# Heterogeneous Ensemble Knowledge Transfer for Training Large Models in Federated Learning

**Yae Jee Cho**[1,2*], **Andre Manoel**[1], **Gauri Joshi**[2], **Robert Sim**[1] and **Dimitrios Dimitriadis**[1]
[1]Microsoft Research
[2]Carnegie Mellon University

## Abstract

Federated learning (FL) enables edge-devices to collaboratively learn a model without disclosing their private data to a central aggregating server. Most existing FL algorithms require models of identical architecture to be deployed across the clients and server, making it infeasible to train large models due to clients' limited system resources. In this work, we propose a novel ensemble knowledge transfer method named Fed-ET in which small models (different in architecture) are trained on clients, and used to train a larger model at the server. Unlike in conventional ensemble learning, in FL the ensemble can be trained on clients' highly heterogeneous data. Cognizant of this property, Fed-ET uses a weighted consensus distillation scheme with diversity regularization that efficiently extracts reliable consensus from the ensemble while improving generalization by exploiting the diversity within the ensemble. We show the generalization bound for the ensemble of weighted models trained on heterogeneous datasets that supports the intuition of Fed-ET. Our experiments on image and language tasks show that Fed-ET significantly outperforms other state-of-the-art FL algorithms with fewer communicated parameters, and is also robust against high data-heterogeneity.

## 1 Introduction

Moving both data collection and model training to the edge, federated learning (FL) has gained much spotlight since it was introduced [McMahan *et al.*, 2017]. In FL, a number of edge-devices (clients), like cell-phones or IoT devices, collaboratively train machine learning models without explicitly disclosing their local data. Instead of communicating their data, the clients locally train their models, and send model updates periodically to the aggregating server. The two distinctive challenges in FL are that clients can have i) limited system resources, and ii) heterogeneous local datasets [Kairouz *et al.*, 2019; Bonawitz *et al.*, 2019]. Many recent work in FL [Wang *et al.*, 2021] overlook the clients' resource constraints, using large homogeneous models on the clients and server. In practice, the clients do not have enough bandwidth or computing power to train large state-of-the-art (SOTA) models, and therefore, are restricted to train smaller and computationally lighter models. Moreover, a naive aggregation of the clients' models can hinder the convergence of the model due to high data-heterogeneity across the clients [Sahu *et al.*, 2020; Cho *et al.*, 2021; Ozkara *et al.*, 2021]. Based on these constraints, the global model trained on clients can fail to work well in practice.

A more realistic approach to learn from the resource-constrained clients in FL is by allowing different models to be deployed across clients depending on their system resources, all while training a larger model for the server. This presents a new challenge where clients return models not only trained on heterogeneous data, but also with different architecture (amongst themselves and the server). Hence, we raise the question: *How can we utilize an ensemble of different models trained on heterogeneous datasets to train a larger model at the server?* We draw insight from ensemble knowledge transfer [Hinton *et al.*, 2015; Allen-Zhu and Li, 2021] to investigate this problem in our work in the FL context.

Previous studies on ensemble knowledge transfer [Lan *et al.*, 2018; Hong *et al.*, 2021; Tran *et al.*, 2020; Park and Kwak, 2020], propose methods to transfer knowledge from *a bag of experts* to a target model, where the ensemble models are trained on similar datasets. These datasets are commonly generated from methods data augmentation or simple data shuffling. In FL, however, the models are trained on heterogeneous data distributions, where some models may show higher inference confidence than others, depending on the data sample used for knowledge transfer. Knowing which model is an *expert* than the others for each data sample is imperative for effective ensemble transfer in FL – especially when there are no hard labels for the data samples.

In this work, we propose a novel ensemble knowledge transfer algorithm for FL named Fed-ET which trains a large model at the server via training small and heterogeneous models at the resource-constrained and data heterogeneous clients. Inspired by the successful usage of knowledge transfer via unlabeled public data [Hinton *et al.*, 2015; Allen-Zhu and Li, 2021], Fed-ET leverages unlabeled data to perform a bi-directional ensemble knowledge transfer be-

*Work done while at Microsoft Research. Corresponding author email: {yaejeec@andrew.cmu.edu}.

| Method | Client Model Heterogeneity | Public Data | Client Access to Public Data | Server Model Size | Possible Tasks |
|---|---|---|---|---|---|
| FedAvg [McMahan *et al.*, 2017] | No | N/A | N/A | = Client Model | Any |
| FedProx [Sahu *et al.*, 2020] | No | N/A | N/A | = Client Model | Any |
| SCAFFOLD [Karimireddy *et al.*, 2020] | No | N/A | N/A | = Client Model | Any |
| MOON [Li *et al.*, 2021] | No | Unlabeled | Required | = Client Model | Only Image |
| FedDF [Lin *et al.*, 2020] | Yes | Unlabeled | Not Required | = Client Model | Any |
| DS-FL [Itahara *et al.*, 2021] | Yes | Unlabeled | Required | = Client Model | Any |
| FedGKT [He *et al.*, 2020] | Yes | N/A | N/A | > Client Model | Only Image |
| FedGEMS [Cheng *et al.*, 2021] | Yes | Labeled | Required | > Client Model | Any |
| **Fed-ET (ours)** | **Yes** | **Unlabeled** | **Not Required** | **> Client Model** | **Any** |

Table 1: Comparison of Related Work with Fed-ET

tween the server and client models. Unlike previous work in FL with knowledge distillation or with sole focus on tackling data-heterogeneity (see Table 1), Fed-ET allows client model-heterogeneity while training a larger model at the server, and can be used for any classification tasks. Moreover, Fed-ET does not impose any overhead to the clients nor assumes that the clients have access to additional data other than its private data. In Fed-ET clients simply perform local training as in standard FL while all the other computations are done by the server. Our main contributions are:

- We propose Fed-ET, the first ensemble transfer algorithm for FL (to the best of our knowledge) using unlabeled data that enables training a large server model with smaller models at the clients, for any classification task.
- We consider the data-heterogeneity in FL by proposing a weighted consensus distillation approach with diversity regularization in Fed-ET that effectively filters out *experts*, showing the corresponding generalization bounds.
- We show Fed-ET's efficacy with image and language classification tasks where Fed-ET achieves higher test accuracy, with more robustness against data-heterogeneity and fewer communication rounds, than other FL algorithms.

## 2 Background and Related Work

**Ensemble Knowledge Transfer.** Knowledge transfer from an ensemble of trained models to a target model has been studied in various areas of machine learning. In [Lan *et al.*, 2018], ensemble knowledge distillation for online learning is proposed where the teacher ensembles are trained on-the-fly to simultaneously train the teachers along with the target model. In [Hong *et al.*, 2021], ensemble reinforcement learning is investigated where an ensemble of policies share knowledge through distillation. In [Tran *et al.*, 2020], ensembles trained on shuffled data are used to transfer knowledge to a target model, and ways to utilize the diversity across these models to improve knowledge transfer are investigated.

The previous work mentioned above, however, is not directly applicable to FL because i) the local models are trained on heterogeneous data, and ii) FL is an iterative process with only a fraction of clients participating in every communication round. Since the server sends its knowledge back to a new set of clients every round in FL, an ensemble knowledge transfer scheme should have a well defined feedback loop from the target model to the ensemble of models. As such, our proposed Fed-ET induces a data-aware weighted consensus from the ensemble of models, with a feedback loop to transfer the server model's knowledge to the client models.

**FL with Knowledge Distillation.** Several studies investigated combining FL with knowledge distillation to allow different models across clients, or to improve the server model. In [Itahara *et al.*, 2021], an entropy-reduction aggregation method of the clients' logits is proposed, lowering the variance of the clients' outputs. In [He *et al.*, 2020], FedGKT is proposed specifically for image classification, using knowledge distillation across CNNs with small CNNs at the clients and a larger CNN at the server. In [Li *et al.*, 2021], MOON is also proposed for image tasks, where contrastive loss is used across models of identical architecture from the server and clients to improve the server model. In [Lin *et al.*, 2020], FedDF is proposed to aggregate heterogeneous models through knowledge distillation with unlabeled public data, but it does not take data-heterogeneity into account and the server model is restricted to the clients' models.

The aforementioned work in FL with knowledge distillation are limited to specific scenarios such as when we have labels in the public dataset, target only image tasks, or have low data-heterogeneity across the clients. Our proposed Fed-ET is not limited to these scenarios while still being able to outperform the baselines in Table 1 as shown in our experiments. We use a weighted consensus-based distillation scheme, where clients with higher inference confidence contribute more to the consensus compared to less confident clients. We also take use of a diversity regularization term, where clients that do not follow the consensus can still transfer useful representations to the server model.

## 3 Federated Ensemble Transfer: Fed-ET

We propose Fed-ET, an ensemble knowledge transfer framework that trains a large server model with small and heterogeneous models trained on clients, using an unlabeled public dataset[1]. Concisely, Fed-ET consists of the three consecutive steps: i) clients' local training and representation transfer, ii)

[1]Applicable datasets are accessible by the server through data generators (e.g., GAN), open-sourced repositories, or data markets.

weighted consensus distillation with diversity regularization, and iii) server representation transfer (see Figure 1).

### 3.1 Preliminaries

We consider a cross-device FL setup with a $N$-class classification task where $K$ clients are connected to a server. Each client $k \in [K]$ has its local training dataset $\mathcal{B}_k$ and each data sample $\xi$ is a pair $(\mathbf{x}, y)$ with input $\mathbf{x} \in \mathbb{R}^d$ and label $y \in [1, N]$. Each client has its local objective $F_k(\mathbf{w}) = \frac{1}{|\mathcal{B}_k|} \sum_{\xi \in \mathcal{B}_k} f(\mathbf{w}, \xi)$ with $f(\mathbf{w}, \xi)$ being the composite loss function. Having a large $\mathbf{w}$ with identical architecture across all resource-constrained clients, as done in the standard FL framework, can be infeasible. Moreover, the local minimums $\mathbf{w}_k^*$, $k \in [1, K]$ minimizing $F_k(\mathbf{w})$ can be different from each other due to data-heterogeneity. Fed-ET tackles these obstacles by training a large server model with data-aware ensemble transfer from the smaller models trained on clients.

Formally, we consider $U$ small and heterogeneous models at the server with $\mathcal{M} = \{1 : \overline{\mathbf{w}}_1, ..., U : \overline{\mathbf{w}}_U\}$ where $\mathcal{M}$ is the hashmap with the keys $1, ..., U$ as model ids, the values $\mathcal{M}[i] = \overline{\mathbf{w}}_i \in \mathbb{R}^{n_i}$ as the models, and $n_i$ as the number of parameters for $i \in [U]$. All of the small models in $\mathcal{M}$ have a representation layer $\overline{\mathbf{h}}_i \in \mathbb{R}^u$, $i \in [U]$, which includes the classification layer, connected to the end of their different model architectures $u \ll n_i, i \in [U]$. Each client is designated its model to use from $\mathcal{M}$ depending on its resource capability. With slight abuse of notation, we denote the model id chosen by client $k$ as $\mathcal{M}(k) \in [1, U]$, and the local model for that client $k \in [K]$ as $\mathbf{w}_k = \overline{\mathbf{w}}_{\mathcal{M}(k)} = \mathcal{M}[\mathcal{M}(k)]$ which has its respective representation layer defined as $\mathbf{h}_k$.

The server has its large model defined as $\overline{\mathbf{w}} \in \mathbb{R}^n$ also with its representation layer defined as $\overline{\mathbf{h}} \in \mathbb{R}^u$. The large server model is assumed to be much larger than the small server models in $\mathcal{M}$, i.e., $n \gg n_i, i \in [U]$. As shown in the following sections, the representation layers $\overline{\mathbf{h}}$ and $\mathbf{h}_k, k \in [K]$ are shared bidirectionally between clients and server to transfer the representations learned from their respective training. Only the server has access to an unlabeled public dataset denoted as $\mathcal{P}$. The local models $\mathbf{w}_k, k \in [K]$, and large server model $\overline{\mathbf{w}}$ output soft-decisions (logits) over the pre-defined number of classes $N$, which is a probability vector over the $N$ classes. We refer to the soft-decision of model $\mathbf{w}_k$ over any input data $\mathbf{x}$ in either the private or public dataset as $s(\mathbf{w}_k, \mathbf{x}) : \mathbb{R}^{n_{\mathcal{M}(k)}} \times (\mathcal{B}_k \cup \mathcal{P}) \rightarrow \Delta_N$, where $\Delta_N$ stands for the probability simplex over $N$.

### 3.2 Ensemble Transfer with Federated Learning

**Step 1: Client Local Training & Representation Transfer.** For each communication round $t$, the server gets the set of $m < K$ clients, denoted as $\mathcal{S}^{(t,0)}$, by selecting them in proportion to their dataset size. The upper-subscript $(t, r)$ denotes for $t$-th communication round and $r$-th local iteration. Note that $\mathcal{S}^{(t,0)}$ is independent of the local iteration index. For each client $k \in \mathcal{S}^{(t,0)}$, the most recent version of its designated model $\mathbf{w}_k^{(t,0)} = \overline{\mathbf{w}}_{\mathcal{M}(k)}^{(t,0)} = \mathcal{M}[\mathcal{M}(k)]$ is sent from the server to the client. The clients perform local mini-batch stochastic-gradient descent (SGD) steps on their local model $\mathbf{w}_k^{(t,0)}$ with their private dataset $\mathcal{B}_k, k \in [K]$. Accordingly, the clients $k \in \mathcal{S}^{(t,0)}$ perform $\tau$ local updates so that for every communication round their local models are updated as:

$$\mathbf{w}_k^{(t,\tau)} = \mathbf{w}_k^{(t,0)} - \frac{\eta_t}{b} \sum_{r=0}^{\tau-1} \sum_{\xi \in \xi_k^{(t,r)}} \nabla f(\mathbf{w}_k^{(t,r)}, \xi) \quad (1)$$

where $\eta_t$ is the learning rate and $\frac{1}{b} \sum_{\xi \in \xi_k^{(t,r)}} \nabla f(\mathbf{w}_k^{(t,r)}, \xi)$ is the stochastic gradient over mini-batch $\xi_k^{(t,r)}$ of size $b$ randomly sampled from $\mathcal{B}_k$. After the clients $k \in \mathcal{S}^{(t,0)}$ finish their local updates, the models $\mathbf{w}_k^{(t,\tau)}, k \in \mathcal{S}^{(t,0)}$ are sent to the server. Each client has different representation layers $\mathbf{h}_k^{(t,\tau)}$ in their respective models $\mathbf{w}_k^{(t,\tau)}, k \in \mathcal{S}^{(t,0)}$. The server receives these models from the clients and updates its large model's representation layer with the ensemble models as $\overline{\mathbf{h}}^{(t,0)} = \frac{1}{m} \sum_{k \in \mathcal{S}^{(t,0)}} \mathbf{h}_k^{(t,\tau)}$. This pre-conditions the large server model with the clients' representations for Step 2 where we train the large server model with the ensemble loss.

**Step 2: Ensemble Loss by Weighted Consensus with Diversity Regularization.** Next, the large server model is trained via a weighted consensus based knowledge distillation scheme from the small models received from the clients. A key characteristic of the ensemble is that each model may be trained on data samples from different data distributions. Hence, some clients can be more confident than others on each of the public data samples. However, all clients may still have useful representations to transfer to the server, even when they are not very confident about that particular data sample. Hence Fed-ET proposes a weighted consensus distillation scheme with diversity regularization, where the large server model is trained on the consensus knowledge from the ensemble of models while regularized by the clients that do not follow the consensus.

**Weighted Consensus.** First we derive a reliable consensus over the ensemble of models by evaluating the variance within the logit vectors $s(\mathbf{w}_k^{(t,\tau)}, \mathbf{x})$, $\mathbf{x} \in \mathcal{P}$ for each client $k \in \mathcal{S}^{(t,0)}$. We denote this variance as $\sigma_s^2(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) := \text{Var}(s(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}))$, which is the variance taken over the $N$ total probability values for the $N$-multi-class classification task. Higher $\sigma_s^2(\mathbf{w}_k^{(t,\tau)}, \mathbf{x})$ indicates a more confident client $k$ about how well it models data sample $\mathbf{x}$, and vice-versa [Camacho-Gómez *et al.*, 2021]. Hence, we weigh the logits from the clients with high $\sigma_s^2(\mathbf{w}_k^{(t,\tau)}, \mathbf{x})$ more heavily compared to low-variance logit clients. Formally, we set a confidence based weighted average over the logits for each data sample $\mathbf{x} \in \mathcal{P}$ denoted as:

$$s^{(t,\tau)}(\mathbf{x}) = \sum_{k \in \mathcal{S}^{(t,0)}} \alpha_k^{(t,\tau)}(\mathbf{x}) s(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) \quad (2)$$

where the weights are defined as:

$$\alpha_k^{(t,\tau)}(\mathbf{x}) = \sigma_s^2(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) / \sum_{l \in \mathcal{S}^{(t,0)}} \sigma_s^2(\mathbf{w}_l^{(t,\tau)}, \mathbf{x}) \quad (3)$$

The resulting weighted consensus logit $s^{(t,\tau)}(\mathbf{x})$ efficiently derives the consensus out of the ensemble of models trained

weighted logits of ensemble clients based on unlabeled public dataset

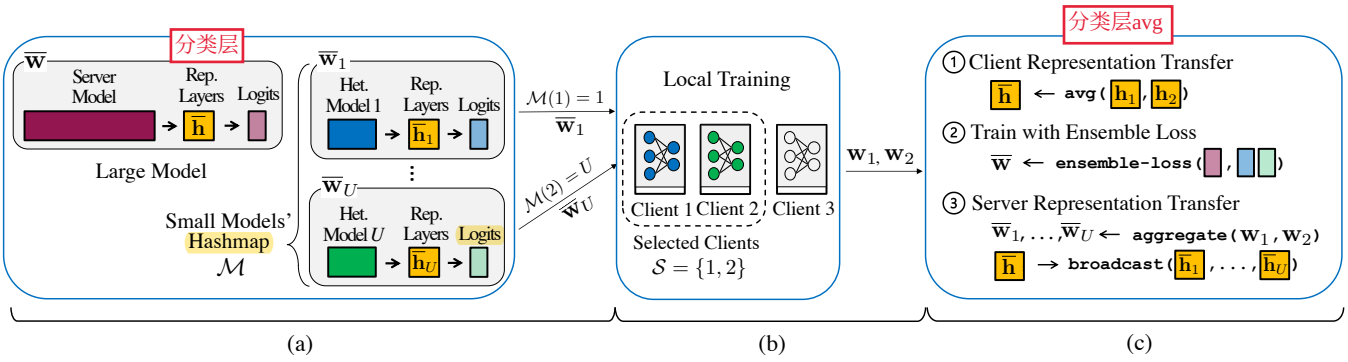


Figure 1: Overview of the Fed-ET framework with 3 clients and $U$ small models; (a): server sends the predesignated small models in $\mathcal{M}$ to the selected clients; (b): clients perform local training and send the updates to the server; (c): server updates its large model with the received updates with FedET's primary 3 steps for ensemble transfer (see Section 3.2).

on heterogeneous datasets due to filtering out the following two main adversaries: **i) the non-experts with low intra-variance within each logit**, and **ii) overly-confident but erroneous outliers** by utilizing the power of ensemble where multiple experts contribute to the consensus.

For each data sample $\mathbf{x}$ we get the most probable label from $s^{(t,\tau)}(\mathbf{x})$ as:

半监督得到pesudo label, 选最大的weighted logits

$$y_s^{(t,\tau)}(\mathbf{x}) = \underset{\text{label}\in[0,N-1]}{\arg\max} \; s^{(t,\tau)}(\mathbf{x}) \tag{4}$$

The pair $(\mathbf{x},\ y_s^{(t,\tau)}(\mathbf{x}))$, $\mathbf{x}\in\mathcal{P}$ is the consensus-derived data sample from the unlabeled public dataset $\mathcal{P}$, which is then used to train the server model with the cross-entropy loss $l((\mathbf{x},\ y_s^{(t,\tau)}(\mathbf{x})), \overline{\mathbf{w}}^{(t,0)})$. The cross-entropy loss term used in the final ensemble loss for training the server model is:

$$\frac{1}{|\mathcal{P}|}\sum_{\mathbf{x}\in\mathcal{P}} l((\mathbf{x}, y_s^{(t,\tau)}(\mathbf{x})), \overline{\mathbf{w}}^{(t,0)}) \tag{5}$$

**Diversity Regularization.** While the confidence based weighted consensus can derive a more reliable consensus from the ensemble, the diversity across the participating models is less represented. Meaningful representation information of what clients learned from their private data should be included, even when certain clients have low-confidence and may have different logits from the consensus. Encouraging diversity across models can improve the generalization performance of ensemble learning [Tran *et al.*, 2020; Park and Kwak, 2020]. Hence, we gather the logits from the clients that do not coincide with the consensus, formally,

$$\mathcal{S}_{div}^{(t,0)}(\mathbf{x}) = \{l : y_s^{(t,\tau)}(\mathbf{x}) \neq \underset{\text{label}\in[0,N-1]}{\arg\max} \; s(\mathbf{w}_l^{(t,\tau)}, \mathbf{x}) \cap \; l \in \mathcal{S}^{(t,0)}\} \tag{6}$$

and formulate a regularization term:

$$s_{div}^{(t,\tau)}(\mathbf{x}) = \sum_{k\in\mathcal{S}_{div}^{(t,0)}} \alpha(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) s(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) \tag{7}$$

where the weights are

$$\alpha_k(\mathbf{x}) = \sigma_s^2(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) / \sum_{l\in\mathcal{S}^{(t,0)}} \sigma_s^2(\mathbf{w}_l^{(t,\tau)}, \mathbf{x}) \tag{8}$$

Accordingly, the diversity regularization term for the final ensemble loss is where $KL(\cdot,\cdot)$ is the KL-divergence loss between two logits:

$$KL(s_{div}^{(t,\tau)}(\mathbf{x}), s(\overline{\mathbf{w}}^{(t,0)}, \mathbf{x})) \tag{9}$$

**Final Ensemble Loss.** Finally, combining the weighted consensus based cross-entropy loss in (5) with the diversity regularization in (9), the server model is updated, in every communication round $t$, by minimizing the following objective function:

CE loss, argmax + KL loss, weighted logits

$$F(\overline{\mathbf{w}}^{(t,0)}) = \frac{1}{|\mathcal{P}|}\sum_{\mathbf{x}\in\mathcal{P}} l((\mathbf{x}, y_s^{(t,\tau)}(\mathbf{x})), \overline{\mathbf{w}}^{(t,0)}) + \lambda KL(s_{div}^{(t,\tau)}(\mathbf{x}), s(\overline{\mathbf{w}}^{(t,0)}, \mathbf{x})) \tag{10}$$

To minimize the ensemble loss in (10), instead of going through the entire dataset $\mathcal{P}$, the server model takes $\tau_s$ mini-batch SGD[2] steps by sampling a mini-batch $\xi_{\mathcal{P}}^{(t,r')}$, $r'\in[0,\tau_s-1]$ of $b_s$ data samples from $\mathcal{P}$ uniformly at random without replacement. Then, for every communication round $t$ the server performs:

$$\overline{\mathbf{w}}^{(t,\tau_s)} = \overline{\mathbf{w}}^{(t,0)} - \frac{\eta_t}{b_s}\sum_{r=0}^{\tau_s-1}\sum_{\xi\in\xi_{\mathcal{P}}^{(t,r)}} \Big[\nabla l((\xi, y_s^{(t,\tau)}(\xi)), \overline{\mathbf{w}}^{(t,r)}) + \lambda\nabla KL(s_{div}^{(t,\tau)}(\xi), s(\overline{\mathbf{w}}^{(t,r)}, \xi))\Big] \tag{11}$$

Note that neither the weighted ensemble term nor the diversity regularization term dominates the ensemble loss in (10) with a reasonable $\lambda$ (see Table 4) and further because each term comes from a different set of clients. The former term is from the majority of the clients following the consensus, while the latter term is from the other clients that do not coincide with that consensus. Due to data-heterogeneity, these two different sets of clients likely change every round making it difficult for a single term to dominate the ensemble loss during training.

[2]Herein, SGD is depicted without loss of generality for other optimization algorithms.

**Algorithm 1** Federated Ensemble Transfer: Fed-ET

**Initialize:** Hashmap of Heterogeneous Models: $\mathcal{M} = \{1 : \overline{\mathbf{w}}_1^{(0,0)}, ..., U : \overline{\mathbf{w}}_U^{(0,0)}\}$; Designated Model Ids for each client $k \in [K]$: $\mathcal{M}(k) \in [1, U]$; Selected set of $m < K$ clients: $\mathcal{S}^{(0,0)}$
每个client对应1到U sub model
**Output:** $\overline{\mathbf{w}}^{(T,0)}$
**For** $t = 0, ..., T-1$ **communication rounds do**:
  **Clients** $k \in \mathcal{S}^{(t,0)}$ **in parallel do:**
    Receive $\mathbf{w}_k^{(t,0)} = \overline{\mathbf{w}}_{\mathcal{M}(k)}^{(t,0)} = \mathcal{M}[\mathcal{M}(k)]$ from server
    Update $\mathbf{w}_k^{(t,\tau)}$ with (1) and send it to server
  **Server do:**
    Receive all updated local model $\mathbf{w}_k^{(t,\tau)}, k \in \mathcal{S}^{(t,0)}$.
分类层Avg
    Transfer client representation $\overline{\mathbf{h}}^{(t,0)} = \frac{1}{m}\sum_{\mathcal{S}^{(t,0)}} \mathbf{h}_k^{(t,\tau)}$
    Update $\overline{\mathbf{w}}^{(t,\tau_s)}$ with (11) and update $\mathcal{M}$ with (12)
    Transfer server representation $\overline{\mathbf{h}}^{(t,\tau_s)}$ models in $\mathcal{M}$
    Get $\mathcal{S}^{(t+1,0)}$ by sampling in proportion to dataset sizes

具有相同结构的small model做avg

**Step 3: Server's Representation Transfer.** Finally, we update the server's small models in $\mathcal{M}$ by aggregating the received clients' models with identical architecture by simple averaging. Concretely, with $\mathcal{S}_i^{(t,0)} := \{k : k \in \mathcal{S}^{(t,0)} \cap \mathcal{M}(k) = i\}$, $i \in [U]$, we update $\mathcal{M}$ as

$$\mathcal{M}[i] = \overline{\mathbf{w}}_i^{(t+1,0)} = \frac{1}{\left|\mathcal{S}_i^{(t,0)}\right|} \sum_{k \in \mathcal{S}_i^{(t,0)}} \mathbf{w}_k^{(t,\tau)}, \; i \in [U] \quad (12)$$

server的分类层单独再传播给所有small model

After this update, the updated $\overline{\mathbf{h}}^{(t,\tau_s)}$ from the server model $\overline{\mathbf{w}}^{(t,\tau_s)}$ is transferred to all the models in $\mathcal{M}$.

**The algorithm of Fed-ET.** In the preceding paragraphs, we have shown three essential components of Fed-ET for federated ensemble transfer with heterogeneous models trained on heterogeneous data distributions. The complete algorithm of Fed-ET can be obtained by using these components in tandem as described in Algorithm 1. Note that FedET is easily extendable to allow clients to define their own model architectures depending on their computing resources.

### 3.3 Generalization Bound for Ensemble Transfer

In Fed-ET, an ensemble of small models trained on heterogeneous data distributions is used to train a large server model for its target test data distribution. We show the generalization properties of a weighted ensemble of models trained on heterogeneous datasets in respect to the server's target distribution, supporting the weighted consensus distillation process of Fed-ET. We consider hypotheses $h : \mathcal{X} \to \mathcal{Y}$, with input space $\mathbf{x} \in \mathcal{X}$, label space $y \in \mathcal{Y}$, and hypotheses space $\mathcal{H}$. The loss function $l(h(\mathbf{x}), y)$ measures the classification performance of $h$ for a single data point $(\mathbf{x}, y)$. We define the expected loss over all data points for an arbitrary data distribution $\mathcal{D}'$ as $\mathcal{L}_{\mathcal{D}'}(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}'}[l(h(\mathbf{x}), y)]$ for $h \in \mathcal{H}$ and assume that $\mathcal{L}(h)$ is convex with range $[0, 1]$. We now present the generalization bound for the server's target distribution with respect to an ensemble of weighted models trained on heterogeneous datasets below in Theorem 1.

**Theorem 1.** *With $K$ clients and a server for FL, we have $\mathcal{D}$ as the server's target test data distribution, and $\mathcal{D}_k$, $\widehat{\mathcal{D}}_k$ as the true and empirical data distribution, respectively, for client $k \in [K]$. We define $h_k = \arg\min_h \mathcal{L}_{\mathcal{D}_k}(h)$ and $\widehat{h}_k = \arg\min \mathcal{L}_{\widehat{\mathcal{D}}_k}(h)$. Then, we have for the weighted ensemble of models $\sum_{i=1}^K \alpha_i h_{\widehat{\mathcal{D}}_i}$ for $K$ clients with arbitrary weights $\alpha_i, i \in [K]$, $\sum_{i=1}^K \alpha_i = 1$, with probability at least $1-\delta$ over the choice of samples, the bound:*

$$\mathcal{L}_{\mathcal{D}}\left(\sum_{i=1}^K \alpha_i h_{\widehat{\mathcal{D}}_i}\right) \leq \sum_{i=1}^K \alpha_i \mathcal{L}_{\widehat{\mathcal{D}}_i}(h_{\widehat{\mathcal{D}}_i}) + \sqrt{\log \delta^{-1}} \sum_{i=1}^K \frac{\alpha_i}{\sqrt{|\mathcal{B}_i|}} + \frac{1}{2}\sum_{i=1}^K \alpha_i d(\mathcal{D}_i, \mathcal{D}) + \sum_{i=1}^K \alpha_i \nu_i \quad (13)$$

*where $\nu_i = \inf_h \mathcal{L}_{\mathcal{D}_i}(h) + \mathcal{L}_{\mathcal{D}}(h)$ and $d(\mathcal{D}_i, \mathcal{D})$ measures the distribution discrepancy between two distributions.*

In Theorem 1, the first, second, and third terms in the upper bound show that the generalization performance of the ensemble transfer worsens by the following qualities of each clients: i) bad local model quality on its own training data, ii) small training dataset size, and iii) large discrepancy between its data distribution $\mathcal{D}_i, i \in [K]$ and server's target data distribution $\mathcal{D}$. Fed-ET aims in giving lower weights to the clients that demonstrate i) and iii) by weighted consensus distillation where the confidence levels and multiple inferences of the clients contribute to the consensus, so that erroneous outliers can be filtered out. The effect of ii) is also considered in Fed-ET by sampling clients in proportion to their dataset sizes.

## 4 Experiments

For all experiments, partial client participation is considered where 10 clients are sampled from the 100 clients for image tasks and the 106 clients for the language task.

**Datasets.** For image datasets, the training dataset is partitioned data heterogeneously amongst a total of 100 clients using the Dirichlet distribution $\text{Dir}_K(\alpha)$ [Hsu *et al.*, 2019]. The public dataset is generated by applying a different data transformation to the data samples (non-overlapping with either the training or test dataset) to further differentiate it with the training dataset. For the language task, we use sentiment classification with Sent140 (Twitter) dataset. For the training dataset, users with more than 100 data samples are treated as the FL clients, leading to a total of 106 clients. For all datasets, non-overlapping users' data samples are used.

**Models.** For image tasks, we set a CNN, ResNet8, and ResNet18 [He *et al.*, 2016] for the small server models, and a VGG19 [Simonyan and Zisserman, 2014] for the large server model. For language tasks, a Tiny-BERT [Bhargava *et al.*, 2021] and a LSTM classifier are set for the small server models, and a Mini-BERT [Bhargava *et al.*, 2021] is set for the large server model. For the representation layers we use a small MLP with dimension 128 which is a small increase in the model size. The small server models in $\mathcal{M}$ are designated (prior to training) to the clients uniformly at random.

| | | $\alpha = 0.1$ (Higher Data-Het.) | | $\alpha = 0.5$ (Lower Data-Het.) | | N/A |
|---|---|---|---|---|---|---|
| | Method | CIFAR10 | CIFAR100 | CIFAR10 | CIFAR100 | Sent140 |
| Model Homogeneous | FedAvg | 71.19 ($\pm$0.27) | 30.21 ($\pm$0.32) | 74.82 ($\pm$0.23) | 33.12 ($\pm$0.13) | 71.51 ($\pm$0.45) |
| | FedProx | 72.45 ($\pm$0.13) | 31.51 ($\pm$0.11) | 75.24 ($\pm$0.19) | 33.63 ($\pm$0.08) | 71.32 ($\pm$0.31) |
| | Scaffold | 75.12 ($\pm$0.20) | 30.61 ($\pm$0.57) | 78.69 ($\pm$0.15) | 34.91 ($\pm$0.61) | 73.28 ($\pm$0.35) |
| | MOON | 75.68 ($\pm$0.51) | 33.72 ($\pm$0.89) | **81.17** ($\pm$0.41) | **42.15** ($\pm$0.72) | N/A |
| Model Heterogeneous | FedDF | 73.81 ($\pm$0.42) | 31.87 ($\pm$0.46) | 76.55 ($\pm$0.32) | 37.87 ($\pm$0.31) | 72.19 ($\pm$0.43) |
| | DS-FL | 65.27 ($\pm$0.53) | 29.12 ($\pm$0.51) | 68.44 ($\pm$0.47) | 33.56 ($\pm$0.55) | 63.12 ($\pm$0.71) |
| | Fed-ET (ours) | **78.66** ($\pm$0.31) | **35.78** ($\pm$0.45) | 81.13 ($\pm$0.28) | 41.58 ($\pm$0.36) | **75.78** ($\pm$0.39) |

Table 2: Best test accuracy achieved by Fed-ET and baselines with varying data-heterogeneity. The large server model is used for evaluation for the model homogeneous baselines and the model heterogeneous baselines that require separate server models.

| | CIFAR10 | CIFAR100 | Sent140 |
|---|---|---|---|
| Method | $C_{\text{acc}}(70\%)$ | $C_{\text{acc}}(30\%)$ | $C_{\text{acc}}(70\%)$ |
| FedAvg | $72 \times 10^9$ | $87 \times 10^9$ | $25 \times 10^9$ |
| FedProx | $70 \times 10^9$ | $86 \times 10^9$ | $22 \times 10^9$ |
| Scaffold | $68 \times 10^9$ | $79 \times 10^9$ | $19 \times 10^9$ |
| MOON | $75 \times 10^9$ | $90 \times 10^9$ | N/A |
| Fed-ET (ours) | $\mathbf{26 \times 10^9}$ | $\mathbf{31 \times 10^9}$ | $\mathbf{9 \times 10^9}$ |

Table 3: Communication cost to achieve the target test accuracy $x$ ($C_{\text{acc}}(x)$) for $\alpha = 0.1$.

**Baselines.** We consider two types of baselines: i) model homogeneous (FedAvg, FedProx, Scaffold, MOON) and ii) model heterogeneous (FedDF, DS-FL). FedGKT assumes full client participation, thus a direct comparison with Fed-ET is not possible. For model homogeneous, we use the large server model for evaluation. For model heterogeneous we use the small server models for the client models and the large server model for the server model (if a separate server model is required).

**Effectiveness of Fed-ET.** In Table 2, we show the best achieved test accuracy of Fed-ET and the baselines for different degrees of data-heterogeneity. Fed-ET achieves higher test accuracy for CIFAR10 with high data-heterogeneity ($\alpha = 0.1$) and Sent140 compared to both the model homogeneous and model heterogeneous baselines. Specifically, for $\alpha = 0.1$, MOON achieves $75\%$ and $33\%$ for CIFAR10 and CIFAR100 respectively at the cost of communicating directly the large VGG19 while Fed-ET achieves higher accuracy of $78\%$ and $35\%$ respectively while using smaller models than VGG19 for the clients. For lower data-heterogeneity ($\alpha = 0.5$), MOON slightly out-performs Fed-ET by around $1\%$ but at the cost of training larger models at the clients.

**Communication Efficiency.** The communication efficiency of Fed-ET is shown in Table 3. We compare the communication cost $C_{\text{acc}}(x)$, the total number of model parameters communicated between the server and clients during training to achieve test accuracy $x$. The baselines in Table 3 require model-homogeneity, and hence communicate the large server model, while Fed-ET communicates the smaller models in $\mathcal{M}$ for each round. Fed-ET is able to achieve the target test accuracy with approximately $3\times$ less number of communicated parameters compared to those of the baselines. Fed-ET enables efficient training with smaller models at the clients, while achieving comparable performance to when large models are used at the clients.

| | Diversity Parameter ($\lambda$) | | |
|---|---|---|---|
| Datasets | 0 | 0.05 | 0.5 |
| CIFAR10 | 76.55 ($\pm$0.25) | **78.66** ($\pm$0.31) | 75.29 ($\pm$0.31) |
| CIFAR100 | 31.71 ($\pm$0.43) | **35.78** ($\pm$0.45) | 30.18 ($\pm$0.55) |
| Sent140 | 72.11 ($\pm$0.28) | 74.37 ($\pm$0.42) | **75.78** ($\pm$0.39) |

Table 4: Effect of diversity regularization in Fed-ET on test accuracy with different values of $\lambda$ for $\alpha = 0.1$.

**Effect of the Diversity Parameter $\lambda$.** In Table 4, we show the performance of Fed-ET with different values of $\lambda$, which modulates the diversity regularization term in (10). With $\lambda = 0$, Fed-ET only uses the weighted consensus to train the large server model without leveraging the diversity across the clients' models. A larger $\lambda$ indicates larger regularization loss to include more diversity across the clients' models. For image tasks the best performance is achieved with $\lambda = 0.05$, indicating that diversity indeed helps in improving generalization of the server model when moderately applied to the training. For the language task, a larger $\lambda = 0.5$ achieves the best performance, demonstrating that depending on the task, more inclusion of the diversity across the models in the ensemble can increase the generalization performance.

## 5 Conclusion

Motivated by the rigid constraint of deploying identical model architectures across the clients/server in many FL algorithms, we propose Fed-ET, an ensemble knowledge transfer framework to train large server models with smaller models trained on clients. Without additional overhead at the clients, Fed-ET transfers knowledge to the target model with a data-aware weighted consensus distillation from an ensemble of models trained on heterogeneous data. Fed-ET achieves high test accuracy with significantly lower communication overhead and robustness against data-heterogeneity. Relevant future steps are evaluating different deploying strategies of heterogeneous models to the clients and extending Fed-ET to a general ensemble knowledge transfer framework.

## References


[Allen-Zhu and Li, 2021] Zeyuan Allen-Zhu and Yuanzhi Li. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*, July 2021.

[Bhargava *et al.*, 2021] Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. Generalization in nli: Ways (not) to go beyond simple heuristics. *arXiv preprint arXiv:2110.01518*, 2021.

[Bonawitz *et al.*, 2019] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards Federated Learning at Scale: System Design. *SysML*, April 2019.

[Camacho-Gómez *et al.*, 2021] Carlos Camacho-Gómez, Sancho Salcedo-Sanz, and David Camacho. *A Review on Ensemble Methods and their Applications to Optimization Problems*, pages 25–45. Springer Singapore, Singapore, 2021.

[Cheng *et al.*, 2021] Sijie Cheng, Jingwen Wu, Yanghua Xiao, Yang Liu, and Yang Liu. Fedgems: Federated learning of larger server models via selective knowledge fusion. *arXiv preprint arXiv:2110.11027*, December 2021.

[Cho *et al.*, 2021] Yae Jee Cho, Jianyu Wang, Tarun Chiruvolu, and Gauri Joshi. Personalized federated learning for heterogeneous clients with clustered knowledge transfer. *arXiv preprint arXiv:2109.08119*, 2021.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[He *et al.*, 2020] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. In *Advances in Neural Information Processing Systems*, 2020.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, March 2015.

[Hong *et al.*, 2021] Zhang-Wei Hong, Prabhat Nagarajan, and Guilherme Maeda. Periodic intra-ensemble knowledge distillation for reinforcement learning. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2021.

[Hsu *et al.*, 2019] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. In *International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with NeurIPS 2019 (FL-NeurIPS'19)*, December 2019.

[Itahara *et al.*, 2021] Sohei Itahara, Takayuki Nishio, Yusuke Koda, Masahiro Morikura, and Koji Yamamoto. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *arXiv preprint arXiv:2008.06180*, 2021.

[Kairouz *et al.*, 2019] Peter Kairouz, H. Brendan McMahan, Brendan Avent, and Aurelien Bellet et. al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[Karimireddy *et al.*, 2020] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for on-device federated learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[Lan *et al.*, 2018] X. Lan, X. Zhu, and S. Gong. Knowledge distillation by on-the-fly native ensemble. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems.*, pages 7528–7538, 2018.

[Li *et al.*, 2021] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. *In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[Lin *et al.*, 2020] Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In *Advances in Neural Information Processing Systems*, 2020.

[McMahan *et al.*, 2017] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agøura y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. *International Conference on Artificial Intelligenece and Statistics (AISTATS)*, April 2017.

[Ozkara *et al.*, 2021] Kaan Ozkara, Navjot Singh, Deepesh Data, and Suhas Diggavi. Quped: Quantized personalization via distillation with applications to federated learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 3622–3634, 2021.

[Park and Kwak, 2020] SeongUk Park and Nojun Kwak. Feature-level ensemble knowledge distillation for aggregating knowledge from multiple networks. In *European Conference on Artificial Intelligence (ECAI)*, 2020.

[Sahu *et al.*, 2020] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. Federated optimization for heterogeneous networks. In *Proceedings of the 3rd MLSys Conference*, January 2020.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[Tran *et al.*, 2020] Linh Tran, Bastiaan S. Veeling, and Kevin Roth et. al. Hydra: Preserving ensemble diversity for model distillation. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2020.

[Wang *et al.*, 2021] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.