mathematics

MDPI

Article

Distributed Collaborative Learning with Representative Knowledge Sharing

Joseph Casey [1], Qianjiao Chen [1], Mengchen Fan [2], Baocheng Geng [2], Roman Shterenberg [1], Zhong Chen [3] and Keren Li [1,*]

[1] Department of Mathematics, University of Alabama at Birmingham, Birmingham, AL 35294, USA
[2] Department of Computer Science, University of Alabama at Birmingham, Birmingham, AL 35294, USA
[3] School of Computing, Southern Illinois University, Carbondale, IL 62901, USA
* Correspondence: kli@uab.edu

Abstract: Distributed Collaborative Learning (DCL) addresses critical challenges in privacy-aware machine learning by enabling indirect knowledge transfer across nodes with heterogeneous feature distributions. Unlike conventional federated learning approaches, DCL assumes non-IID data and prediction task distributions that span beyond local training data, requiring selective collaboration to achieve generalization. In this work, we propose a novel collaborative transfer learning (CTL) framework that utilizes representative datasets and adaptive distillation weights to facilitate efficient and privacy-preserving collaboration. By leveraging Energy Coefficients to quantify node similarity, CTL dynamically selects optimal collaborators and refines local models through knowledge distillation on shared representative datasets. Simulations demonstrate the efficacy of CTL in improving prediction accuracy across diverse tasks while balancing trade-offs between local and global performance. Furthermore, we explore the impact of data spread and dispersion on collaboration, highlighting the importance of tailored node alignment. This framework provides a scalable foundation for cross-domain generalization in distributed machine learning.

Keywords: collaborative transfer learning; knowledge distillation; contrastive learning; federated learning

MSC: 68W15; 62H10; 62E17; 68T99

check for updates

Academic Editor: Jinfeng Xu

Received: 21 February 2025
Revised: 16 March 2025
Accepted: 17 March 2025
Published: 19 March 2025

Citation: Casey, J.; Chen, Q.; Fan, M.; Geng, B.; Shterenberg, R.; Chen, Z.; Li, K. Distributed Collaborative Learning with Representative Knowledge Sharing. *Mathematics* **2025**, *13*, 1004. https://doi.org/10.3390/math13061004

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Distributed machine learning (DML) seeks to leverage data and computation distributed across multiple nodes to collaboratively train models while preserving privacy and scalability. Traditional approaches to DML often encounter challenges from heterogeneity, where variability in feature distributions, response distributions, or both leads to misalignment between local models and global objectives. These challenges necessitate methods that enable effective collaboration without compromising privacy.

A particularly complex setting arises when raw data and model parameters cannot be exchanged, and the prediction task distribution spans a broader space than any single node's training data. This situation occurs in competitive AI development, where organizations (e.g., developing systems like ChatGPT) share performance metrics on public datasets without revealing proprietary models. Similarly, a model trained on general-purpose data may need to adapt to a new domain, such as healthcare, by leveraging insights from external models trained on medical data, even without direct access to sensitive records. These

challenges motivate the development of distributed learning approaches that preserve privacy while enabling cross-domain generalization.

A key application of CTL is in privacy-sensitive domains such as healthcare and competitive AI development. In healthcare, hospitals and research institutions must train predictive models without sharing sensitive patient data due to privacy regulations like HIPAA. CTL enables collaboration by allowing institutions to exchange only predictions on a representative dataset rather than raw patient records or model parameters. Similarly, in AI model development, organizations can improve model performance by distilling knowledge from external sources without revealing proprietary training data. These applications demonstrate how CTL facilitates privacy-preserving collaboration while enabling cross-institutional learning.

Recent advances in Distributed Collaborative Learning (DCL) aim to address such challenges by facilitating indirect knowledge sharing across nodes. Notably, large-scale models like DeepSeek have successfully leveraged knowledge distillation to enhance training efficiency while maintaining model performance [1]. This highlights the viability of distillation-based approaches for enabling collaboration among distributed nodes with privacy constraints. Unlike Multi-Task Learning (MTL), which assumes distinct objectives for each node, DCL assumes an underlying shared conditional distribution distorted by localized heterogeneity. By leveraging shared representative datasets, DCL allows nodes to exchange predictions instead of raw data or model parameters, enabling privacy-preserving collaboration and reducing communication overhead for large models.

One promising method within this paradigm involves the use of representative datasets to identify "qualified teachers" among nodes. Nodes leverage predictions on these datasets to assess alignment with their objectives, enabling selective knowledge distillation. This strategy enhances model generalization across heterogeneous tasks, particularly when prediction task distributions exceed the scope of local training data. Inspired by the success of knowledge transfer methods such as distillation [2], this approach facilitates efficient convergence and ensures generalization to non-local tasks. Knowledge distillation has already demonstrated its effectiveness in large-scale collaborative training environments, as seen in DeepSeek, where models are trained efficiently using teacher–student learning paradigms. This underscores the role of distillation in reducing communication overhead and enabling scalable distributed learning.

This work explores how collaborative transfer learning principles can address these challenges, providing a scalable and privacy-aware framework for distributed learning. By combining representative datasets with adaptive distillation weights, we demonstrate how nodes can effectively align heterogeneous feature distributions, select reliable collaborators, and improve model performance across diverse prediction tasks.

2. Challenges and Advances in Distributed Collaborative Learning

DCL builds upon foundational paradigms like federated learning and transfer learning, while addressing unique challenges in privacy-aware and heterogeneous distributed systems.

2.1. Federated Learning (FL)

FL enables decentralized model training by aggregating locally updated model parameters into a global model, without sharing raw data [3]. A notable example is the FedAvg algorithm, which averages client updates using Stochastic Gradient Descent (SGD). While FedAvg laid the groundwork for FL, its assumption of IID data distributions often fails in real-world scenarios, leading to performance degradation due to client heterogeneity [4].

Several methods have been developed to address this limitation. Li et al. [5] introduced a proximal regularization term to align client updates with the global model, reducing

the impact of heterogeneity. Similarly, Tan et al. [6] proposed AdaFed, which adapts aggregation weights dynamically based on client participation, improving both global accuracy and convergence speed. For faster convergence, Xie et al. [7] demonstrated an asynchronous approach that achieved near-linear convergence for convex and certain non-convex problems. Other advancements include leveraging dataset size ratios for model consistency [8] and incorporating dynamic regularization [9] to adjust local objectives dynamically, further mitigating model drift.

2.2. Personalized Federated Learning (PFL)

PFL extends FL by tailoring models to individual client distributions, addressing the one-size-fits-all limitation of standard FL. Approaches like KN-Fu [10] and parameterized strategies [11] introduce similarity metrics, such as soft activation functions, to measure alignment between client models and enable personalization. These strategies emphasize the need for adaptive weights to balance global collaboration with local personalization.

Tan et al. [12] proposed a framework that incorporates adaptive regularization techniques for personalized model training, bridging the gap between global and client-specific models. Their work highlights the importance of addressing client heterogeneity by dynamically adjusting the balance between shared and personalized components, an idea central to our use of adaptive distillation weights.

Additionally, Jeong et al. [13] proposed FedMatch, which decomposes model parameters to optimize communication between supervised and unsupervised learning components, further improving the effectiveness of personalized training.

2.3. Knowledge Distillation and Representative Datasets

Knowledge distillation, a form of transfer learning, facilitates knowledge sharing by compressing complex models into smaller, efficient ones. This technique has proven particularly valuable in large-scale distributed training, as demonstrated by DeepSeek, which integrates knowledge distillation to optimize model convergence and maintain high performance while reducing computational overhead [1]. In a collaborative learning setting, distillation enables nodes to transfer knowledge without sharing raw data or full model parameters, making it a strong alternative to traditional model aggregation in federated learning.

Recent advances have demonstrated the importance of adaptive weighting in knowledge transfer, where confidence-based or similarity-based methods improve the effectiveness of model distillation [14,15]. This aligns with importance sampling techniques in transfer learning and domain adaptation, where distillation weights reflect the reliability of each teacher node [16,17]. Prior works [2,18] emphasize that knowledge transfer effectiveness depends on teacher–student alignment, motivating our approach of using Energy Coefficients for adaptive distillation weighting.

In federated learning, dynamic model importance weighting has been explored in heterogeneous FL settings [19,20], where similarity-based aggregation improves personalization.

A notable related work is FedED (Federated Ensemble Distillation) [21], which transfers knowledge via ensembles of local models instead of a single teacher–student relationship. While FedED aggregates predictions from multiple clients, our method personalizes distillation weights using Energy Coefficients, ensuring knowledge transfer is selectively tailored to each node's task distribution. This distinction is particularly critical in heterogeneous settings where generalization beyond local training data is essential.

Hinton et al. [2] first introduced this concept, demonstrating how teacher–student frameworks could retain high performance. Subsequent works refined this approach: Passalis and Tefas [22] matched probability distributions of teacher and student models in feature space, Huang and Wang [23] introduced loss functions minimizing distributional

discrepancies, and Tian et al. [18] incorporated contrastive learning to align teacher–student representations effectively.

In distributed settings, the use of representative datasets provides a practical alternative to raw data or parameter sharing. Li and Yang [24] constructed representative data points for large datasets to enable efficient training and alignment. Similarly, FedKD [25] used knowledge distillation to facilitate communication-free collaboration, where student models learned from teacher models across clients.

These ideas, along with the success of large-scale implementations like DeepSeek, inspired our use of representative datasets for privacy-preserving collaboration in heterogeneous distributed environments. By distilling knowledge from models trained on different local distributions, our approach mitigates the impact of heterogeneity and enhances generalization across diverse prediction tasks.

2.4. Distributed Collaborative Learning (DCL)

DCL builds on FL and knowledge distillation by focusing on indirect knowledge sharing to address feature and response heterogeneity. Unlike FL, DCL does not aggregate models into a single global model but enables nodes to learn from aligned predictions. Recent advances, such as FedProc [26], demonstrate how prototypes and contrastive loss can guide local objectives towards a global optimum. Similarly, our work incorporates adaptive distillation weights informed by the Energy Coefficient to improve inter-node alignment.

A notable distinction in our approach is the use of a representative dataset to measure inter-node similarity and select reliable teachers. This concept is particularly valuable when prediction task distributions span broader spaces than local training distributions, a common challenge in heterogeneous environments. By evaluating teacher–student alignment through predictions on public datasets, our framework ensures that collaboration enhances model performance while respecting privacy constraints.

The key challenges in DCL involve measuring inter-model similarity, identifying reliable teachers, and aligning heterogeneous feature distributions. This work builds on advancements in FL, PFL, and knowledge distillation to propose a collaborative transfer learning framework that leverages representative datasets and adaptive distillation weights for scalable, privacy-aware distributed learning.

3. Algorithm

Collaborative transfer learning (CTL) is a distributed learning paradigm designed for scenarios where nodes with heterogeneous feature distributions and individualized objectives collaborate to enhance model performance. CTL is designed to address scenarios where local training distributions may not fully align with the broader prediction task distributions, necessitating collaboration among nodes.

3.1. Formulation of Objective

In this paper, we focus on training local models to perform classification tasks in a distributed CTL setting. CTL is designed to address scenarios where local training distributions may not fully align with the broader prediction task distributions, necessitating collaboration among nodes.

3.1.1. Prediction Task Distributions and Local Training Distributions

Consider K nodes $\mathcal{N}_1, \mathcal{N}_2, \ldots \mathcal{N}_K$ with corresponding local datasets $\{\mathcal{D}_1, \mathcal{D}_2, \ldots \mathcal{D}_K\}$, where $\mathcal{D}_k = \{(y_i, \mathbf{x}_i) : \mathbf{x}_i \in \mathcal{X}_k, y_i \in \{0, 1\}, i \in I_k\}$. Here,

- $\mathcal{X}_k \subseteq \mathbb{R}^d$ represents the feature space of the training data for node k;
- I_k is the index set of a local dataset on node k;

- $n_k = |I_k|$ represents the number of data points at node k;
- $n = \sum_{k=1}^{K} n_k$ denotes the total number of observations across all nodes.

The local training distribution is defined by the joint distribution $P_k(\mathbf{x}, y)$ over $\mathcal{D}_k$. In many real-world scenarios, the prediction task distribution, denoted as $Q_k(\mathbf{z}, y)$ with $\mathbf{z} \in \mathcal{Z}_k$, is distinct from $P_k(\mathbf{x}, y)$, where $\mathcal{Z}_k$ represents the feature space for the prediction task, which may partially or fully overlap with $\mathcal{X}_k$ or may represent a completely different domain.

For example, a node trained on general-purpose data ($\mathcal{X}_k$) might need to adapt its predictions to specialized domains, such as medical datasets ($\mathcal{Z}_k$). Similarly, a node focused on autonomous driving in standard conditions ($\mathcal{X}_k$) could be required to predict performance in extreme weather scenarios ($\mathcal{Z}_k$).

The divergence between $P_k(\mathbf{x}, y)$ and $Q_k(\mathbf{z}, y)$ poses significant challenges, as local models are inherently limited by their training data. CTL addresses this misalignment by enabling nodes to collaborate through shared predictions, improving performance on diverse tasks while respecting privacy constraints.

3.1.2. Core Idea of CTL

CTL fosters collaboration among nodes by leveraging shared representative datasets $\tilde{\mathcal{D}} = \bigcup_{k=1}^{K} \tilde{\mathcal{D}}_k$ and adaptive distillation weights. Each node constructs a representative dataset $\tilde{\mathcal{D}}_k = (\tilde{\mathbf{x}}_h, \tilde{y}_h)_{h=1}^{r_k}$ consisting of pseudo-data points generated from its local data $\mathcal{D}_k$. These representatives act as surrogates for the node's data distribution and form a collaborative medium for indirect knowledge transfer. By exchanging predictions $p_k(\tilde{\mathbf{x}}) = f_k(\tilde{\mathbf{x}}; \theta_k)$ on $\tilde{\mathcal{D}}$ rather than raw data or model parameters, nodes can enhance their models while maintaining strict privacy guarantees.

Collaboration is selective. We assume a partially connected network where the number of connected nodes is limited, or connections between nodes are expensive. Consequently, each node seeks to identify and prioritize the optimal guest nodes, those whose models are most relevant to enhancing its own performance, while adhering to privacy constraints.

3.1.3. Collaborative Objective

The objective for each node is to optimize its local model $f_k(\mathbf{x}; \theta_k)$ while incorporating knowledge from other nodes to improve generalization. The objective function for node k at iteration t is defined as

$$\mathcal{L}_k^{(t)} = (1 - \alpha) \cdot \mathcal{L}(\mathcal{D}_k) + \alpha \cdot \sum_{j \neq k} \lambda_{j,k}^{(t)} \mathcal{L}\left(p_j^{(t-1)}, p_k^{(t-1)}\right) \tag{1}$$

where

- $\mathcal{L}(\mathcal{D}_k)$ represents the loss on the local dataset $\mathcal{D}_k$.
- $\mathcal{L}\left(p_j^{(t-1)}, p_k^{(t-1)}\right)$ is the distillation loss, which measures the discrepancy between predictions from guest node j and the local model k on the shared representative dataset. This is typically calculated using the Kullback–Leibler (KL) divergence.
- $\lambda_{j,k}^{(t)}$ represents the adaptive distillation weight, which dynamically scales the contribution of each guest node j to the learning process at node k.
- The tuning parameter α balances the importance of local optimization and collaborative knowledge transfer.

This formulation enables nodes to iteratively refine their models by balancing local optimization with collaborative insights from knowledge distillation. By leveraging insights from others, nodes progressively align their models with diverse prediction tasks, addressing heterogeneity between training and prediction distributions. Figure 1 illustrates

this collaborative learning process, where nodes iteratively refine their models by balancing local optimization with insights from guest nodes through knowledge distillation.

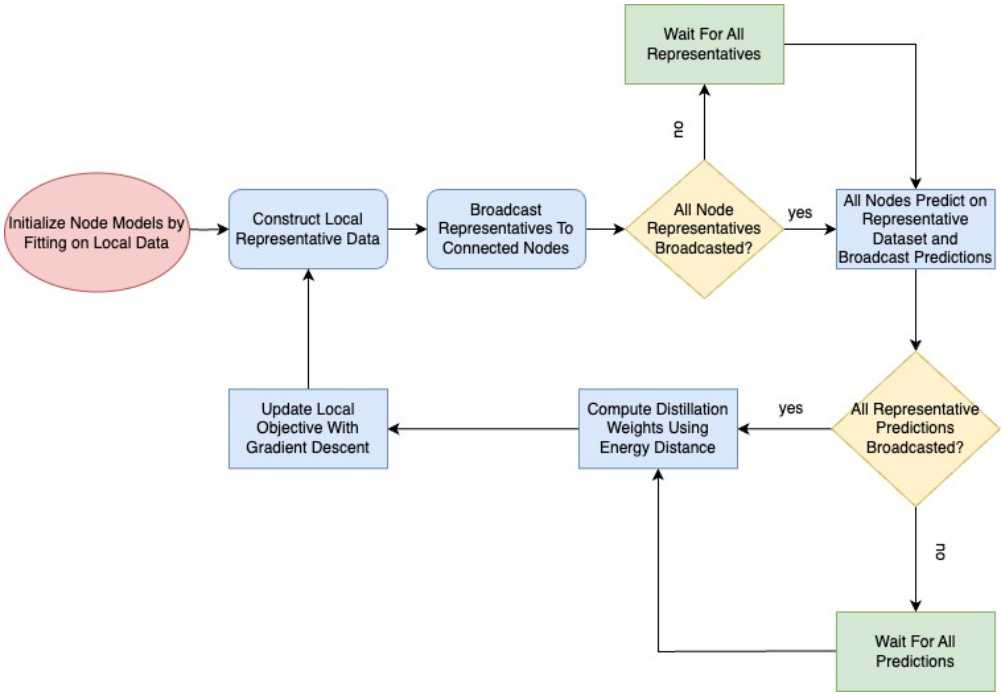


Figure 1. Flowchart of the collaborative transfer learning framework.

3.2. Local Learning

Each node $\mathcal{N}_k$ trains independently on its private dataset $\mathcal{D}_k$ by performing iterations of gradient descent to minimize a local loss function. For classification tasks, the cross-entropy loss is a commonly used choice, particularly for binary classification. It is defined as

$$\mathcal{L}(\mathcal{D}_k) = -\frac{1}{n_k} \sum_{i \in I_k} \log p(y_i|\mathbf{x}_i; \theta_k), \tag{2}$$

where $p(y|\mathbf{x}; \theta_k)$ is the local prediction function parameterized by θ_k. For binary classification, $p(y|\mathbf{x}; \theta_k)$ is typically modeled using a sigmoid activation

$$p(y|\mathbf{x}; \theta_k) = \frac{1}{1 + e^{-\theta_k^T x}}. \tag{3}$$

While this formulation works well for binary classification, the CTL framework is flexible and supports more sophisticated models, such as deep neural networks, that can learn complex patterns from data. These advanced models are particularly useful in scenarios involving multi-class tasks or high-dimensional feature spaces.

In this setup, each node optimizes its local model θ_k solely based on its private data $\mathcal{D}_k$, ensuring that sensitive data are not exposed to other nodes during the learning process.

3.3. Knowledge Distillation via Representatives

In CTL, collaboration among nodes is facilitated through the exchange of information that encapsulates key data characteristics or model predictions. Due to privacy constraints, raw data cannot be shared; instead, each node constructs a set of pseudo-data points called representatives. These representatives approximate the structure and distribution of the node's local data and collectively form a shared public dataset. This approach ensures that collaboration can occur without compromising privacy or requiring direct access to raw data or model parameters.

The representative dataset for node k, denoted as $\tilde{\mathcal{D}}_k = \{(n_h, \tilde{\mathbf{x}}_h, \tilde{y}_h)\}_{h=1}^{r_k}$ is derived from the local data $\mathcal{D}_k$ and, if applicable, the local model $f(\cdot;\theta_k)$. Here, r_k represents the number of representative points constructed by node k, and n_h is the weight associated with the hth representative. For instance, representatives can be created per mini-batch, by sub-partitioning the local dataset, or by constructing one representative per response class. The collective representative dataset across all nodes is denoted by $\tilde{\mathcal{D}} = \{\tilde{\mathcal{D}}_k\}_{k=1}^{K}$, containing a total of $\kappa = \sum_{k=1}^{K} r_k$ representatives.

In this paper, we simplify the construction of representatives by using class-wise centroids, where each node contributes two representative points:

1. The centroid for class 0, with

$$\tilde{\mathbf{x}}_k^{(0)} = \frac{1}{n_{k,0}} \sum_{i \in I_k \cap y_i = 0} \mathbf{x}_i, \quad \tilde{y}_k^{(0)} = 0, \tag{4}$$

   with $n_{k,0}$ the number of data points with response value 0 at node k.
2. The centroid for class 1, with

$$\tilde{\mathbf{x}}_k^{(1)} = \frac{1}{n_{k,1}} \sum_{i \in I_k \cap y_i = 1} \mathbf{x}_i, \quad \tilde{y}_k^{(1)} = 1, \tag{5}$$

   with $n_{k,1}$ the number of data points with response value 1 at node k.

This approach ensures that representatives are independent of the model parameters θ_k and therefore remain fixed throughout training.

To maintain privacy and avoid sharing model parameters, nodes exchange information solely via predictions. Predictions are computed on the shared representative dataset and communicated between nodes to assist in model updates. The predictions for node k are given by

$$p_k = \left\{ p_k^{(h)} = f(\tilde{\mathbf{x}}_h; \theta_k) \right\}_{h=1}^{\kappa}.$$

where $p_k^{(h)}$ represents the predicted value of the local model $f(\cdot;\theta_k)$ for the representative point $\tilde{\mathbf{x}}_h$.

While training on a given node k, we refer to this node as the home node, and all other nodes as guest nodes. To distill knowledge from guest nodes to the home node, we utilize the KL divergence between the corresponding posterior probability distributions. The distillation loss between the home node k and guest node j on the representative dataset is defined as

$$\mathcal{L}(p_j, p_k) = \frac{1}{n} \sum_{h=1}^{\kappa} n_h D_{KL}(p_j^{(h)} | p_k^{(h)}), \tag{6}$$

where $D_{KL}(p_j^{(h)} \| p_k^{(h)})$ represents the KL divergence between the predictions of guest node j and home node k for representative point h, and n_h is the weight associated with representative h.

This formulation enables nodes to transfer knowledge effectively while adhering to privacy constraints by sharing only aggregated information through representatives and predictions.

3.4. Distillation Weights

The distillation weight $\lambda_{j,k}$ dynamically determines the contribution of guest node j in refining the local model at node k. It reflects the similarity between feature spaces and

their alignment with the prediction task distribution, ensuring that collaboration prioritizes relevant and effective knowledge transfer.

This similarity can be quantified using the Energy Coefficient H between random variables X and Y [27,28],

$$H = \frac{2\mathbb{E}\|X - Y\| - \mathbb{E}\|X - X'\| - \mathbb{E}\|Y - Y'\|}{2\mathbb{E}\|X - Y\|},$$

where X and Y' are independent copies of X and Y, respectively. H evaluates the heterogeneity of feature distributions, and an efficient approximation of pairwise Euclidean distances was proposed to reduce computational complexity from $O(n^2d)$ to $O(nd)$ [29]. Instead of directly computing all pairwise distances,

$$\mathbb{E}\|X - Y\| = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\|X_i - Y_j\|,$$

which scales quadratically in n, the approximation estimates the expected Euclidean distances using the first four moments, avoiding the quadratic cost of direct pairwise calculations. The reduced complexity makes it well suited for large-scale distributed learning.

In distributed learning, prediction task distributions are often represented by their moments rather than observed samples. Thus, the approximated Energy Coefficient becomes particularly advantageous, enabling nodes to effectively quantify feature space similarity even in the absence of raw data or empirical expectations.

By comparing predictions on representative datasets, nodes identify reliable collaborators (i.e., "qualified teachers") whose models align well with their prediction tasks. Adaptive distillation weights $\lambda_{j,k}^{(t)}$ guide this collaboration, ensuring that knowledge transfer is both relevant and effective. The iterative framework balances local optimization and knowledge distillation, progressively aligning models across diverse prediction tasks.

3.4.1. Energy Coefficients with Feature Importance

The Energy Coefficient H between two random vectors $X = (X_1, X_2, \ldots, X_d)$ and $Y = (Y_1, Y_2, \ldots, Y_d)$ reflects their similarity, leveraging key moments of their feature distributions. For each feature i, the approximation of the expected Euclidean distance is derived using moments such as means (μ_{X_i}, μ_{Y_i}), variances ($\sigma^2_{X_i}$, $\sigma^2_{Y_i}$), skewness (γ_{3X_i}, γ_{3Y_i}), and kurtosis (γ_{4X_i}, γ_{4Y_i}) as follows

$$\mathbb{E}[\|X_i - Y_i\|] \approx \sqrt{\nu_{X_iY_i}}\left(1 - \frac{C_{4X_iY_i} + 4C_{3X_iY_i}\delta_{\mu_i} + 2\nu^2_{X_iY_i} - 2\delta^4_{\mu_i}}{8\nu^2_{X_iY_i}}\right),$$

where $\nu_{X_iY_i} = \sigma^2_{X_i} + \sigma^2_{Y_i} + \delta^2_{\mu_i}$, $C_{4X_iY_i} = \sigma^4_{X_i}\gamma_{4X_i} + \sigma^4_{Y_i}\gamma_{4Y_i}$, and $C_{3X_iY_i} = \sigma^3_{X_i}\gamma_{3X_i} - \sigma^3_{Y_i}\gamma_{3Y_i}$, $\delta_{\mu_i} = \mu_{X_i} - \mu_{Y_i}$.

The individual Energy Coefficient H_i for feature i is then computed based on the expected Euclidean distances between X_i and Y_i, their independent copies, and the overall distribution characteristics.

To account for the varying contributions of features to model predictions, we incorporate feature importance w_i. This weight is derived from the host model using techniques such as gradient magnitudes, SHAP values, or Gini importance. Specifically, we define w_i as

$$w_i = \frac{|q_i - q_0|}{\sum_{j=1}^{d}|q_j - q_0|}$$

where $q_0, q_1, \ldots, q_d$ are current predictions for pseudo-observations $z_0, z_1, \ldots, z_d \in \mathbb{R}^d$ constructed by setting the ith component to 1 and others to 0. This weight quantifies the relative contribution of feature i to the model's predictions.

The final Energy Coefficient H between X and Y is then computed as a weighted sum of the feature-specific Energy Coefficients

$$H = \sum_{i=1}^{d} w_i H_i.$$

This formulation allows the Energy Coefficient to reflect both the statistical similarity between individual features and their relative importance, improving its relevance for distributed learning tasks.

3.4.2. Class-Wise Energy Coefficients

For binary classification, the Energy Coefficient is refined by considering class-wise distributions. Specifically, $H^{(0)}$ and $H^{(1)}$, representing the Energy Coefficients for classes 0 and 1, are computed independently. The final Energy Coefficient $\hat{H}$ between a node and a prediction task distribution is derived as the weighted average of $H^{(0)}$ and $H^{(1)}$, where the weights are proportional to the class distribution at the node. This approach provides finer granularity and enhances the relevance of collaboration by incorporating class-specific information.

3.4.3. Distillation Weight Formula

To enable a finer distinction between nodes' capacities to handle specific prediction tasks, we consider sub-distributions $\mathcal{Z}_1, \ldots, \mathcal{Z}_{\tau_k}$ within a node's prediction task distribution. This framework assumes that splitting the prediction task distribution sharpens the Energy Coefficient H values, facilitating more decisive allocation of distillation weights. The Energy Coefficient between node j and sub-distribution i is denoted by $\hat{H}_j^{(i)}$.

For node k, the distillation weights for each guest node j are computed as

$$\lambda_{j,k} = \sum_{i=1}^{\tau_k} \hat{H}_k^{(i)} (1 - \hat{H}_j^{(i)}) \tag{7}$$

Here, $\hat{H}_k^{(i)}$ closer to 1 indicates that node k is poorly aligned with sub-distribution $\mathcal{Z}_i$ and requires assistance. Conversely, $\hat{H}_j^{(i)}$ closer to 0 indicates that guest node j is well aligned with $\mathcal{Z}_i$ and thus a suitable collaborator.

This formula allows the dynamic adjustment of weights, ensuring that collaboration is focused on the most relevant and effective nodes for improving prediction tasks.

3.5. Complete Algorithm for Collaborative Transfer Learning

Having detailed the components of our framework, we now present the complete algorithm for CTL in Algorithm 1. This algorithm integrates the local objective, knowledge transfer via representative datasets, and adaptive distillation weights to enable effective collaboration among nodes with heterogeneous data distributions.

Algorithm 1 Collaborative Transfer Learning (CTL)

Initialization:
for each node $k = 1, \ldots, K$ **do**
  Fit local model $f(\cdot; \theta_k^{(0)})$ on $\mathcal{D}_k$
  Construct representative dataset $\tilde{\mathcal{D}}_k^{(0)}$
  Broadcast $\tilde{\mathcal{D}}_k^{(0)}$ to connected nodes
end for
[Synchronize]: Wait until all $\tilde{\mathcal{D}}_j^{(0)}$ are received
for each node $k = 1, \ldots, K$ **do**
  Compute predictions $p_k^{(0)} = f(\tilde{\mathbf{x}}_h^{(0)}; \theta_k^{(0)})$
  Broadcast $p_k^{(0)}$ to connected nodes
end for
[Synchronize]: Wait until all $p_j^{(0)}$ are received
for $t = 1, \ldots, T$ **do**
  for each node $k = 1, \ldots, K$ **do**
    Update distillation weights $\lambda_{j,k}^{(t)}$ using Energy Coefficient H by
    Update local objective function $\mathcal{L}_k^{(t)}$
    Perform gradient descent to update $\theta_k^{(t)}$
    Construct updated representative dataset $\tilde{\mathcal{D}}_k^{(t)}$
    Broadcast $\tilde{\mathcal{D}}_k^{(t)}$ to connected nodes
  end for
  [Synchronize]: Wait until all $\tilde{\mathcal{D}}_j^{(t)}$ are received
  for each node $k = 1, \ldots, K$ **do**
    for each updated $\tilde{\mathcal{D}}_j^{(t)}$ **do**
      Compute updated predictions $p_k^{(t)} = f(\tilde{\mathbf{x}}_h^{(t)}; \theta_k^{(t)})$
    end for
    Broadcast $p_k^{(t)}$ to connected nodes
  end for
end for

4. Simulation Studies

4.1. Simulation: Performance of Collaborative Learning Strategies

We conducted an experiment utilizing the UCI ML Breast Cancer Wisconsin (Diagnostic) dataset [30], which consists of 569 observations with $d = 30$ features of tumor measurements $\mathbf{X}$. The targets $\mathbf{y}$ are zero or one, indicating the tumor being cancerous or not.

To illustrate the proposed collaborative learning algorithm, we simulated a larger dataset from the original dataset to suit the study's purpose. We generated transformed versions of the dataset by applying a shift to each feature $\mathbf{x}_j \to \mathbf{x}_j + c_j \cdot \sigma_j + \epsilon$, where $\hat{\sigma}_j$ is the standard deviation of feature j, $c_j \sim TN(\mu = 0, \sigma = 2, a = -5, b = 5)$ is sampled from a truncated normal distribution, and $\epsilon \sim N(\mu = 0, \sigma = 0.1)$ is random noise. We fit a logistic regression model $p(Y|X; \theta)$ to the original dataset to generate target labels for the simulated dataset.

By applying 300 shifts to the original dataset, we generated $569 \times 300 = 170,700$ observations, denoted as $\mathcal{D}$. The data were then randomly split into a training set (80%, $\mathcal{D}_T$) and a testing set (20%, $\mathcal{D}_\tau$).

4.1.1. Node Creation

To simulate nodes with different feature distributions, we applied k-means clustering (one iteration) on the first two principal components of $\mathcal{D}$, setting $k = 10$. Each data point

in $\mathcal{D}_T$ and $\mathcal{D}_\tau$ was assigned to one of these 10 clusters, creating 10 nodes. This clustering ensured each node had distinct feature distributions.

To increase the prediction task complexity, we assigned each node k to predict not only on its own testing data but also on the testing data from two additional random nodes. This design enlarged the prediction task distribution beyond the local training data distribution, necessitating collaborative learning. Nodes were evaluated on their testing accuracy for all three prediction tasks over multiple communication rounds.

4.1.2. Comparison of Guest Node Selection Strategies

We evaluated five strategies for assigning guest nodes to each host node:

1. *All*: All other nodes are used as guest nodes.
2. *Best*: Select the two nodes with the lowest Energy Coefficients H relative to each prediction task.
3. *Random*: Select two random guest nodes.
4. *Worst*: Select the two nodes with the highest H relative to each prediction task.
5. *Local*: No guest nodes are used (local-only training).

Each strategy was evaluated over 100 simulations, with performance measured by testing accuracy across 100 communication rounds.

4.1.3. Results

Figure 2 presents the mean testing accuracy across 100 simulations for various guest-node selection strategies.

Panel (a) shows the overall mean testing accuracy across all prediction tasks. The *local* strategy consistently achieved the lowest accuracy due to its inability to generalize to non-local prediction tasks. In contrast, all other strategies significantly outperformed the *worst* strategy, which failed to align guest nodes with the prediction tasks. Among these, the *best* strategy demonstrated the fastest improvement during the initial iterations and maintained the highest accuracy overall.

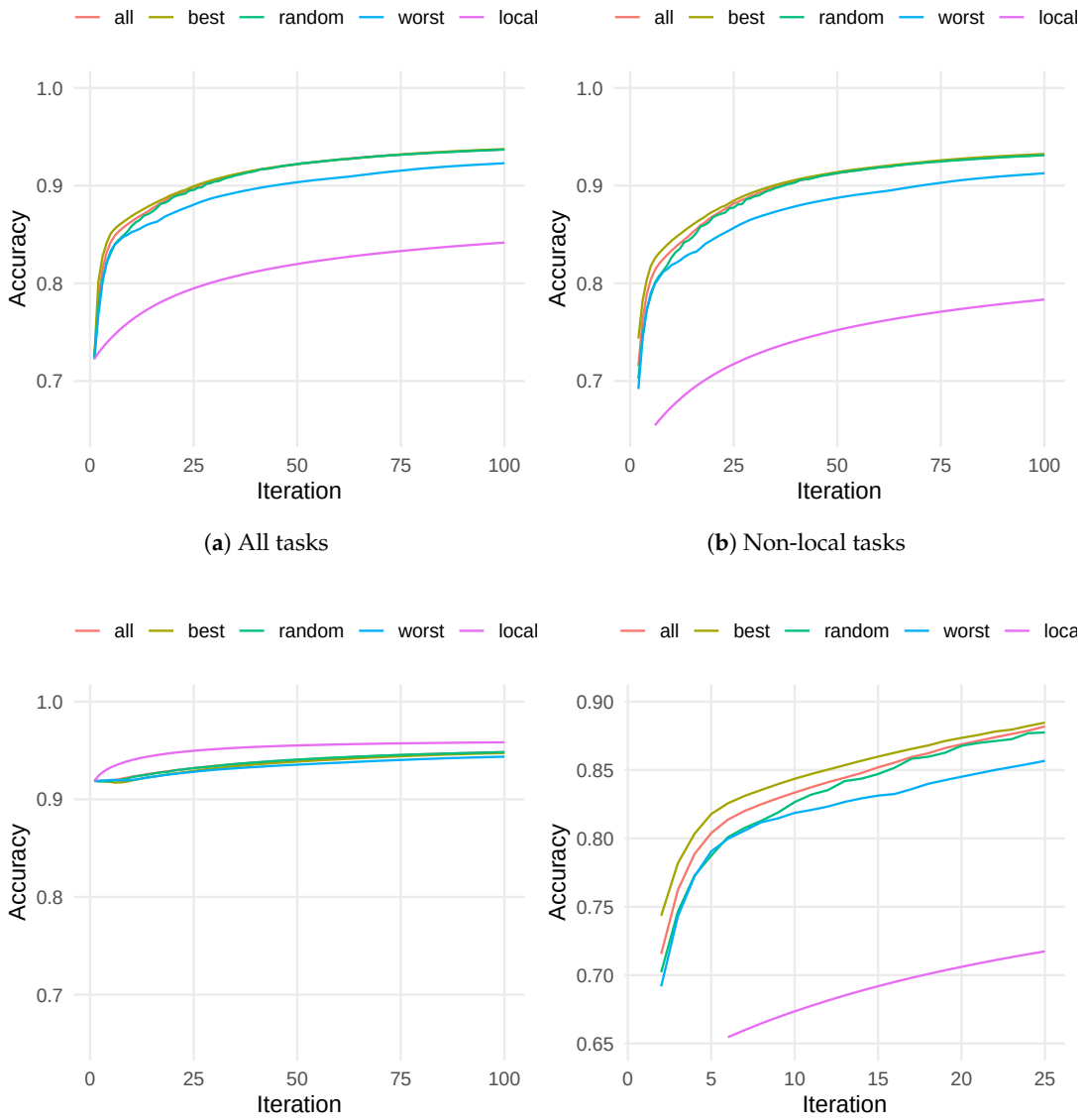
Panel (b) focuses on prediction tasks where the distribution differs from the local training data. The *local* strategy struggled the most, as models trained solely on local data lack the capacity to adapt to non-local distributions. The *best* strategy significantly outperformed the others by leveraging well-aligned guest nodes to provide complementary information, resulting in consistent improvement.

Panel (c) evaluates the mean accuracy for prediction tasks aligned with the local training data. As expected, the *local* strategy outperformed all others by overfitting to its local distribution, maximizing performance on its own tasks. However, this came at the expense of poor generalization to non-local tasks. Collaborative strategies such as *best* and *all* achieved a balance between local and non-local tasks, showing slightly lower accuracy on local tasks but significantly improved generalization overall.

Panel (d) provides a zoomed-in view of the early-stage accuracy on non-local tasks (first 25 iterations), highlighting the adaptation speed of different strategies. The *best* strategy was the first to surpass an accuracy of 0.85, reaching 0.8503 at iteration 12, demonstrating its rapid adaptation to non-local tasks. Other strategies took longer: *worst* reached 0.8521 at iteration 23, *all* at iteration 15 (0.8519), and *random* at iteration 16 (0.8517). The *local* strategy, on the other hand, failed to reach 0.85 even after 100 iterations, with a final accuracy of 0.7834. This comparison underscores the importance of selecting well-aligned guest nodes for efficient knowledge transfer and rapid improvement.

These findings highlight the trade-offs in collaborative learning. Strategies such as *best* demonstrate the value of selecting well-aligned guest nodes to achieve faster and more consistent improvements in accuracy across diverse prediction tasks.

Figure A2 in Appendix A presents the individual accuracy per node per simulation. The *best* strategy demonstrated steady and consistent improvement over iterations, whereas other strategies, including *all*, exhibited oscillations in accuracy for certain nodes. This highlights the importance of selecting well-aligned guest nodes for robust and stable learning. Similar trends were observed across all tasks (Figure A1), while for local tasks (Figure A3), collaborative strategies showed no clear advantage.



(**a**) All tasks

(**b**) Non-local tasks

(**c**) Local tasks

(**d**) Early-stage on non-local tasks

Figure 2. Performance of guest-node selection strategies over 100 simulations. (**a**) Mean accuracy across all prediction tasks. (**b**) Mean accuracy for tasks from distributions different from local training data. (**c**) Mean accuracy for tasks aligned with the local training data distribution. (**d**) A zoomed-in view of early-stage accuracy on non-local tasks (first 25 iterations), highlighting the initial adaptation dynamics of different strategies.

Figure 3 illustrates a specific example where the *best* strategy significantly outperformed the *local* strategy. This example underscores the necessity of leveraging guest nodes for collaborative learning, demonstrating how effective collaboration can enable local nodes to overcome limitations imposed by their training data.

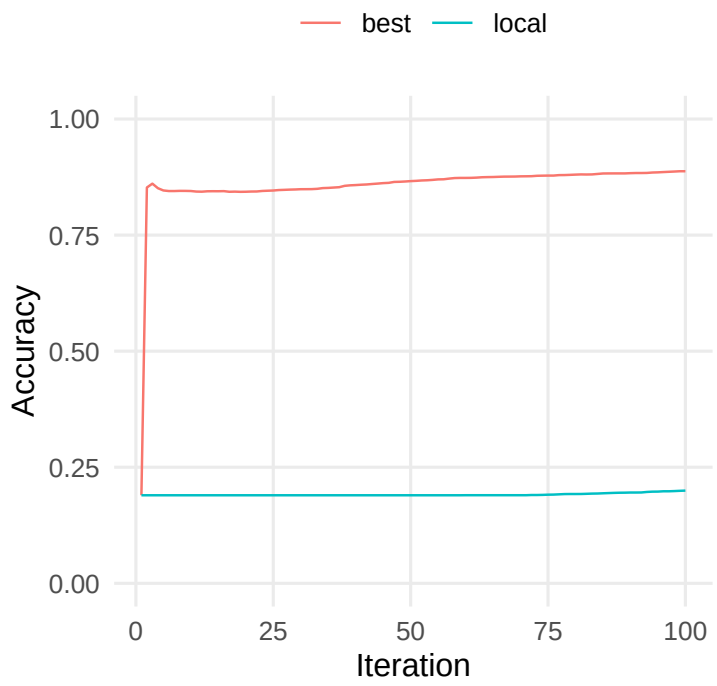


Figure 3. Testing accuracy for Node 4 in Simulation 1 under the *best* and *local* strategies. The *best* strategy significantly outperformed *local* by leveraging well-aligned guest nodes, highlighting the importance of collaboration in improving performance on non-local prediction tasks.

4.2. Simulation: Effect of Spread and Dispersion on Collaboration

In this simulation, we explored how spread and dispersion in data distributions affected the improvement of the *best* strategy compared to *local* in our CTL framework. Here, spread refers to the inter-node distance, measuring how far apart the data distributions of different nodes are, while dispersion refers to the intra-node variance, indicating how concentrated or diverse the data within each node are.

To examine these effects, we simulated data under controlled conditions, varying the spread and dispersion systematically.

4.2.1. Data Generation

We first generated model parameters used to define the decision boundary for label generation. Each element of $\theta \in \mathbb{R}^{30}$ (with $d = 30$ features) was sampled as $\theta_j \sim Unif(1, 2)$ and set to negative with 50% probability. The norm of θ was normalized to three for stability.

For each of the $K = 10$ nodes, we sampled a mean vector with each component $\mu_{k,j} \sim Unif(-1, 1) \times spread$, where *spread* controls the inter-node distance. Each node's data were then drawn from a multivariate normal distribution $\mathbf{X}_k^{(i)} \sim \mathcal{N}(\mu_k, \Sigma)$, where $\Sigma = dispersion^2 \times S$, and S is a fixed covariance matrix with 1 on the diagonal and 0.5 elsewhere. The parameter *dispersion* controlled the intra-node variability, and all nodes shared the same Σ.

To create labels $Y_k^{(i)}$, we perturbed the original θ for each node. For node k, θ was adjusted as

$$\theta_j \to \theta_j + (|\theta_j| \times p_j) \times a, \quad p_j \sim \mathcal{N}(0, 1), \quad a = \frac{dispersion}{10},$$

where a ensures that the perturbation depends on the level of dispersion. The perturbed θ_k was then used to compute the probability of label 1, $Y_k^{(i)} = 1$ with probability $p_{\theta_k}(\mathbf{X}_k^{(i)})$.

This process ensured that label generation reflected both local variability and inter-node heterogeneity.

Each node's dataset $\mathcal{D}_k = (\mathbf{X}_k^{(i)}, Y_k^{(i)})_{i=1}^{n_k}$ contained $n_k = 500$ samples, with 20% held out for validation.

4.2.2. Results

Figure 4 illustrates the impact of spread and dispersion on the performance of *best* and *local* strategies.

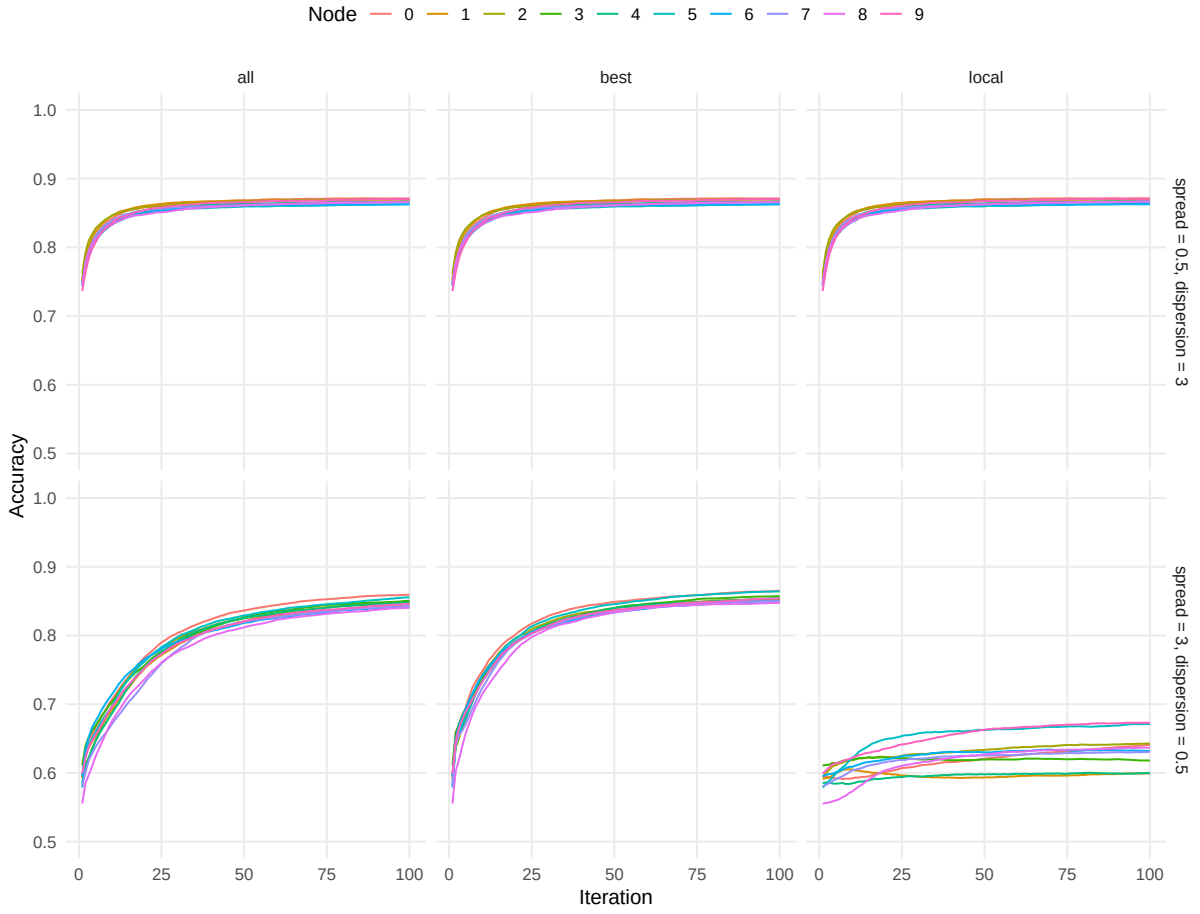


Figure 4. Performance comparison under varying spread and dispersion for CTL strategies. The top three panels (left to right: *all*, *best*, *local*) depict results for high dispersion (large intra-node variance) and low spread (small inter-node distances), where local models are relatively self-sufficient. The bottom three panels (left to right: *all*, *best*, *local*) correspond to low dispersion (small intra-node variance) and high spread (large inter-node distances), highlighting the advantages of collaboration when local data are insufficient. The *best* strategy consistently outperforms *local* in the high-spread, low-dispersion scenario by leveraging complementary information from distant nodes.

Top panels correspond to high dispersion (large intra-node variance) and low spread (small inter-node distances). Here, nodes had sufficiently diverse features locally, making the *local* strategy effective. Consequently, *best* showed limited improvement over *local*.

Bottom panels correspond to low dispersion (small intra-node variance) and high spread (large inter-node distances). In this case, *local* performed poorly since nodes lacked sufficient diversity within their own data. However, the *best* strategy showed significant improvement by leveraging information from distant nodes with complementary features.

These results highlight that when nodes are closely clustered (low spread) but have diverse local data (high dispersion), the *local* strategy performs comparably to *best*. Con-

versely, when nodes are far apart (high spread) but have limited local variability (low dispersion), collaboration becomes crucial, and *best* significantly outperforms *local*.

This simulation demonstrates the importance of balancing spread and dispersion to optimize collaborative learning. Effective collaboration is particularly beneficial when local data distributions are insufficient to represent the broader prediction task.

5. Discussion and Future Work

5.1. Convergence

Our proposed algorithm involved optimizing a combination of local classification loss $\mathcal{L}_{CE}$ and knowledge distillation loss $\mathcal{L}_{KL}$ at each node. In standard gradient descent with a strongly convex and L-smooth objective function $\mathcal{L}(\theta)$, the convergence rate follows

$$\|\theta_{t+1} - \theta^*\| \leq (1 - \frac{\mu}{L})\|\theta_t - \theta^*\|. \tag{8}$$

Since our loss function is a mixture of local and distillation terms, we examined whether similar convergence guarantees could be established.

At each iteration, the optimal parameters θ^* depend on predictions from other nodes, making the optimization problem non-stationary. If θ^t is the optimal parameter at iteration t, then we assume the drift between optimal solutions across iterations satisfies

$$\left\|\theta^*_{t+1} - \theta^*_t\right\| \leq B, \tag{9}$$

where B is a bound on how much the optimal parameter changes due to updates at other nodes. Intuitively, B depends on the number of local epochs and the step size, as frequent or aggressive updates at one node can amplify shifts in θ^*_t. Importantly, B is also influenced by the distillation weight α, since stronger reliance on guest models induces greater shifts in the local optimum.

If B is sufficiently small relative to the update magnitude of gradient descent, we can expect the algorithm to continue converging to a moving target. However, since gradient descent steps decrease in magnitude over time, eventually, B may exceed the step size, preventing further progress. To mitigate this, we require that α gradually decreases as training progresses, i.e., $\alpha(T) \to 0$ as $T \to \infty$. If B is a continuous function of α, then $B(T) \to 0$, ensuring stable convergence.

The magnitude of B is also affected by how guest nodes are selected. When a node primarily interacts with nearby nodes whose models evolve similarly, B remains small, leading to smoother convergence. Conversely, if a node relies on guest nodes with highly divergent distributions, the shifting optimal parameters may destabilize training. This highlights the importance of selecting well-aligned guest nodes, as reflected in our distillation weight formulation.

A remaining open question is whether nodes should only distill from the single closest node (in terms of prediction task distribution) or from multiple nodes. While distilling from a diverse set of nodes could improve robustness, it may also introduce excessive variation, increasing B. Future work should explore whether distilling from a single well-aligned node is preferable to distilling from a broader set with adaptive weighting.

Consider two nodes k and j with training data distributions $P_k(X, Y)$ and $P_j(X, Y)$. The objective function for node k under our algorithm is:

$$\mathcal{L}_k(\theta_k) = \mathcal{L}_{CE} + \mathcal{L}_{KL}(p_{\theta_j}||p_{\theta_k}),$$

which expands to:

$$\mathcal{L}_k(\theta_k) = -\mathbb{E}_{P_k}[\log(p_{\theta_k}(Y|X))] + \mathbb{E}_{P_j}[\log(p\theta_j(Y|X)) - \log(p_{\theta_k}(Y|X))].$$

By comparison, training only on local data would yield:

$$\mathcal{L}_k(\theta_k) = -\mathbb{E}_{P_k}[\log(p_{\theta_k}(Y|X))].$$

To justify the benefits of distillation, we must analyze how minimizing $\mathcal{L}_{KL}$ affects generalization to unseen distributions, particularly when $P_j(X, Y)$ is closer to the target distribution than $P_k(X, Y)$. Theoretical bounds on this improvement remain an open area of research.

5.2. Computational Efficiency and Scalability

The CTL framework is designed to optimize computational efficiency while maintaining scalability in distributed learning environments. Unlike traditional federated learning methods that require frequent global aggregation, CTL enables indirect knowledge transfer via knowledge distillation, significantly reducing communication overhead. By leveraging representative datasets rather than exchanging raw data or model parameters, the framework minimizes the number of updates required per iteration, making it particularly well suited for large-scale networks.

A key efficiency improvement in CTL comes from the adaptive distillation weights, which dynamically select the most relevant guest nodes based on Energy Coefficients. This selective knowledge transfer prevents unnecessary computations, reducing redundant distillation steps compared to ensemble-based methods such as FedMD [31] and FedBE [32]. Instead of averaging knowledge from all available sources, CTL prioritizes information from the most informative nodes, leading to faster convergence with lower computational cost.

Furthermore, the computation of Energy Coefficients is optimized using a moment-based approximation of pairwise Euclidean distances [29], reducing complexity from $O(n^2d)$ to $O(nd)$. This allows efficient similarity estimation between node distributions without direct pairwise calculations, making the framework scalable to large numbers of nodes and high-dimensional feature spaces.

As the number of nodes grows, a potential challenge is the increasing cost of evaluating similarity metrics across all node pairs. To further improve scalability, future work can explore hierarchical clustering techniques to group similar nodes and reduce the number of comparisons required. Additionally, decentralized implementations of CTL, where nodes only exchange information with a subset of neighbors rather than a global network, could provide further improvements in efficiency.

By balancing computational cost with effective knowledge transfer, CTL provides a scalable alternative to traditional federated learning approaches, making it well suited for large, heterogeneous distributed systems.

5.3. Future Work

In our algorithm, the representative dataset was constructed using class-wise centroids, which remained fixed throughout the training process. While this approach is computationally efficient, it is not adaptive to the evolving state of model training and cannot fully capture updates in the model parameters. One possible improvement is to adopt Score-Matching Representatives (SMRs) [24], which construct representatives by matching the score function values (i.e., gradients) of the local objective function. This method provides a more adaptive representation of the model's behavior during training.

However, its applicability is limited to smooth objective functions, making it unsuitable for models with non-differentiable components or highly complex architectures.

Another limitation lies in our underutilization of the representative response $\tilde{y}_k$. In its current implementation, $\tilde{y}_k$ provides static class information but does not dynamically contribute to the model alignment process. However, $\tilde{y}_k$ holds critical information about the underlying model from node $\mathcal{N}_k$. When genuine model differences exist between nodes, the degree of mismatch between a local model's predictions on representative features $\tilde{\mathbf{x}}_k$ and their corresponding $\tilde{y}_k$ could serve as a valuable indicator of alignment. This mismatch could be leveraged in two ways: as a weighting factor to merge the KL-divergence terms in the distillation process or as a component of the distillation weights $\lambda_{j,k}$. Incorporating this information could lead to more adaptive and accurate knowledge transfer across nodes.

While this work demonstrates the efficacy of our framework for classification tasks using logistic regression and centroid-based representatives, its applicability to more complex model architectures (e.g., neural networks or transformers) warrants further exploration. Complex models may require alternative representative construction techniques or extended notions of distillation weights to handle high-dimensional feature spaces effectively. Investigating how our framework generalizes across such models remains an open area of research.

A noteworthy aspect of our proposed framework is the concept of splitting the prediction task distribution into sub-distributions $\mathcal{Z}_1, \ldots, \mathcal{Z}_{\tau_k}$ to better differentiate nodes' capacities to handle specific tasks. While our simulation studies approximate this by assigning additional prediction tasks from other nodes, we do not provide a formal method for achieving such splits. Future research could explore systematic approaches to partition prediction task distributions, potentially leveraging clustering algorithms, domain knowledge, or task-specific metrics. These methods could further enhance the precision of Energy Coefficient values and improve the allocation of distillation weights in diverse collaborative learning scenarios.

Similarly, another promising direction is to split the network of nodes into meaningful clusters to improve scalability and reduce computational overhead. In large-scale distributed settings, evaluating similarity metrics across all node pairs can become prohibitive. One potential enhancement is to employ hierarchical clustering to group nodes based on their feature distributions or Energy Coefficients, limiting similarity computations to within-cluster interactions. Additionally, decentralized implementations of CTL could be explored, where nodes only exchange information with a subset of neighbors instead of maintaining full connectivity. By adopting localized collaboration strategies, CTL can maintain its effectiveness while significantly reducing communication and computational costs in large-scale deployments.

A critical challenge in distributed learning is the presence of adversarial nodes that may inject misleading information or attempt to infer sensitive data from shared predictions. While traditional federated learning uses Byzantine-resilient aggregation and robust statistics to mitigate such threats, CTL relies on knowledge distillation, which requires different security measures. A promising direction for improving robustness is incorporating adaptive trust mechanisms, where nodes adjust distillation weights based on historical consistency in predictions, reducing the influence of unreliable collaborators. Additionally, anomaly detection techniques can be applied to representative datasets to identify and filter out manipulated data before knowledge distillation. Differential privacy methods may also help protect sensitive information by injecting calibrated noise into shared predictions, limiting adversarial nodes' ability to extract private data. Future research will explore these strategies to enhance CTL's security and resilience against adversarial attacks.

Another area for future work is the computational scalability of our framework. The computation of Energy Coefficients and distillation weights involves evaluating pairwise

node alignments, which may become computationally expensive in networks with a large number of nodes. Exploring efficient approximations for these coefficients and reducing the communication overhead in collaborative settings will be critical for scaling to industrial applications with thousands of nodes.

Author Contributions: Conceptualization, J.C., Q.C. and M.F.; methodology, J.C. and Q.C.; software, J.C.; validation, B.G., R.S. and Z.C.; formal analysis, J.C.; investigation, J.C.; writing—original draft preparation, J.C., Q.C. and K.L.; writing—review and editing, K.L.; visualization, K.L.; supervision, K.L.; funding acquisition, K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Faculty Development Grant Program, University of Alabama at Birmingham.

Data Availability Statement: The code and simulation outputs supporting the findings of this study will be made publicly available on GitHub at https://github.com/jcasey98/DCLRep (accessed on 1 March 2025) upon publication. Readers can access the repository for reproducibility and further experimentation.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DCL	Distributed Collaborative Learning
CTL	Collaborative transfer learning
DML	Distributed machine learning
MTL	Multi-Task Learning
FL	Federated learning
PFL	Personalized Federated Learning

Appendix A. Additional Figures

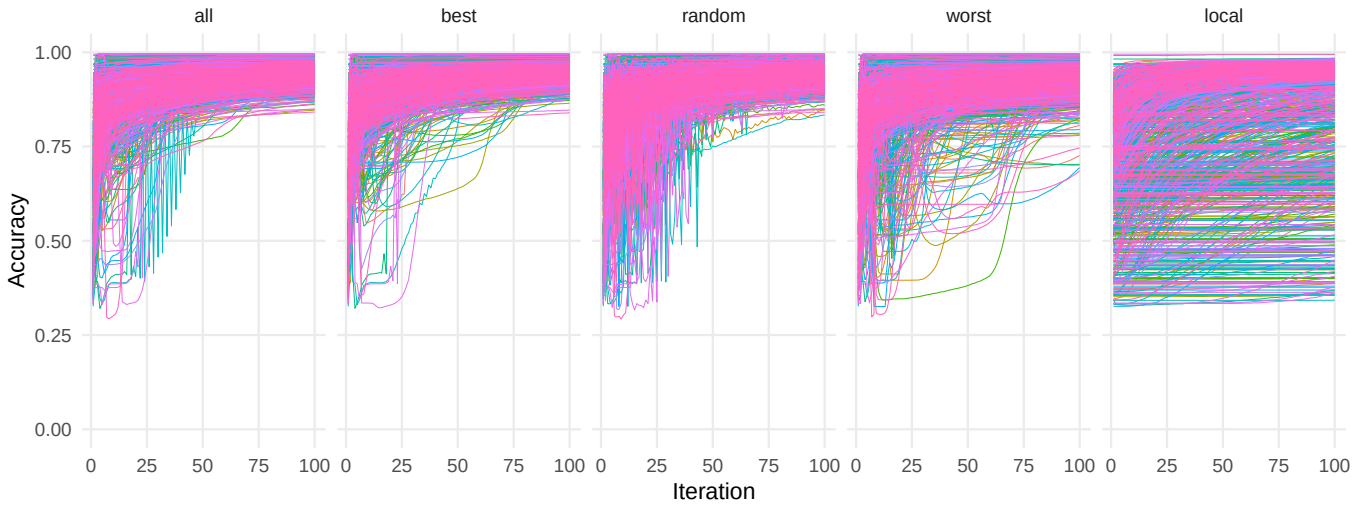


Figure A1. Accuracy curves for prediction tasks across all tasks, including both local and non-local distributions. The figure highlights the performance of various guest-node selection strategies across individual simulations. The *best* strategy consistently improves accuracy across iterations, while other strategies, particularly *worst*, exhibit slower and less stable improvements. Different line colors correspond to different host nodes.

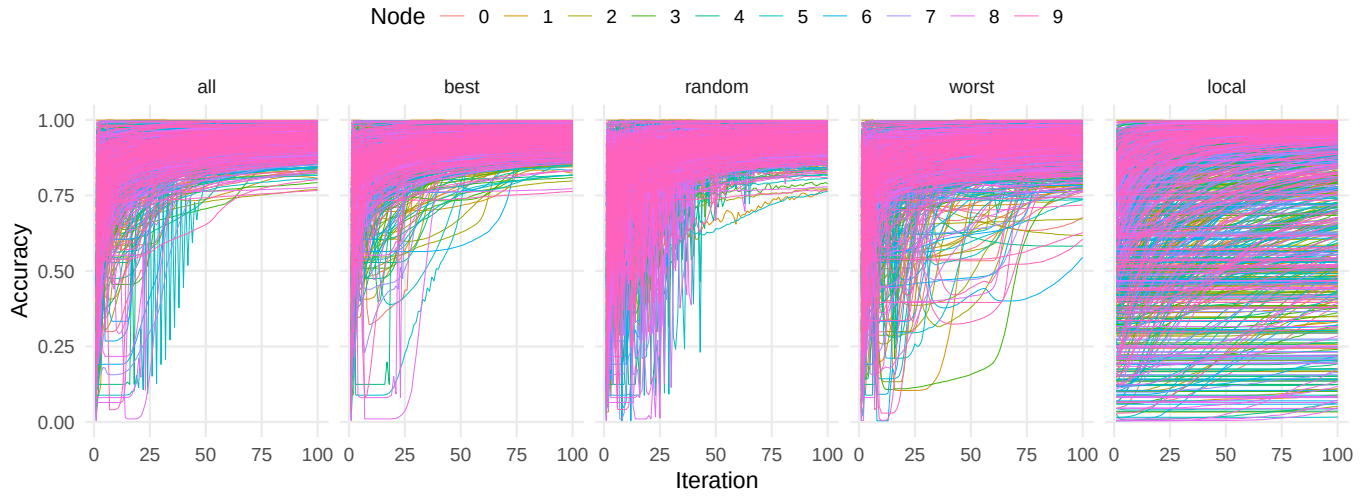


Figure A2. Accuracy curves for prediction tasks differing from the local training data distribution. The figure highlights the performance of various guest-node selection strategies across individual simulations. The *best* strategy demonstrates faster and more consistent improvements, while poorly aligned strategies fail to enhance accuracy for certain nodes.

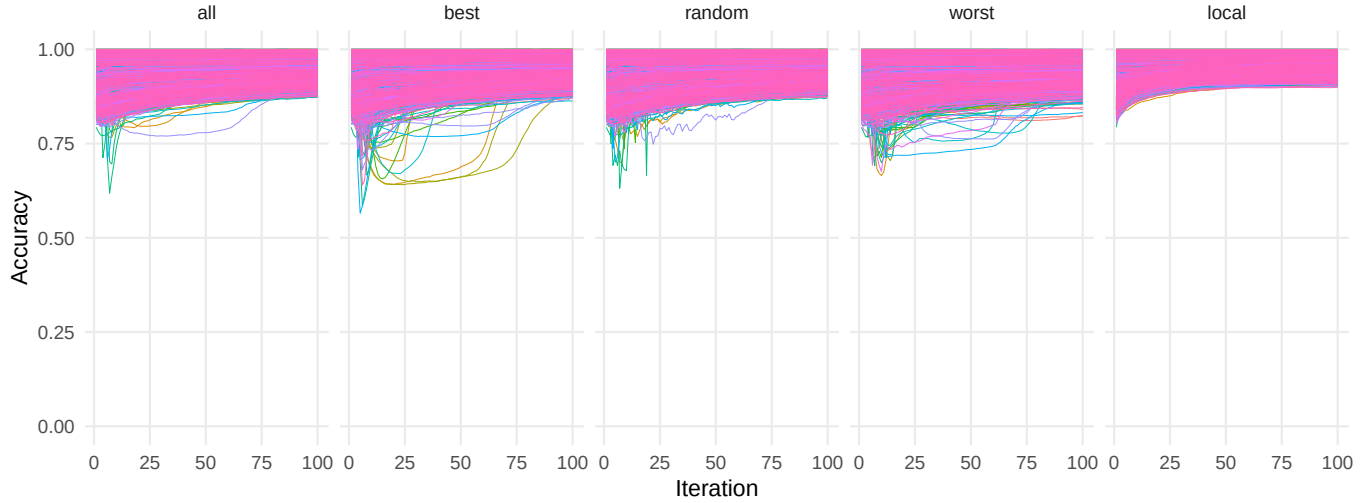


Figure A3. Accuracy curves for prediction tasks aligned with the local training data distribution. The *local* strategy maintains the highest accuracy, as it solely optimizes for its own data. In contrast, collaborative strategies (*best*, *all*, etc.) do not show noticeable improvement over iterations, highlighting the trade-off between local optimization and generalization to non-local tasks. Different line colors correspond to different host nodes.

References

1. DeepSeek-AI; Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; et al. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv* **2025**. [CrossRef]
2. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**. [CrossRef]
3. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial Intelligence and Statistics, PMLR, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
4. Reddi, S.; Charles, Z.; Zaheer, M.; Garrett, Z.; Rush, K.; Konečný, J.; Kumar, S.; McMahan, H.B. Adaptive Federated Optimization. *arXiv* **2021**. [CrossRef]
5. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated optimization in heterogeneous networks. *Proc. Mach. Learn. Syst.* **2020**, *2*, 429–450.

6. Tan, L.; Zhang, X.; Zhou, Y.; Che, X.; Hu, M.; Chen, X.; Wu, D. AdaFed: Optimizing participation-aware federated learning with adaptive aggregation weights. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 2708–2720. [CrossRef]
7. Xie, C.; Koyejo, S.; Gupta, I. Asynchronous Federated Optimization. *arXiv* **2020**. [CrossRef]
8. Chen, Y.; Huang, W.; Ye, M. Fair Federated Learning under Domain Skew with Local Consistency and Domain Diversity. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 12077–12086.
9. Acar, D.A.E.; Zhao, Y.; Navarro, R.M.; Mattina, M.; Whatmough, P.N.; Saligrama, V. Federated Learning Based on Dynamic Regularization. *arXiv* **2021**. [CrossRef]
10. Seyedmohammadi, S.J.; Atapour, S.K.; Abouei, J.; Mohammadi, A. KnFu: Effective Knowledge Fusion. *arXiv* **2024**, arXiv:2403.11892.
11. Zhang, J.; Guo, S.; Ma, X.; Wang, H.; Xu, W.; Wu, F. Parameterized knowledge transfer for personalized federated learning. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 10092–10104.
12. Tan, A.Z.; Yu, H.; Cui, L.; Yang, Q. Towards personalized federated learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *34*, 9587–9603. [CrossRef]
13. Jeong, W.; Yoon, J.; Yang, E.; Hwang, S.J. Federated Semi-Supervised Learning with Inter Client Consistency & Disjoint Learning. *arXiv* **2021**. [CrossRef]
14. Malinin, A.; Mlodozeniec, B.; Gales, M. Ensemble distribution distillation. *arXiv* **2019**, arXiv:1905.00076.
15. Cho, J.H.; Hariharan, B. On the efficacy of knowledge distillation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 4794–4802.
16. Gretton, A.; Smola, A.; Huang, J.; Schmittfull, M.; Borgwardt, K.; Schölkopf, B. Covariate shift and local learning by distribution matching. In *Dataset Shift in Machine Learning*; MIT Press: Cambridge, MA, USA, 2009; pp. 131–160.
17. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
18. Tian, Y.; Krishnan, D.; Isola, P. Contrastive Representation Distillation. *arXiv* **2022**. [CrossRef]
19. Li, T.; Hu, S.; Beirami, A.; Smith, V. Ditto: Fair and robust federated learning through personalization. In Proceedings of the International Conference on Machine Learning. PMLR, Online, 18–24 July 2021; pp. 6357–6368.
20. Fallah, A.; Mokhtari, A.; Ozdaglar, A. Personalized federated learning: A meta-learning approach. *arXiv* **2020**, arXiv:2002.07948.
21. Sui, D.; Chen, Y.; Zhao, J.; Jia, Y.; Xie, Y.; Sun, W. Feded: Federated learning via ensemble distillation for medical relation extraction. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 2118–2128.
22. Passalis, N.; Tefas, A. Learning deep representations with probabilistic knowledge transfer. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 268–284.
23. Huang, Z.; Wang, N. Like What You Like: Knowledge Distill via Neuron Selectivity Transfer. *arXiv* **2017**. [CrossRef]
24. Li, K.; Yang, J. Score-matching representative approach for big data analysis with generalized linear models. *Electron. J. Stat.* **2022**, *16*, 592–635. [CrossRef]
25. Wu, C.; Wu, F.; Lyu, L.; Huang, Y.; Xie, X. Communication-efficient federated learning via knowledge distillation. *Nat. Commun.* **2022**, *13*, 2032. [CrossRef]
26. Mu, X.; Shen, Y.; Cheng, K.; Geng, X.; Fu, J.; Zhang, T.; Zhang, Z. Fedproc: Prototypical contrastive federated learning on non-iid data. *Future Gener. Comput. Syst.* **2023**, *143*, 93–104. [CrossRef]
27. Székely, G.J. *E-Statistics: The Energy of Statistical Samples*; Bowling Green State University: Bowling Green, OH, USA , 2003; Volume 3, pp. 1–18.
28. Székely, G.J.; Rizzo, M.L. A new test for multivariate normality. *J. Multivar. Anal.* **2005**, *93*, 58–80. [CrossRef]
29. Fan, M.; Geng, B.; Shterenberg, R.; Casey, J.A.; Chen, Z.; Li, K. Measuring Heterogeneity in Machine Learning with Distributed Energy Distance. *arXiv* **2025**. [CrossRef]
30. Bennett, K.P.; Mangasarian, O.L. Robust linear programming discrimination of two linearly inseparable sets. *Optim. Methods Softw.* **1992**, *1*, 23–34. [CrossRef]
31. Li, D.; Wang, J. Fedmd: Heterogenous federated learning via model distillation. *arXiv* **2019**, arXiv:1910.03581.
32. Chen, H.Y.; Chao, W.L. Fedbe: Making bayesian model ensemble applicable to federated learning. *arXiv* **2020**, arXiv:2009.01974.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.