes evidence accumulation on incorrect classes. Critically, the KL weight λ anneals from zero to its maximum value over the first half of training. This annealing schedule allows the model to first learn accurate predictions before regularizing uncertainty estimates, preventing premature commitment to low-confidence predictions. In FL contexts, we use rounds rather than epochs for annealing since nodes may perform variable numbers of local epochs.

C. Trust-Aware Aggregator Implementation

The evidential trust aggregator implements Algorithm 3 with practical enhancements for stability and efficiency. Trust scores are computed through cross-evaluation with evaluation limited to 100 validation samples to bound computational cost per neighbor. For networks with high connectivity, this limitation ensures that compatibility assessment remains tractable while providing sufficient statistical confidence in uncertainty estimates.

The adaptive trust threshold implements tightening as described in Section V. Optional exponential moving average (EMA) smoothing provides temporal stability to trust assessments. Per-neighbor trust histories are maintained, and new trust scores are blended with historical values using momentum parameter $\gamma \in (0, 1)$. Higher momentum values ($\gamma > 0.7$) enable rapid adaptation to changing peer behavior. Lower values ($\gamma < 0.5$) provide stability against noisy evaluations, beneficial in high-heterogeneity settings where trust scores may fluctuate due to legitimate distribution differences rather than reliability changes.

The self-weight parameter α controlling personalization admits tuning based on expected heterogeneity. In homogeneous settings where all nodes have similar distributions, lower α values ($\alpha < 0.3$) enable aggressive collaboration. In highly heterogeneous settings, higher values ($\alpha > 0.7$) preserve personalization by limiting peer influence. Our experiments suggest $\alpha = 0.5$ provides a reasonable default balancing collaboration and personalization across varied heterogeneity levels.

D. Wearable Sensor Dataset Integration

A key challenge in FL research is bridging the gap between standard benchmark datasets and realistic sensor data characteristics. Wearable datasets often require preprocessing (sliding window segmentation, feature extraction, signal synchronization) that, if performed differently across studies, confounds experimental comparisons.

MURMURA addresses this through dataset adapters that standardize preprocessing while exposing key parameters (window size, stride, feature sets) as configuration options. This design enables researchers to systematically study how preprocessing choices interact with personalization strategies.

The framework provides adapters for three established wearable datasets (UCI HAR [22], PAMAP2 [23], PPG-DaLiA [24]), each supporting multiple partitioning strategies. Critically, adapters support both synthetic Dirichlet partitioning (for controlled heterogeneity) and natural partitioning by subject ID (for realistic distribution patterns). This dual capability enables validation that findings from controlled experiments generalize to deployment-like conditions where heterogeneity arises from genuine user diversity.

The concentration parameter α in Dirichlet partitioning governs heterogeneity severity: $\alpha = 0.1$ creates extreme non-IID conditions where each node receives samples from only 1–2 classes, $\alpha = 0.5$ produces moderate heterogeneity, and $\alpha = 1.0$ approaches IID. This parameterized control enables systematic investigation of personalization performance across the heterogeneity spectrum, directly supporting the evaluation objectives in Section VII.

E. Reproducibility and Experimental Infrastructure

Reproducibility is ensured through comprehensive seed management and configuration capture. At experiment initialization, seeds are set deterministically for Python's built-in random number generator, NumPy's random state, PyTorch's CPU random number generator, and all CUDA random number generators if GPU execution is enabled. Additionally, PyTorch's cuDNN backend is configured for deterministic operation, trading a small performance penalty for perfect reproducibility across runs.

All experimental parameters are specified via YAML configuration files following the schema defined in Section IV. This declarative approach enables version control of experimental configurations, facilitates parameter sweeps through configuration templates, and ensures complete reproducibility by capturing all relevant settings.

VII. PERFORMANCE EVALUATION

This section presents experimental evaluation of MURMURA, examining personalization performance under varying data heterogeneity, convergence behavior, and sensitivity to hyperparameter choices.

A. Experimental Setup

1) Datasets: We evaluate on three established wearable sensor datasets for human activity recognition:

- **UCI HAR** [22]: Smartphone accelerometer and gyroscope data from 30 subjects performing 6 activities (walking, walking upstairs, walking downstairs, sitting, standing, lying). Contains 10,299 samples with 561 extracted features.
- **PAMAP2** [23]: IMU data from 9 subjects performing 12 activities including household and exercise tasks. We extract 40 features per sliding window from three body-worn sensors.
- **PPG-DaLiA** [24]: Photoplethysmography and accelerometer signals from 15 subjects under real-life conditions. We formulate activity classification from 8 activity types with time-frequency features.

2) Baselines: We compare against four representative aggregation strategies:

- **FedAvg** [2]: Standard federated averaging without Byzantine resilience or personalization mechanisms.
- **BALANCE** [25]: Distance-based filtering with adaptive thresholds for Byzantine resilience.
- **Sketchguard** [20]: Count-Sketch compression with cosine similarity filtering for communication-efficient Byzantine resilience.
- **UBAR** [26]: Two-stage filtering combining distance-based screening with loss-based evaluation.

3) Implementation Details: We deploy 30 nodes in a fully-connected topology, training for 30 rounds with 5 local epochs per round. Models use a three-layer fully-connected architecture with ReLU activations and an evidential output layer for MURMURA. We use SGD with learning rate 0.01 and batch size 32. For evidential trust aggregation, we set self-weight

TABLE II
CLASSIFICATION ACCURACY (%) UNDER VARYING HETEROGENEITY

Algorithm	α	**UCI HAR**	**PAMAP2**	**PPG-DaLiA**
	0.1	85.6	90.5	26.6
FedAvg	0.5	97.9	92.6	66.5
	1.0	98.5	89.2	72.9
	0.1	75.3	98.4	52.3
BALANCE	0.5	98.9	95.8	72.2
	1.0	99.2	84.0	67.6
	0.1	94.4	97.9	54.2
Sketchguard	0.5	98.8	95.6	69.3
	1.0	99.0	92.1	67.6
	0.1	85.6	87.5	75.9
UBAR	0.5	96.3	94.0	75.1
	1.0	97.4	86.3	73.2
	0.1	**95.4**	**98.8**	63.9
Murmura	0.5	98.2	97.4	**78.8**
	1.0	98.4	90.0	72.6

$\alpha = 0.5$, accuracy weight $w_a = 0.5$, and initial trust threshold $\tau = 0.3$. All experiments use 5 random seeds; we report mean accuracy and standard deviation across nodes.

B. Research Questions

Our evaluation addresses three research questions:

RQ1: How does evidential trust-aware aggregation affect model personalization under varying degrees of data heterogeneity?

RQ2: Does uncertainty-based peer selection improve convergence speed compared to baseline aggregation strategies?

RQ3: How sensitive is the framework to hyperparameter choices?

C. RQ1: Personalization Under Heterogeneity

Table II presents classification accuracy across datasets and heterogeneity levels. Under high heterogeneity ($\alpha = 0.1$), MURMURA achieves the highest accuracy on UCI HAR (95.4%) and PAMAP2 (98.8%), demonstrating effective personalization when data distributions diverge substantially across nodes. PPG-DaLiA exhibits lower absolute accuracy across all methods; we discuss the dataset-specific challenges and their implications in Section VII-F. The low standard deviation (7.3% on UCI HAR compared to 18.4% for FedAvg) indicates consistent performance across nodes despite heterogeneous local distributions.

Figure 2 visualizes performance across heterogeneity levels, averaged across all datasets. MURMURA maintains stable accuracy as heterogeneity increases, while baseline methods exhibit more pronounced degradation. This stability stems from the trust-aware mechanism: under high heterogeneity, nodes with incompatible distributions exhibit elevated epistemic uncertainty during cross-evaluation, triggering exclusion from aggregation. Nodes effectively "choose" compatible peers, preserving personalized models.

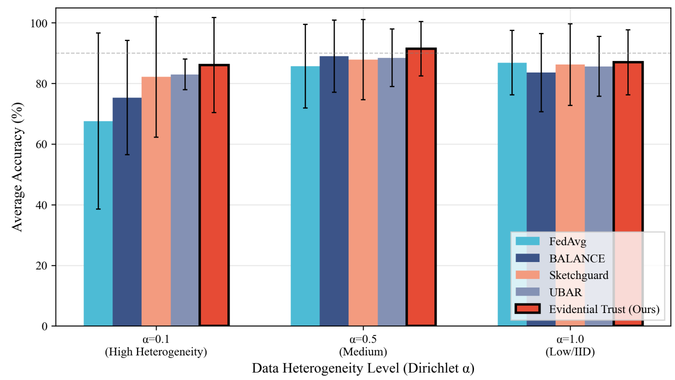

Fig. 2. Model accuracy across data heterogeneity levels (Dirichlet α), averaged across all three datasets. Lower α indicates higher heterogeneity. MURMURA (Evidential Trust) maintains consistent performance as heterogeneity increases.

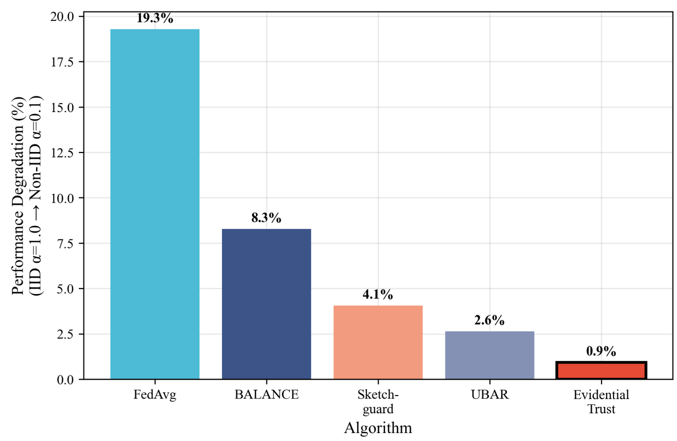

Fig. 3. Performance degradation from IID ($\alpha = 1.0$) to non-IID ($\alpha = 0.1$) conditions. Lower values indicate better robustness to heterogeneity. MURMURA shows minimal degradation (0.9%) compared to baselines.

To quantify robustness to heterogeneity, we measure *performance degradation*: the accuracy drop when transitioning from IID ($\alpha = 1.0$) to highly non-IID ($\alpha = 0.1$) conditions. Figure 3 presents these results. MURMURA exhibits the smallest degradation (0.9% average across datasets), compared to FedAvg (19.3%), BALANCE (8.3%), Sketchguard (4.1%), and UBAR (2.6%). This 20$\times$ reduction in degradation compared to FedAvg demonstrates that evidential trust-aware aggregation effectively isolates nodes from incompatible peer influence.

The personalization quality is further evidenced by examining per-node variance. Figure 4 shows accuracy with standard deviation across nodes at $\alpha = 0.1$. MURMURA achieves not only higher mean accuracy but also lower variance, indicating that personalization benefits all nodes rather than improving some at the expense of others.

D. RQ2: Convergence Analysis

Figure 5 presents convergence speed measured as rounds required to reach peak accuracy. MURMURA converges in 4.0 rounds on average, compared to 29.5 rounds for FedAvg—a 7.4$\times$ speedup. This acceleration arises from two factors: (1)

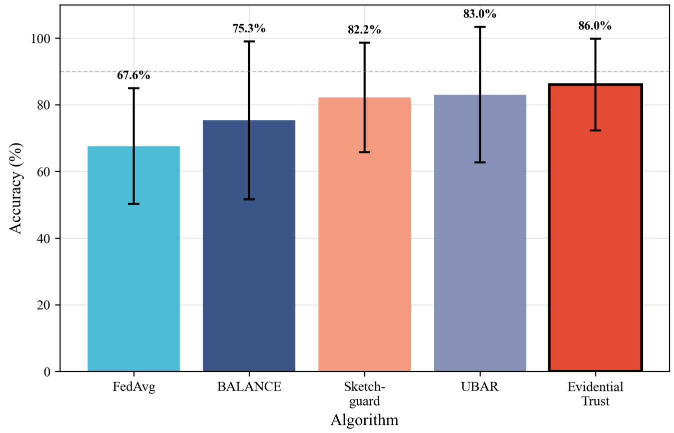

Fig. 4. Model personalization under high heterogeneity ($\alpha = 0.1$). Bars show mean accuracy; error bars show standard deviation across nodes. MURMURA achieves high accuracy with low variance across nodes.

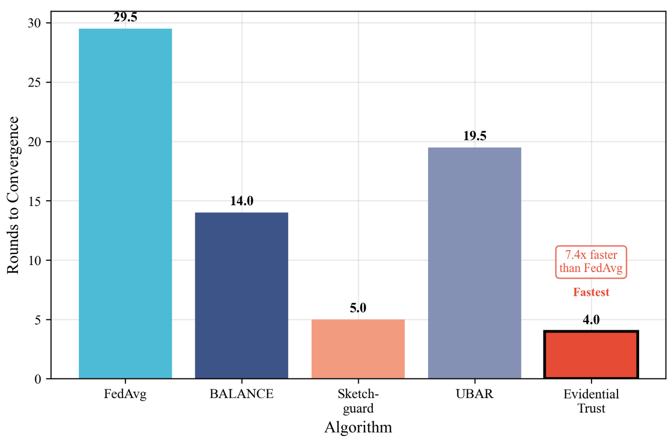

Fig. 5. Convergence speed comparison showing rounds to reach peak accuracy. MURMURA converges 7.4× faster than FedAvg by filtering incompatible peer updates.

evidential trust filtering excludes updates from incompatible or undertrained peers that would otherwise introduce noise, and (2) the personalized aggregation weights prioritize high-quality updates from compatible peers.

The fast convergence has practical implications for wearable IoT deployments: fewer communication rounds reduce energy consumption and network bandwidth usage, critical constraints for battery-powered devices. Combined with the single-pass uncertainty computation, MURMURA provides an efficient path to personalized models.

E. RQ3: Hyperparameter Sensitivity

Table III presents ablation results examining sensitivity to key hyperparameters. For each parameter, we vary its value while holding others constant at defaults, reporting mean accuracy and standard deviation across the tested range.

The results demonstrate remarkable stability across hyperparameter choices. Standard deviations remain below 1% for UCI HAR and PAMAP2 across all parameters, and below 4% for the more challenging PPG-DaLiA dataset.

TABLE III
ABLATION STUDY: HYPERPARAMETER SENSITIVITY

Parameter	**UCI HAR**	**PAMAP2**	**PPG-DaLiA**
Accuracy weight (λ)	98.3 ± 0.3	97.1 ± 0.2	78.6 ± 0.4
Self-weight (ω)	98.3 ± 0.5	96.8 ± 1.4	79.1 ± 3.7
Trust threshold (τ)	98.4 ± 0.2	97.3 ± 0.2	78.9 ± 0.2
Vacuity threshold (ν)	98.4 ± 0.2	97.2 ± 0.1	78.7 ± 0.1

The *accuracy weight* ($\lambda \in \{0.3, 0.5, 0.7, 0.9\}$) balances prediction accuracy against uncertainty in trust computation; performance remains stable across this range. The *self-weight* ($\omega \in \{0.3, 0.5, 0.6, 0.7, 0.9\}$) controls personalization strength; higher values favor local models while lower values enable more collaboration. The *trust threshold* ($\tau \in \{0.05, 0.1, 0.2, 0.3\}$) and *vacuity threshold* ($\nu \in \{0.3, 0.5, 0.7, 0.9\}$) govern peer filtering stringency; the framework remains robust across a wide range. This stability reduces the need for extensive hyperparameter tuning in deployment, an important practical consideration for wearable IoT applications where per-device optimization is infeasible.

F. Discussion

The experimental results support several conclusions regarding evidential trust-aware personalization:

Personalization emerges from selective collaboration. Rather than explicitly clustering nodes or learning separate models, MURMURA achieves personalization through autonomous peer selection. Each node's model naturally adapts to its local distribution by incorporating updates only from compatible peers, while still benefiting from collaboration when compatible peers exist.

Fast convergence reduces resource requirements. The considerable speedup over baselines translates directly to reduced communication rounds, energy consumption, and time-to-deployment, all of which are critical factors for battery-constrained wearable devices.

Robustness simplifies deployment. The stability across hyperparameter choices reduces the burden of per-deployment tuning, making MURMURA practical for heterogeneous IoT environments where individual device optimization is impractical.

The results on PPG-DaLiA, while showing improvement over baselines at moderate heterogeneity, reveal limitations under extreme heterogeneity. This dataset exhibits substantial inter-subject variability in physiological signals, creating scenarios where few truly compatible peers exist. In such cases, MURMURA's conservative filtering leads to near-isolated training, which, while preserving personalization, limits knowledge transfer. Future work could explore graduated trust mechanisms that extract partial value from moderately compatible peers.

VIII. Conclusion and Future Work

We presented MURMURA, a framework enabling trust-aware model personalization in decentralized federated learning through evidential uncertainty quantification. The key insight is that epistemic uncertainty from evidential models directly indicates peer compatibility: high uncertainty when evaluating peer models on local data reveals distributional mismatch, enabling nodes to maintain personalized models through selective collaboration rather than forced consensus.

Experimental evaluation on three wearable IoT datasets demonstrates substantial improvements over baseline methods. MURMURA reduces performance degradation from IID to non-IID conditions by $20\times$ compared to FedAvg (0.9% vs 19.3%), converges $7.4\times$ faster, and maintains stable accuracy across hyperparameter choices. The framework's single-pass uncertainty computation and modular architecture make it suitable for resource-constrained wearable deployments.

Future work should investigate temporal dynamics where peer compatibility may evolve over time, enabling adaptive personalization in changing environments. Deployment on physical wearable devices would validate practical applicability and reveal real-world performance characteristics. Additionally, graduated trust mechanisms that extract partial value from moderately compatible peers could improve performance when few highly compatible peers exist.

MURMURA demonstrates that evidential deep learning provides a principled foundation for compatibility-aware personalization in decentralized settings, enabling collaborative learning that respects the natural heterogeneity of distributed data while achieving significant improvements in convergence speed and personalization quality.

Code Availability

The complete implementation of MURMURA is available as open-source software at https://github.com/Cloudslab/murmura.

Acknowledgment

This work is supported by the Australian Research Council (ARC) through Discovery Project grant DP240102088.

References

[1] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

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

[3] A. Lalitha, S. Shekhar, T. Javidi, and F. Koushanfar, "Fully decentralized federated learning," in *Third workshop on bayesian deep learning (NeurIPS)*, vol. 12, 2018.

[4] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE transactions on neural networks and learning systems*, vol. 34, no. 12, pp. 9587–9603, 2022.

[5] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," *Advances in neural information processing systems*, vol. 31, 2018.

[6] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, "A unified theory of decentralized sgd with changing topology and local updates," in *International conference on machine learning*. PMLR, 2020, pp. 5381–5393.

[7] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[8] I. Hegedűs, G. Danner, and M. Jelasity, "Gossip learning as a decentralized alternative to federated learning," in *IFIP International Conference on Distributed Applications and Interoperable Systems*. Springer, 2019, pp. 74–90.

[9] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," *arXiv preprint arXiv:1910.10252*, 2019.

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

[11] C. T Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with moreau envelopes," *Advances in neural information processing systems*, vol. 33, pp. 21 394–21 405, 2020.

[12] 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.

[13] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in neural information processing systems*, vol. 33, pp. 3557–3568, 2020.

[14] R. Dai, L. Shen, F. He, X. Tian, and D. Tao, "Dispfl: Towards communication-efficient personalized federated learning via decentralized sparse training," in *International Conference on Machine Learning*. PMLR, 2022, pp. 4587–4604.

[15] M. Fan, K. Li, T. Zhang, Q. Tian, and B. Geng, "Pfeddst: Personalized federated learning with decentralized selection training," *arXiv preprint arXiv:2502.07750*, 2025.

[16] H. Zhou, W. Chen, L. Cheng, J. Liu, and M. Xia, "Trustworthy fault diagnosis with uncertainty estimation through evidential convolutional neural networks," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 11, pp. 10 842–10 852, 2023.

[17] M. Wang, L. Wang, X. Xu, K. Zou, Y. Qian, R. S. M. Goh *et al.*, "Federated uncertainty-aware aggregation for fundus diabetic retinopathy staging," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2023, pp. 222–232.

[18] J. Chen and J. Zhu, "Tpfl: A trustworthy personalized federated learning framework via subjective logic," *arXiv preprint arXiv:2410.12316*, 2024.

[19] P. Erdős and A. Rényi, "On the evolution of random graphs," *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.

[20] M. Rangwala, F. Azzedin, R. O. Sinnott, and R. Buyya, "Sketchguard: Scaling byzantine-robust decentralized federated learning via sketch-based screening," *arXiv preprint arXiv:2510.07922*, 2025.

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

[22] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. L. Reyes-Ortiz *et al.*, "A public domain dataset for human activity recognition using smartphones." in *Esann*, vol. 3, no. 1, 2013, pp. 3–4.

[23] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *2012 16th international symposium on wearable computers*. IEEE, 2012, pp. 108–109.

[24] A. Reiss, I. Indlekofer, P. Schmidt, and K. Van Laerhoven, "Deep ppg: Large-scale heart rate estimation with convolutional neural networks," *Sensors*, vol. 19, no. 14, p. 3079, 2019.

[25] M. Fang, Z. Zhang, Hairi, P. Khanduri, J. Liu, S. Lu, Y. Liu, and N. Gong, "Byzantine-robust decentralized federated learning," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 2874–2888.

[26] S. Guo, T. Zhang, H. Yu, X. Xie, L. Ma, T. Xiang, and Y. Liu, "Byzantine-resilient decentralized stochastic gradient descent," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 4096–4106, 2021.