SemiFDA: Domain Adaptation in Semi-Supervised Federated Learning

Michele Craighero*, Giorgio Rossi*, Beatrice Rossi†, Diego Carrera†,
Diego Stucchi†, Pasqualina Fragneto† and Giacomo Boracchi*

*Politecnico di Milano, Milano (MI) 20133, Italy
{name.surname}@polimi.it

†STMicroelectronics, Agrate Brianza (MB) 20864, Italy
{name.surname}@st.com

***Abstract*—Semi-Supervised Federated Learning (SSFL) aims to improve a pretrained model using unlabeled data from clients. Traditional SSFL solutions relying on pseudo-labels or autoencoders often struggle in the presence of domain shift, i.e. a difference in data distributions between the server and the clients. In this paper we present SemiFDA, the first solution to effectively handle domain shift in SSFL. After training an initial classifier on the server's labeled data, we establish an unsupervised learning process at clients to train feature extractors based on encoders. This process adopts a custom unsupervised loss function that promotes the clients' encoders to align their feature distributions with those extracted by the encoder at server. The updated encoders are then aggregated at the server using Federated Averaging and sent back for the next iteration, while the classification head remains frozen to preserve the benefits of aligning features locally. Furthermore, we design an experimental framework to mimic various levels of domain shift and test SSFL methods in real-world scenarios, including HAR and Digit Classification. Our results also demonstrate the detrimental effects of domain shift in SSFL and show that SemiFDA outperforms other solutions under these challenging conditions.**

I. Introduction

Semi-Supervised Federated Learning (SSFL) is a subfield of Federated Learning (FL) that aims to improve a model by using partially labeled data. SSFL approaches can be categorized as *labels-at-clients*, when clients have access to both labeled and unlabeled data, and *labels-at-server*, when clients have only unlabeled data and the server holds the labeled data. This work focuses on the latter case, which is more realistic, since clients in many FL tasks, such as Human Activity Recognition (HAR), cannot label their own data automatically. Despite often ignored, SSFL settings typically imply an additional challenge: *domain shift*, which refers to differences in data distributions between the server and clients or among the clients themselves. Many tasks, including HAR, are inherently affected by a form of domain shift called *client heterogeneity*, as clients (or subjects) vary in physical traits, sensing equipment, and how they carry out activities.

Popular SSFL methods often combine traditional *Semi-Supervised Learning* (SSL) solutions based on *pseudo-labeling* with Federated Averaging [7], such as in FedMatch [6] and FedRGD [13]. The state-of-the-art SSFL solution based on pseudo-labels is SemiFL [4], which uses advanced data augmentations and consistency regularization techniques, combining [2] and [8], with an alternating training strategy. Other SSFL methods do not rely on pseudo-labels and instead perform unsupervised learning at the clients by using AutoEncoders (AEs) [15] or GANs [14]. However, all these methods are ineffective in the presence of domain shift, which may lead to incorrect pseudo-labels negatively impacting performance [11]. Moreover, standard approaches based on AEs also struggle with diverse and realistic datasets [5].

In this paper, we introduce SemiFDA, a new SSFL framework that incorporates an unsupervised adaptation process to address domain shift. SemiFDA grounds on an iterative SSFL scheme where clients have only access to unlabeled data and locally fine-tune their model encoders. The encoders updated by the clients are then aggregated by the server and sent back for the next iteration. The rationale behind SemiFDA is to train the clients' encoders in an unsupervised manner by aligning their latent feature spaces with that of the encoder at the server. This alignment enables a classification head, initially trained on labeled server data, to classify both server and clients data effectively. We achieve this by minimizing a custom unsupervised loss function inspired by the CORAL loss [9] and employing and encoder-only aggregation scheme which avoids retraining the whole model. To benchmark the impact of domain shift in SSFL, we design a novel experimental framework where competing SSFL methods are tested under different levels of domain shift. We test our framework on two learning tasks: HAR and Digit Classification, specifically organizing the datasets to mimic different levels of domain shift. We underline that our experimental framework is general and can be used for other learning tasks as well.

To summarize, our contributions are the following:

- We introduce SemiFDA[1], the first effective solution to face domain shift in SSFL. SemiFDA incorporates: *i*) a domain adaptation strategy based on a custom unsupervised loss and *ii*) a new aggregation scheme at server.
- We design a novel experimental framework to reproduce different levels of domain shift and test SSFL methods.
- Through extensive experiments, we show the detrimental effects caused by the domain shift in SSFL and we prove that SemiFDA outperforms several competitors.

[1]The code is available at https://github.com/Michelec1997/SemiFDA.

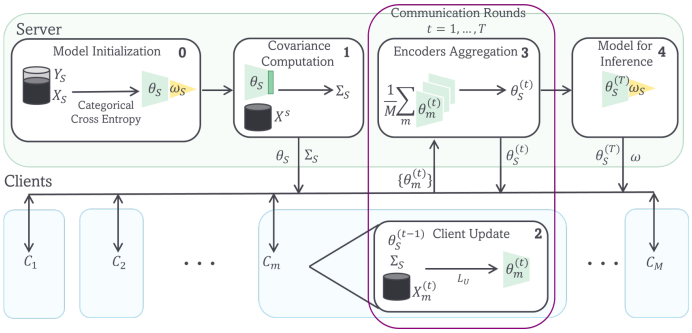

Fig. 1. Illustrative description of SemiFDA, our solution to handle domain shift in SSFL. For a detailed explanation of the steps, refer to Section III-C.

II. Problem Formulation

Let $\mathcal{S}$ represent a *server* and let $\mathcal{C}_1, \ldots, \mathcal{C}_M$ denote a set of *clients* that communicate with $\mathcal{S}$ and collaborate in a SSFL scheme. We consider a labels-at-server scenario, where the server $\mathcal{S}$ has a *labeled dataset* $\{X_S, Y_S\}$, with X_S denoting the inputs and Y_S the corresponding labels. Each client $\mathcal{C}_m$ has access to its *unlabeled dataset* X_m. Let $\phi^s = \phi^s(x, y)$ and $\phi^m = \phi^m(x, y)$ denote the joint distribution of inputs and labels on the server and on the client $\mathcal{C}_m$, respectively. In our domain shift settings we assume that there is at least a client $\mathcal{C}_m$ for which ϕ^s and ϕ^m are different, and potentially different clients are characterized by different distributions.

Our goal is to train a single classification model $\mathcal{M}$ by utilizing both the labeled data from $\mathcal{S}$ and the unlabeled datasets from all the clients $\mathcal{C}_m$ within a SSFL scheme, to achieve high accuracy on client data.

III. Proposed Solution

In this Section, we introduce SemiFDA, our solution for Semi-Supervised Federated Domain Adaptation. SemiFDA, depicted in Figure 1 and detailed in Algorithm 1, is designed to operate in a SSFL scenario where there is a single server $\mathcal{S}$ with unlimited memory and computational capabilities, and multiple resource-constrained clients $\mathcal{C}_1 \ldots, \mathcal{C}_M$ that continuously gather unlabeled data. We assume that the learning scheme develops along multiple communication rounds, in which the server and clients exchange models and additional parameters, but not data. In SemiFDA, the classification model, denoted as $\mathcal{M}$, consists of 2 parts: a feature extractor $\mathcal{E}$ which we design as an *encoder*, and a classification head $\mathcal{K}$. In the following, $\mathcal{E}(x, \theta)$ represents the output of the encoder $\mathcal{E}$ with parameters θ for the input x, while the output of the full classification model is denoted as $y = \mathcal{M}(x)$ and can be expressed as $\mathcal{M}(x, [\theta, \omega]) = \mathcal{K}(\mathcal{E}(x, \theta), \omega)$.

The rationale behind SemiFDA is to compensate for domain shift by training all the client encoders $\mathcal{E}(\cdot, \theta_m)$ so that the distribution of latent features extracted by $\mathcal{E}(\cdot, \theta_m)$ on the data of client $\mathcal{C}_m$ is aligned with the distribution of the features extracted by $\mathcal{E}(\cdot, \theta_S)$ from server data. This alignment allows the classification head $\mathcal{K}$, which is trained only on labeled server data, to accurately classify data on client $\mathcal{C}_m$ using the local encoder $\mathcal{E}(\cdot, \theta_m)$. In the SemiFDA framework, all encoders $\mathcal{E}(\cdot, \theta_m)$ are trained at clients during the client update step, as outlined in Section III-A. This training is driven by an unsupervised loss function that we specifically design to align the distribution of latent features from both clients and server encoders. Then, the aggregation phase described in Section III-B involves aggregating the client encoders $\mathcal{E}(\cdot, \theta_m)$ via Federated Averaging, while the parameters ω of $\mathcal{K}$ remain unchanged after the initial training at server.

A. Local Unsupervised Training

The first ingredient of SemiFDA is a fully unsupervised loss function that enables local training while compensating for domain shift. During our local unsupervised training, each client $\mathcal{C}_m$ focuses exclusively on the encoder $\mathcal{E}$, which is trained to align the distribution of each local latent space with the one of the server. To achieve this, we minimize the squared Frobenius norm of the difference between the covariance matrices of features extracted by the server encoder and those extracted over batches of client data. Specifically, we update $\mathcal{E}$ at $\mathcal{C}_m$ by solving the optimization problem:

$$\theta_m = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}_U(\theta, X_m, \Sigma_S)\,, \tag{1}$$

where the unsupervised loss function $\mathcal{L}_U$ is defined as

$$\mathcal{L}_U(\theta, X_m, \Sigma_S) = ||\Sigma_S - \text{cov}[\mathcal{E}(X_m, \theta)]||_F^2\,. \tag{2}$$

Here $\text{cov}[\cdot]$ denotes the operator computing the sample covariance, and Σ_S denotes the covariance matrix of the features extracted by $\mathcal{E}(\cdot, \theta_S)$ at server, which is exchanged between the server and clients only once after the model initialization.

Our loss $\mathcal{L}_U$ shares with CORAL [9] the idea of aligning covariance matrices of latent features to perform unsupervised domain adaptation. In [9], such alignment was achieved by defining roto-translation in the feature space of a SVM. Deep CORAL [10] adopts a similar principle to compensate for domain shift when training Deep CNN for classification. However, in our SSFL scenario, neither CORAL nor Deep CORAL can be directly applied, as they would either require to collect client data at the server, which clearly violates privacy, or to distribute the server data to clients, which is not feasible due to memory and communication constraints. Additionally, the proposed $\mathcal{L}_U$ represents the sole training loss at clients (and not a regularization term as in [10]), making our learning process at clients unsupervised.

B. Encoder-only Aggregation Scheme

The second ingredient of SemiFDA is an encoder-only aggregation scheme, which is essential to address domain shift in our SSFL scenario. We aggregate via FedAvg [7] all the locally updated encoder parameters $\{\theta_m^{(t)}\}_{m=1}^M$, which are sent by each client to the server, to obtain a unique global encoder of parameters $\theta_S^{(t)}$. In contrast, we never update the parameters ω_S of the classification head $\mathcal{K}$ and the covariance matrix Σ_S of latent features at server, after they have been initially trained with labeled data from the server. The aggregated global encoder is not retrained on server data, thus model updates are based only on unsupervised training at clients.

Algorithm 1 SemiFDA

1: **Server Executes:**
2: *//Centralized Model Initialization*
3: Train $\mathcal{M}$ on the training set (X_S, Y_S) and define encoder and classification head parameters $[\theta_S, \omega_S]$
4: $\Sigma_S = \text{cov}[\mathcal{E}(X_S, \theta_S)]$ *//Covariance Computation*
5: send$(\theta_S, \omega_S, \Sigma_S)$ to all clients $\{\mathcal{C}_m\}_{m=1}^M$
6: **Federated Learning Loop:**
7: **for** each communication round $t = 1 \dots T$
8: **for** each client $\mathcal{C}_m, m = 1 \dots M$ **in parallel do**
9: $\theta_m^{(t)} \leftarrow ClientUpdate(\mathcal{C}_m, \theta_S^{(t-1)}, \Sigma_S, X_m^{(t)})$
10: **end for**
11: $\theta_S^{(t)} \leftarrow \frac{1}{M} \sum_{m=1}^M \theta_m^{(t)}$ *//Encoders Aggregation*
12: send$(\theta_S^{(t)})$ to all clients $\{\mathcal{C}_m\}_{m=1}^M$
13: **end for**
14: **Inference:**
15: $\hat{y} = \mathcal{K}(\mathcal{E}(x, \theta_S^{(T)}), \omega_S)$
16: **ClientUpdate**$(\mathcal{C}_m, \theta_S, \Sigma_S, X_m)$:
17: $\theta_m = \theta_S$
18: $\mathcal{B} \leftarrow$ (split X_m into batches)
19: **for** each local epochs $l = 1, \dots, L$ **do**
20: **for** batch $B \in \mathcal{B}$ **do**
21: $\theta_m \leftarrow \theta_m - \eta \nabla \mathcal{L}_U(B, \theta_m, \Sigma_S)$
22: **end for**
23: **end for**
24: return θ_m to server.

Our intuition is that in a federated context affected by domain shift it is crucial to establish a fixed reference point using labeled server features, and gradually align with that reference through unsupervised iterations. In our case, the reference is defined by the latent representation of the server, characterized by the covariance matrix Σ_S and coupled with the classification head $\mathcal{K}$. The alignment is thus obtained through local unsupervised training and encoder-only aggregation. In particular, it is crucial to avoid retraining the entire model on the server's data at each iteration, and to maintain Σ_S and $\mathcal{K}$ fixed. In fact, any update would modify the distribution of features at server where $\mathcal{K}$ is trained, thus nullifying the benefits of local feature alignment. This could result in significant instability and fluctuations during training, as shown by our experiments.

C. SemiFDA

The overall scheme of SemiFDA is illustrated in steps in Figure 1 and the pseudocode is reported in Algorithm 1. Here, we illustrate each step in detail.

a) Step 0 - Model Initialization: $\mathcal{S}$ pre-trains in an end-to-end manner the encoder $\mathcal{E}(\cdot, \theta_S)$ and the classification head $\mathcal{K}(\cdot, \omega_S)$ by minimizing a classification loss over the labeled training set (X_S, Y_S). This training phase (line 3) defines the parameters of the encoder θ_S and the classification head ω_S.

b) Step 1 - Covariance Computation at server: $\mathcal{S}$ extracts the latent features corresponding to X_S using the initialized encoder $\mathcal{E}(\cdot, \theta_S)$, computes the covariance matrix $\Sigma_S = \text{cov}[\mathcal{E}(X_S, \theta_S)]$ and sends Σ_S to the clients (lines 4-5). The clients are thus able to perform inference on their own data using the full model $\mathcal{M} = \mathcal{K}(\mathcal{E}(x, \theta_S), \omega_S)$. After these initial steps, the *Client Update* and *Encoders Aggregation* (steps 2 and 3 of Figure 1) are iterated over T communication rounds, progressively updating the encoder parameters $\theta_S^{(t)}$.

c) Step 2 - Client Update: at communication round t, each client $\mathcal{C}_m$ locally updates its encoder $\mathcal{E}(\cdot, \theta_m)$ by minimizing the loss (2). Each client trains its encoder for a fixed number of epochs on all the data gathered during the t-th iteration, namely $X_m^{(t)}$, which is split in batches (line 18). The batches need to contain a sufficient number of samples to estimate the covariance matrices in the latent space of $\mathcal{E}(\cdot, \theta_{\Updownarrow})$, and are discarded after the parameters update. The parameters $\theta_m^{(t)}$ are updated (line 21) as follows:

$$\theta_m^{(t)} = \theta_m^{(t)} - \eta \nabla \mathcal{L}_U(B, \theta_m^{(t)}, \Sigma_S). \tag{3}$$

d) Step 3 - Encoders Aggregation: at the end of the communication round t, $\mathcal{S}$ receives the updated local encoders' parameters $\{\theta_m^{(t)}\}_{m=1}^M$ from the M clients and aggregates them via FedAvg as $\theta_S^{(t)} = \frac{1}{M} \sum_{m=1}^M \theta_m^{(t)}$. The resulting weights $\theta_S^{(t)}$ define the aggregated encoder $\mathcal{E}(\cdot, \theta_S^{(t)})$ (line 11), and are sent back to the clients for the next iteration (line 12).

e) Step 4 - Inference: the clients can perform on-device inference after any communication round t by combining the current aggregated encoder $\mathcal{E}(\cdot, \theta_S^{(t)})$ with the classification head $\mathcal{K}(\cdot, \omega_S)$. Similarly, at the end of the last communication round $t = T$, the encoder $\mathcal{E}(\cdot, \theta_S^{(T)})$ is combined with the fixed classification head $\mathcal{K}(\cdot, \omega_S)$ within a unique model $\mathcal{M}$ that is sent to the clients (line 15) for on-device inference. This model $\mathcal{M}$ is not specifically designed for a client $\mathcal{C}_m$, but rather a model incorporating local updates from all clients, as the encoder employed by $\mathcal{M}$ is not fine-tuned on $\mathcal{C}_m$.

IV. Experiments

A. Experimental Framework

Our experimental framework aims to evaluate SemiFDA and other SSFL works across different degrees of domain shift. We focus on the HAR task, which is inherently affected by the domain shift and by lack of labels on clients. Additionally, we examine the Digit Classification task to show that SemiFDA can be applied beyond just one specific use case. In HAR, we consider 3 publicly available datasets: USC-HAD, RealWorld, and MobiAct. We resample all data to a frequency of 50Hz and we split the recordings to segments of 64 samples. We consider only 4 activities these datasets have in common: *standing*, *walking*, *running*, and *jumping*. In Digit Classification, we consider 4 benchmark datasets: MNIST, MNIST-M, SVHN and USPS. For both tasks, we organize the datasets in different ways between the server and the clients, to study the following 4 scenarios:

a) No domain shift: in this case we consider i.i.d. data partitions among the server and the clients, without any form of domain shift. We explore this scenario using Digit Classification datasets, which consist of i.i.d. samples. We

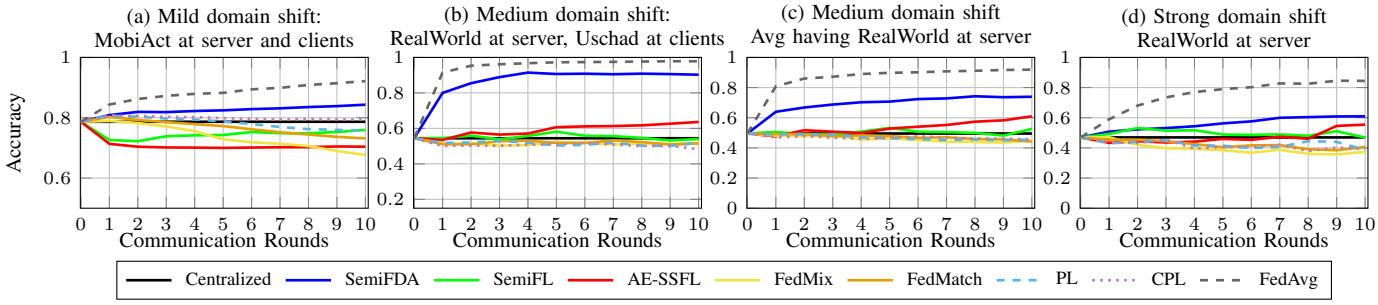

Fig. 2. Classification accuracies achieved by SemiFDA w.r.t. baselines and competitors in the mild (a), medium (b, c) and strong (d) domain shift. In (d) we have both MobiAct and USCHAD at clients (as for (c)), but the encoder parameters are aggregated exclusively among clients referring to the same dataset.

conduct 4 experiments, one for each digit dataset, where we randomly split the samples between the server and the clients. To create a realistic FL setting, we restrict the training set size at the server to prevent the initial classifier from saturating its performance. Specifically, for each experiment, we use only 250 labeled images for supervised training at the server.

b) Mild domain shift: this is a peculiar case of HAR (and other subject-specific tasks) due to *client heterogeneity*. We simulate this scenario by selecting a single HAR dataset at a time and splitting that between the server and the clients. This results in 3 distinct experiments, one for each HAR dataset. In each experiment, we randomly select 2 random subjects at the server, and we distribute the remaining subjects among the clients, ensuring that each client has data from distinct users. As in the previous case, to simulate a realistic FL environment, we prevent the Centralized model from saturating its performance. We did not use Digit datasets, even though we could have artificially designed splits to generate non-i.i.d. partitions on clients, e.g., by altering class proportions, as this would have not reflected a realistic form of domain shift.

c) Medium domain shift: we increase the level of domain shift by looking at pairs of datasets, selecting one at the server and distributing the other among the clients. In Digit Classification, we perform 3 different experiments, using the MNIST dataset on the server and one of the other datasets at the clients. In HAR, we test all the 6 combinations of dataset pairs, using the first at the server and dividing the second among the clients preserving subject-wise splits.

d) Strong domain shift: this scenario corresponds to facing a change of distributions both between the server and the clients and between the clients themselves. In Digit Classification, we consider all datasets collectively, using the MINST dataset on the server and dividing the data from the 3 other datasets among the clients, with each client getting data from just one dataset. In HAR, we perform 3 different experiments, each time using a different dataset on the server and distributing the other among the clients, by preserving dataset-wise and subject-wise splits.

We compare SemiFDA against various alternatives in the SSFL domain. To ensure a fair comparison, we use the same model architecture, hyperparameters and train-test splits as SemiFDA for all the methods. Our classification model is a standard CNN, comprising a feature extractor (an encoder), and a classification head. Specifically, the encoder has 4 Convolutional layers with ReLu activations, each followed by a Max Pooling layer. The classification head consists solely of Dense layers, each followed by Dropout. The bottleneck size is set to 16, which represents the dimension of the latent space, where we compute the covariance matrix. The model is initially trained until convergence (300 epochs for HAR, 30 epochs for Digit Classification) with an early stopping callback on the validation loss (validation split=0.2). In each local iteration, encoders are trained for 30 epochs, with a batch size of 64. In each experiment we consider 10 communication rounds and we set the participation rate of the clients to 100% for simplicity. We split the data of each client in *train* and *test* sets using a $80\% - 20\%$ split, and we adopt Adam optimizer with a learning rate of 1e-3. As *baselines*, we consider 1) the *Centralized model* (lower bound), which trains the classifier only with the server's labeled data without any update from clients; 2) the supervised *FedAvg* [7] (upper bound), which utilizes labeled data from clients. Since FedAvg depends on client supervision, it cannot be considered a competitor. We design a few *competitors* by combining centralized SSL techniques based on pseudo-labeling (PL) with Federated Averaging [7]: *i) Vanilla PL* ([1]+[7], denoted in tables and figures as PL), *ii) CPL* ([12]+[7], *iii) FedMix* ([8]+[7]), and *iv) FedMatch* ([2]+[7]). Other competitors are *v) AEs for SSFL* from [15], which is based on AutoEncoders, and *vi) SemiFL* [4], which represents the state-of-the-art in the SSFL scenario.

B. Results

To quantitatively assess the classification performance of SemiFDA, for every experiment we measure the *accuracy* on test data at clients at the end of the iterations. We compare the accuracy achieved by each method using the statistical procedure of [3], which employs a Friedman test followed by post-hoc tests. In all the tables of results, we highlight in bold the method achieving the best accuracy if it outperforms the second-best one with a statistical significance of 5%.

a) No domain shift: results for the Digit Classification task, reported in Table I, show that, in the absence of domain shift, SemiFL [4] outperforms all competitors, including our SemiFDA, confirming its state-of-the-art performance. This is not surprising, as SemiFDA is tailored to face domain shift and in i.i.d. settings the fact that classification head is not retrained during iterations is detrimental. Notably, if we simply retrain the classification head $\mathcal{K}$ on the server at each iteration (results

TABLE I
NO DOMAIN SHIFT IN DIGIT-CLASSIFICATION: SAME DATASET AT SERVER AND CLIENTS

Dataset	Centralized	PL	CPL	FedMatch	FedMix	AE-SSFL	SemiFL	SemiFDA	SemiFDA-retr	FedAvg
MNIST	.15 (.01)	.92 (.06)	.93 (.06)	.92 (.08)	.91 (.07)	.79 (.01)	.93 (.06)	.11 (.01)	.68 (.04)	.98 (.06)
MNIST-M	.12 (.01)	.55 (.07)	.59 (.08)	.51 (.06)	.54 (.09)	.18 (.02)	.58 (.07)	.10 (.01)	.20 (.02)	.89 (.11)
SVHN	.16 (.01)	.39 (.02)	.41 (.02)	.33 (.02)	.39 (.02)	.19 (.01)	.45 (.02)	.14 (.01)	.17 (.01)	.76 (.03)
USPS	.22 (.02)	.94 (.05)	.93 (.05)	.96 (.05)	.95 (.05)	.87 (.01)	.95 (.05)	.29 (.02)	.88 (.04)	.90 (.06)
Avg	.17 (.01)	.70 (.05)	.72 (.06)	.68 (.06)	.70 (.06)	.42 (.01)	.73 (.05)	.17 (.01)	.42 (.03)	.84 (.07)

TABLE II
MILD DOMAIN SHIFT IN HAR: SAME DATASET AT SERVER AND CLIENTS

Dataset	Centralized	PL	CPL	FedMatch	FedMix	AE-SSFL	SemiFL	SemiFDA	FedAvg
USCHAD	.88 (.02)	.81 (.02)	.90 (.02)	.66 (.02)	.59 (.04)	.93 (.01)	.95 (.01)	.92 (.01)	.96 (.01)
RealWorld	.73 (.02)	.71 (.04)	.69 (.04)	.71 (.04)	.59 (.04)	.81 (.01)	.76 (.02)	.85 (.01)	.90 (.01)
MobiAct	.79 (.02)	.76 (.03)	.79 (.03)	.73 (.03)	.68 (.04)	.70 (.03)	.76 (.03)	.84 (.02)	.92 (.01)
Avg	.80 (.02)	.76 (.03)	.79 (.03)	.70 (.03)	.62 (.04)	.81 (.02)	.82 (.02)	**.87 (.01)**	.93 (.01)

TABLE III
MEDIUM DOMAIN SHIFT IN DIGIT CLASSIFICATION AND HAR: DIFFERENT DATASET AT SERVER AND CLIENTS

Dataset at server	Dataset at clients	Centralized	PL	CPL	FedMatch	FedMix	AE-SSFL	SemiFL	SemiFDA	FedAvg
MNIST	MNIST-M	.22 (.01)	.13 (.01)	.13 (.01)	.13 (.01)	.13 (.01)	.27 (.01)	.19 (.01)	.43 (.01)	.89 (.02)
MNIST	SVHN	.14 (.01)	.16 (.01)	.16 (.01)	.15 (.01)	.17 (.01)	.08 (.01)	.14 (.01)	.12 (.01)	.79 (.04)
MNIST	USPS	.88 (.01)	.72 (.02)	.69 (.03)	.83 (.01)	.75 (.03)	.67 (.01)	.87 (.01)	.94 (.01)	.99 (.01)
Avg	Avg	.41 (.01)	.34 (.01)	.33 (.01)	.37 (.01)	.35 (.02)	.34 (.01)	.40 (.01)	.50 (.01)	.89 (.02)
USCHAD	RealWorld	.49 (.03)	.45 (.03)	.46 (.03)	.46 (.03)	.45 (.03)	.49 (.03)	.47 (.04)	**.62 (.04)**	.89 (.02)
USCHAD	MobiAct	.21 (.02)	.29 (.03)	.32 (.03)	.31 (.03)	.29 (.03)	.21 (.02)	.35 (.04)	**.50 (.04)**	.87 (.04)
RealWorld	USCHAD	.54 (.03)	.52 (.04)	.48 (.03)	.51 (.04)	.52 (.03)	.64 (.05)	.54 (.04)	**.90 (.02)**	.98 (.01)
RealWorld	MobiAct	.45 (.05)	.39 (.05)	.43 (.05)	.37 (.05)	.38 (.05)	.58 (.04)	.51 (.04)	.58 (.05)	.86 (.03)
MobiAct	USCHAD	.36 (.02)	.30 (.02)	.30 (.02)	.34 (.02)	.33 (.03)	**.66 (.03)**	.38 (.03)	**.67 (.07)**	.96 (.03)
MobiAct	RealWorld	.70 (.03)	.70 (.03)	.70 (.03)	.68 (.03)	.69 (.03)	.76 (.03)	.71 (.03)	.58 (.04)	.90 (.01)
Avg	Avg	.46 (.03)	.44 (.03)	.45 (.03)	.45 (.03)	.44 (.03)	.56 (.03)	.49 (.04)	**.64 (.05)**	.91 (.03)

reported as *SemiFDA-retr* in Table I), SemiFDA becomes effective in scenarios without domain shift as well.

b) Mild domain shift: results from the 3 HAR datasets, reported in Table II, show that SemiFDA outperforms on average both the Centralized model and all the competitors. Both [15] and SemiFL [4] can improve the Centralized model, albeit less than our SemiFDA, while other alternatives can even decrease its performance.

c) Medium domain shift: the results for Digit Classification are reported in Table III. On average, SemiFDA improves over the Centralized model and outperforms all the competitors in 2 out of 3 experiments. On the contrary, all PL solutions and [15], on average, decrease the performance of the Centralized model, resulting in a negative learning effect. This happens because these SSFL solutions are not designed to handle domain shift, which can lead to incorrect pseudo-labels. HAR results are reported in Table III. SemiFDA outperforms both the Centralized model and the other competitors in all the experiments, but one. Notably, SemiFDA significantly improves the Centralized model even when the server and client datasets differ significantly and the initial accuracy is very low (such as the case USCHAD-MobiAct). In contrast, methods based on PL or AEs fail to perform effectively in presence of domain shifts. As illustrated in Figure 2(b), SemiFDA increases the accuracy from 0.54 to 0.9 in the RealWorld-USCHAD case, achieving results comparable to the supervised FedAvg and outperforming all other competitors.

d) Strong domain shift: the results for Digit Classification, adopting MNIST at server, are reported in the last row of Table IV. SemiFDA outperforms both the Centralized model and all other competitors in all the cases, demonstrating it is always the best solution to cope with domain shift. The results for HAR are reported in Table IV, where again SemiFDA emerges as the top-performing solution on average, surpassing all competitors and the Centralized model. Moreover, both Table IV and Figure 2 indicate that current SSFL solutions may negatively impact the performance of the Centralized model. A comparison between Figure 2(c) and Figure 2(d) reveals there is a significant domain shift among clients that affects the accuracy of all methods, including SemiFDA. Figure 2(c) reports the *average accuracy* of the two *medium domain shift* experiments with RealWorld at the server. Figure 2(c) can be seen as the performance achieved when using the SSFL solutions in combination with an oracle that enables aggregating models trained on clients with data from the same HAR dataset. For all methods, this average accuracy is higher than the accuracy obtained in the *strong domain shift* case of Figure 2(d), confirming that this scenario poses an additional

TABLE IV

STRONG DOMAIN SHIFT IN HAR AND DIGIT CLASSIFICATION: IN HAR WE USE 1 DATASET AT SERVER, AND THE REMAINING 2 AT CLIENTS. THE ACCURACY IS COMPUTED OVER CLIENTS HAVING DATA FROM THE SAME DATASET AND AVERAGED OVER ALL CLIENTS (AVG). IN DIGIT CLASSIFICATION, WE USE MNIST ON THE SERVER AND THE OTHER 3 DATASETS (MNIST-M, SVHN AND USPS) AT CLIENTS.

Dataset at server	Centralized	PL	CPL	FedMatch	FedMix	AE-SSFL	SemiFL	SemiFDA	FedAvg
USCHAD	.31 (.02)	.37 (.02)	.35 (.03)	.35 (.03)	.33 (.03)	.28 (.01)	.48 (.04)	.53 (.06)	.87 (.04)
RealWorld	.47 (.03)	.39 (.04)	.39 (.04)	.41 (.04)	.37 (.03)	.55 (.06)	.47 (.04)	.61 (.04)	.84 (.04)
MobiAct	.56 (.03)	.54 (.02)	.54(.02)	.55 (.03)	.54 (.03)	**.72 (.05)**	.51 (.02)	.53 (.05)	.84 (.03)
Avg	.45 (.03)	.43 (.03)	.43 (.03)	.44 (.03)	.41 (.03)	.52 (.05)	.49 (.03)	**.56 (.05)**	.85 (.04)
MNIST	.41 (.01)	.37 (.01)	.36 (.01)	.41 (.01)	.37 (.01)	.35 (.01)	.41 (.01)	**.53 (.01)**	.63 (.01)

challenge for SSFL and necessitates improved aggregation strategies to address severe domain shift among clients.

C. Ablation Study

To demonstrate that all the components of SemiFDA are needed to effectively handle domain shift, we discuss our design choices by analyzing 3 alternative solutions. The first, *Non-Federated SemiFDA*, removes the aggregation step from SemiFDA, and can also be viewed as an unsupervised personalization method, where individual clients fine-tune their local encoders with their own unlabeled data. The second alternative, *SemiFDA Regularization*, modifies the method in [15] by adding our loss (2) as a regularization term when training the local AEs at the clients. The third alternative, *SemiFDA-retr*, involves iteratively fine-tuning $\mathcal{K}$ on the server, while freezing the aggregated encoder. In Figure 3, these variants are compared to SemiFDA in the *medium domain shift* scenario. While *Non-Federated SemiFDA* improves over the Centralized model, it still underperforms w.r.t. SemiFDA. Comparing the accuracy of *SemiFDA Regularization* against [15] in Figure 2(b), we observe that the regularization is beneficial, but suboptimal w.r.t. SemiFDA. This confirms that it is crucial to use the proposed loss as primary learning loss, and not as regularization term. Also the *SemiFDA-retr* variant is suboptimal w.r.t. SemiFDA in presence of domain shift, leading to oscillations and instability in the training process. The iterative fine-tuning brings the latent space back to the server's data distribution, without compensating for domain shift.

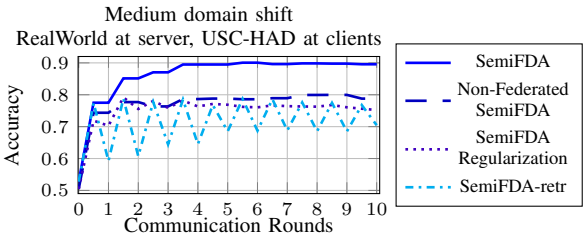

Fig. 3. Classification accuracies achieved by the 3 *ablation study* alternatives w.r.t. the original SemiFDA.

V. CONCLUSION AND FUTURE WORKS

This paper presents SemiFDA, a novel solution to handle domain shift in SSFL, which employs a custom unsupervised loss function on clients to align their feature distributions to those of the server, where the classification head is trained. We assess the performance of SemiFDA on HAR and Digit Classification by designing an experimental framework that mimic different levels of domain shift. Our experiments demonstrate that SemiFDA outperforms existing SSFL methods in several scenarios with domain shifts. Future works will focus on enhancing SemiFDA to better address substantial domain shifts among clients, e.g. by incorporating clustering techniques.

REFERENCES

[1] Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *IJCNN*. IEEE, 2020.

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *NeurIPS*, 32, 2019.

[3] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

[4] Enmao Diao, Jie Ding, and Vahid Tarokh. Semifl: Semi-supervised federated learning for unlabeled clients with alternate training. In *NeurIPS*, volume 35, 2022.

[5] Sannara Ek, Romain Rombourg, Francois Portet, and Philippe Lalanda. Federated self-supervised learning in heterogeneous settings: Limits of a baseline approach on har. In *PerCom Workshops*. IEEE, 2022.

[6] Wonyong Jeong, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. Federated semi-supervised learning with inter-client consistency & disjoint learning. *arXiv preprint arXiv:2006.12097*, 2020.

[7] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[8] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *NeurIPS*, 33, 2020.

[9] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation, 2015.

[10] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV Workshops*, 2016.

[11] Xu Yang, Yanan Gu, Kun Wei, and Cheng Deng. Exploring safety supervision for continual test-time domain adaptation. In *IJCAI*, 2023.

[12] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *NeurIPS*, 34, 2021.

[13] Zhengming Zhang, Yaoqing Yang, Zhewei Yao, Yujun Yan, Joseph E Gonzalez, Kannan Ramchandran, and Michael W Mahoney. Improving semi-supervised federated learning by reducing the gradient diversity of models. In *ICBD*. IEEE, 2021.

[14] Chen Zhao, Zhipeng Gao, Qian Wang, Zijia Mo, and Xinlei Yu. Fedgan: A federated semi-supervised learning from non-iid data. In *International Conference on Wireless Algorithms, Systems, and Applications*, 2022.

[15] Yuchen Zhao, Hanyang Liu, Honglin Li, Payam Barnaghi, and Hamed Haddadi. Semi-supervised federated learning for activity recognition. *arXiv preprint arXiv:2011.00851*, 2020.