Scalable Collaborative Learning via Representation Sharing

Frédéric Berdoz
EPFL
frederic.berdoz@epfl.ch

Abhishek Singh
MIT
abhi24@mit.edu

Martin Jaggi
EPFL
martin.jaggi@epfl.ch

Ramesh Raskar
MIT
raskar@mit.edu

Abstract

Decentralized machine learning has become a key conundrum for multi-party artificial intelligence. Existing algorithms usually rely on the release of model parameters to spread the knowledge across users. This can raise several issues, particularly in terms of communication if the models are large. Additionally, participants in such frameworks cannot freely choose their model architecture as they must coincide to collaborate. In this work, we present a novel approach for decentralized machine learning, where the clients collaborate via online knowledge distillation using a contrastive loss (contrastive w.r.t. the labels). The goal is to ensure that the participants learn similar features on similar classes without sharing their input data nor their model parameters. To do so, each client releases averaged last hidden layer activations of similar labels to a central server that only acts as a relay (i.e., is not involved in the training or aggregation of the models). Then, the clients download these last layer activations (feature representations) of the ensemble of users and distill their knowledge in their personal model using a contrastive objective. For cross-device applications (i.e., small local datasets and limited computational capacity), this approach increases the utility of the models compared to independent learning, is communication efficient and is scalable with the number of clients. We prove theoretically that our framework is well-posed, and we benchmark its performance against standard collaborative learning algorithms on various datasets using different model architectures.

1 Introduction

Motivated by concerns such as data privacy, large scale training and others, Machine Learning (ML) research has seen a rise in different types of collaborative ML techniques. Collaborative ML is typically characterized by an orchestrator algorithm that enables training ML model(s) over data from multiple owners without requiring them to share their sensitive data with untrusted parties. Some of the well known algorithms include Federated Learning (FL) [24], Split Learning (SL) [9] and Swarm Learning [35]. While the majority of the works in collaborative ML rely upon a centralized coordinator, in this work, we design a new decentralized learning framework where the server plays a secondary role. In fact, its only purpose is to compute global averages for a secondary objective, which could also be approximated with sufficient accuracy using a peer-to-peer network (under mild connectivity constraints on the communication graph). Our main idea is to share learned feature representations of each class among users and to use these representations cleverly during local training (using a contrastive objective that is fully compatible with a peer-to-peer network). From a

2022 Trustworthy and Socially Responsible Machine Learning (TSRML 2022) co-located with NeurIPS 2022.

theoretical point of view, the objective of the scheme is to maximize the mutual information between these representations. Since the clients can choose which features are aggregated and shared, our framework enables the clients to assign different privacy levels to different samples. Our decentralized approach also ensures that the overall system remains asynchronous and functions as expected even if all but two clients are offline in the whole system. Finally, our framework makes it convenient to account for model heterogeneity and model personalization, since every user can select a subset of peers based on their goals of generalization and personalization. While some of these advantages have been introduced in recent FL based schemes, our framework allows natural integration of such several ideas.

2 Related Work

Federated Learning FL is considered to be the first formal framework for collaborative learning. In their initial paper, McMahan et al. [24] introduce a new algorithm called `FedAvg`, in which each client performs several optimization steps on their local private dataset before sending the updated model back to the server for aggregation using weight averaging. While this approach alleviates the communication cost of the baseline collaborative optimization algorithm `FedSGD`, it also decreases the personalization capacity of the global model due to the naive model averaging, especially in heterogeneous environments. Several algorithms have been proposed to address these limitations, in particular `FedProx` [20], `FedPer` [1], `FedMa` [33], `FedDist` [27], `FedNova` [34], `Scaffold` [15] and `VRL-SGD` [22]. Concerning the server update, Reddi et al. [26] introduce federated versions of existing adaptive optimization algorithms like `Adagrad`, `Adam` and `Yogi`, and Michieli and Ozay [25] present `FedProto`, where an attention mechanism is used for clever aggregations. The attention coefficients are computed using *prototypes* (i.e., per class averages of last hidden layer activations, a.k.a. feature representations). While our framework also uses such prototypes, it is conceptually very different as we use them directly in the local objective function (and not in the aggregation). Although all these algorithms usually improve the convergence rate, they suffer from the same constraints as `FedAvg`, i.e., homogeneous model architecture for every client, high communication overhead and *non-tunable* collaboration, all potential barriers for participation.

Fully Decentralized Learning The use of a central server in traditional FL constitutes a single point of failure and can also become a bottleneck when the number of clients grows, as shown by Lian et al. [21]. To alleviate these issues, Vanhaesebrouck et al. [31] formalize a new framework where each client participates in the learning task via a peer-to-peer network using gossip algorithms [28, 7]. In this configuration, there is no global solution and each client has its own personalized model, which enables both personalization and generalization. On the other hand, it creates other challenges about convergence, practical implementation and privacy [14]. Moreover, as in FL, the entire model must be released at every communication round, which can constitute a barrier for participation for the same reasons.

Collaborative Learning via Knowledge Distillation A growing body of literature has recently investigated ways of using online knowledge distillation (KD) [3, 11] for collaborative learning in order to alleviate the need of sharing model updates (FL) or individual smashed data (SL). Jeong et al. [13] present Federated Knowledge Distillation (FD), where each client uploads its mean (per class) logits to a central server, who aggregates and broadcasts them back. These soft labels are then used as the teacher output for the KD loss during local training. A closely related idea is to compute the mean logits on a common public dataset [19, 12, 4], but we argue that selecting this dataset can induce bias and is not always feasible, since additional trust is needed for its selection, and sufficient relevant data might be lacking. Besides FD, KD can enable collaborative learning in various ways: In an attempt to decrease the communication cost, Wu et al. [36] introduce `FedKD`, in which each client trains a (large) teacher network on their private data and transfer locally its knowledge to a smaller student model, which is then used in a standard FL algorithm. On the other hand, Lin et al. [23] and Chen and Chao [5] use KD on an unlabeled synthetic or public dataset to make the FL aggregation algorithm more robust. Our approach differs significantly from these schemes, as it does not rely on traditional FL algorithms.

3 Methods

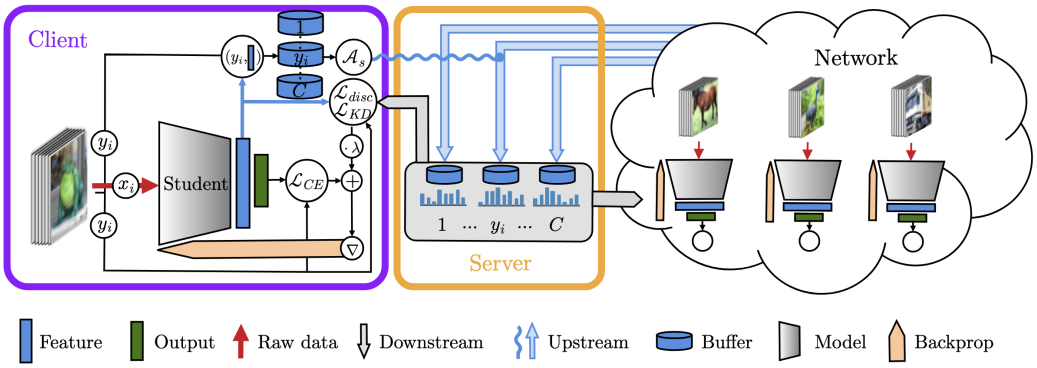


Figure 1: The proposed framework: clients exchange averaged ($\mathcal{A}_s$) feature representations for each class and use these representations in their own local training, but keep their raw data private (on device). The ensemble of participants (network) constitutes the teacher, and each single participant is a student. The server acts mainly as a relay as it does not take part in the training or aggregation of the models. At each communication round, the student downloads a subset of representations and uploads some of its own.

Preamble and Motivation Consider any classification task on d-dimensional raw inputs with C distinct classes and a set of N participating users $\{u\}_{u=1}^{N}$, each with a local private dataset $\mathcal{D}_u$ and a model f_u:

$$\mathcal{D}_u := \{(\mathbf{x}_i, y_i) \overset{iid}{\sim} p_u\}_{i=1}^{n_i}, \qquad p_u(\mathbf{x}, y) := p_{\mathbf{X},Y|U=u}(\mathbf{x}, y), \qquad f_u = \tau_u \circ \phi_u,$$

where $p_{\mathbf{X},Y,U}$ represents the joint probability of choosing a user U and drawing a sample $(\mathbf{X}, Y)$ from its distribution, and τ_u, ϕ_u represent the linear classifier and neural network (up to last hidden layer) of user u, respectively (with potentially different architectures across users). More precisely, let d' be the output dimension of ϕ_u and $\mathbf{w}_u = \{\boldsymbol{\theta}_u, \mathbf{W}_u, \mathbf{b}_u\}$ be model weights of user u. We have:

$$\phi_u : \mathbb{R}^d \to \mathbb{R}^{d'} \qquad \tau_u : \mathbb{R}^{d'} \to \mathbb{R}^C$$
$$\mathbf{x} \mapsto \mathbf{x}' := \phi_u(\mathbf{x}; \boldsymbol{\theta}_u) \qquad \mathbf{x}' \mapsto \mathbf{z} := \tau_u(\mathbf{x}'; \mathbf{W}_u, \mathbf{b}_u) = \mathbf{W}_u\mathbf{x}' + \mathbf{b}_u$$

where $\mathbf{x}, \mathbf{x}'$ and $\mathbf{z}$ are the raw input, the feature representation and the logits, respectively. We motivate our approach as follows. Assuming no sharing of raw data $\mathbf{x}$ (for privacy concerns), any collaborative learning framework falls in one of two buckets: weight sharing or activation sharing. Sharing weights (i.g., FL) comes with strong constraints (communication, model architecture, etc.) and might not always be suitable. Concerning activation sharing (i.g., SL), it can be done at any layer (hidden or output). However, since activations are usually strongly correlated to the raw input [32], sharing single sample activations can raise privacy concerns. Instead, sharing averaged activations can easily be met with differential privacy guarantees (see privacy in **??**). Due to the high non-linearity of neural networks, sharing averaged activations only makes sense at the output layer or last hidden layer (since the classifier τ_u is linear). Sharing only averaged outputs (averaged over samples of the same class) like in FD has been shown to have limited success as the quantity of shared information is restricted to the dimension of the output C, and we argue in this paper that sharing the feature representations (i.e., outputs of the last hidden layer, also averaged over samples of the same class) is more flexible and leads to better results. In this light, our objective is to collaboratively learn the best feature representation for each class, using contrastive (w.r.t. classes) representation learning [29] and feature-based knowledge distillation [8]. In other words, we want to learn collaboratively (i.e., only once) the structure of the feature representation space so that each client does not need to find it on its own with its limited amount of data and/or computational capacity.

Contrastive Objective We now introduce a contrastive objective function for private online knowledge distillation (i.e., when users synchronously learn personalized models without sharing their raw

input data and by collaborating via online distillation). This new objective function and its derivation are partly inspired by the offline contrastive loss presented in Tian et al. [29] and van den Oord et al. [30], with a few important differences. In the offline non-private setting, both the teacher and student models ϕ_t and ϕ_s have access to the same dataset $\{x_i\}_{i=1}^n$. In that scenario, the representations $\phi_s(x_i)$ and $\phi_t(x_j)$ are *pulled apart* if $i \neq j$ and are *pulled together* if $i = j$. However, in the private setting, ϕ_s does not have access to the raw input data that was used to train the teacher model, but has its own private dataset. At a given communication round and from the perspective of user u, we define ϕ_u as the student model and ϕ_U as the teacher model, where $U \sim p_U$ is a user selected at random. Consider the following procedure: u samples $Y \sim p_{Y|U=u}$ from its own data distribution and then samples either jointly (i.e., from the same observation of Y) or independently (from two independent observations of Y) the two random vectors $\mathbf{\Phi}_s$ and $\mathbf{\Phi}_t$ defined as follows:

$$\mathbf{X} \sim p_{\mathbf{X}|Y,U=u} \qquad \mathbf{\Phi}_s := \phi_u(\mathbf{X}), \tag{1}$$

$$U \sim p_U, \quad (\mathbf{X}_1, ..., \mathbf{X}_{n_{avg}}) \overset{iid}{\sim} p_{\mathbf{X}|Y,U}, \qquad \mathbf{\Phi}_t := \frac{1}{n_{avg}} \sum_{i=1}^{n_{avg}} \phi_U(\mathbf{X}_i). \tag{2}$$

The parameter n_{avg} defines over how many samples we take the average, which in turn defines the concentration of the distribution of $\mathbf{\Phi}_t$. From the (student) perspective of client u and in the spirit of collaboration, the goal is to maximize the mutual information $\mathcal{I}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)$. Still from the perspective of u, let $p_{s,t}$, p_s and p_t be the joint and marginal distributions of $\mathbf{\Phi}_s$ and $\mathbf{\Phi}_t$, respectively, and let I be a Bernoulli random variable indicating if a tuple $(\mathbf{\Phi}_s, \mathbf{\Phi}_t)$ has been drawn from the joint distribution $p_{s,t}$ or from the product of marginals $p_s p_t$. Finally, let $q(\mathbf{s}, \mathbf{t}, i)$ be the joint distribution of $(\mathbf{\Phi}_s, \mathbf{\Phi}_t, I)$ such that $q(\mathbf{s}, \mathbf{t}|i = 1) = p_{s,t}(\mathbf{s}, \mathbf{t})$ and $q(\mathbf{s}, \mathbf{t}|i = 0) = p_s(\mathbf{s})p_t(\mathbf{t})$ and suppose that the prior $q(i)$ satisfy $q(i = 1) = \frac{1}{K+1}$ and $q(i = 0) = \frac{K}{K+1}$, i.e., for each sample from the distribution $p_{s,t}$, we draw K samples from the distribution $p_s p_t$. We can show the following bound.

Theorem 1. *Using the above notation, let $h(i, \mathbf{s}, \mathbf{t})$ be any estimate of $q(i|\mathbf{s}, \mathbf{t})$ with Bernoulli parameter $\hat{h}(\mathbf{s}, \mathbf{t})$, and let $\mathcal{L}_{disc}(h, \phi_u)$ be defined as follows:*

$$\mathcal{L}_{disc}(h, \phi_u) := -\mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[\log \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] - K \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_s p_t} \left[\log(1 - \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)) \right]. \tag{3}$$

The mutual information $\mathcal{I}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)$ can be bounded as

$$\mathcal{I}(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \geq \log(K) - \mathcal{L}_{disc}(h, \phi_u), \tag{4}$$

with equality iff $h = q$ (better estimates lead to tighter bounds). The proof is joined in the supplementary material.

Hence, by minimizing $\mathcal{L}_{disc}$ in Eq. (4), we optimize a lower bound on the mutual information $\mathcal{I}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)$. Taking advantage of the classifier τ_u, a natural choice for h is the discriminator h_u with Bernoulli parameter

$$\hat{h}_u(\mathbf{s}, \mathbf{t}; \mathbf{W}_u, \mathbf{b}_u) = \langle \text{softmax}(\tau_u(\mathbf{s})), \text{softmax}(\tau_u(\mathbf{t})) \rangle. \tag{5}$$

With this choice, $\hat{h}_u(\mathbf{s}, \mathbf{t})$ can be interpreted as the estimated probability that the features $\mathbf{s}$ and $\mathbf{t}$ come from the same class. Note that in their work, Tian et al. [29] train an external discriminator (i.e., that is not defined using the model classifier τ_u).

Final Objective Intuitively, from the perspective of u, minimizing $\mathcal{L}_{disc}$ ensures that its classifier τ_u can distinguish if two feature representations, one local and one from another user, come from the same class. To improve the algorithm convergence and to ensure that τ_u can classify the feature representation $\mathbf{\Phi}_t$ of another client (similar as in Invariant Risk Minimization [2]), we also introduce a classical feature-based KD term $\mathcal{L}_{KD}$. This term minimizes the L_2 distance between the local and global feature representations of a same class (we define the global representation of class c as the expected value $\mathbb{E}_{(X,U) \sim p_{X,U|Y=c}}[\phi_U(\mathbf{X})]$). An important distinction between $\mathcal{L}_{KD}$ and $\mathcal{L}_{disc}$ is that the first one uses an *inter*-client averaged representation, whereas the second one uses an *intra*-client averaged representation. Combining $\mathcal{L}_{KD}$ and $\mathcal{L}_{disc}$ with the standard cross-entropy loss $\mathcal{L}_{CE}(\tau_u, \phi_u)$ using the meta parameters λ_{KD} and λ_{disc}, the final optimization problem of u becomes:

$$\text{Find} \quad \boldsymbol{\theta}_u^\star, \mathbf{W}_u^\star, \mathbf{b}_u^\star = \underset{\boldsymbol{\theta}_u, \mathbf{W}_u, \mathbf{b}_u}{\text{argmin}} \; \mathcal{L}_{CE}(\tau_u, \phi_u) + \lambda_{KD} \mathcal{L}_{KD}(\phi_u) + \lambda_{disc} \mathcal{L}_{disc}(\hat{h}_u, \phi_u). \tag{6}$$

Communication In terms of communication, the uplink and downlink volumes are of order $\mathcal{O}((M_\uparrow + 1)Cd')$ and $\mathcal{O}(N(M_\downarrow + 1)Cd')$ per round, where $M_\uparrow$ and $M_\downarrow$ represent the number of $\mathbf{\Phi}_t$ realizations (per class) that are uploaded and downloaded by the clients, respectively (these parameters can be tuned to match the communication capacity of the network). For comparison, these volumes are of order $\mathcal{O}(D)$ and $\mathcal{O}(ND)$ for FL (with D the model size) and $\mathcal{O}(nd')$ and $\mathcal{O}(Nnd')$ for SL. Because in most scenarios $D \gg n \gg d' \gg C$ and since $M_\uparrow$ and $M_\downarrow$ are tunable, we observe that our framework is communication efficient compared to FL and SL.

Relaxation to peer-to-peer We emphasize here that our contrastive objective $\mathcal{L}_{disc}$ is fully compatible with a peer-to-peer configuration, since the server only acts as a relay for the observations of $\mathbf{\Phi}_t$. For the traditional feature-based KD objective $\mathcal{L}_{KD}$, we use one global representation per class (i.e., one for the whole network), which in theory can only be computed by a central entity. One way to alleviate this could be to use the average of all the observations of $\mathbf{\Phi}_t$ that were downloaded by user u as a proxy for the global feature representations. However, to focus solely on the effectiveness of the proposed objectives rather than the topology of the network, we only present experiments in which a central entity computes the global representations.

4 Experiments

Datasets, models and training We run several experiments with the MNIST [6], Fashion-MNIST [37] and CIFAR10 [17] datasets. For MNIST, we use a simple convolutional neural network (CNN) similar to LeNet5 ($\approx 30K$ parameters) [18] and for Fashion-MNIST and CIFAR10, we use ResNet9 ($\approx 2,4M$ parameters) and ResNet18 ($\approx 11.3M$ parameters) architectures [10], respectively. For the dimension of feature representation space, we set $d' = 84$ for LeNet5, $d' = 128$ for ResNet9 and $d' = 256$ for ResNet18. In order to simulate a scenario where the data is sparse, we only select a fraction of the train dataset (1200 samples for MNIST, 6000 for Fashion-MNIST and 10000 for CIFAR10) that we split uniformly at random across $N \in \{2, 5, 10\}$ users. For the validation, we use the entire test dataset for each task (10000 samples). In order to have fair comparisons, we train all the models for the same number of communication rounds, and we stop the training as soon as framework has reasonably converged ($r = 100$ for MNIST/LeNet5, $r = 20$ for Fashion-MNIST/ResNet9 and CIFAR10/ResNet18). We compare our framework with centralized learning (or CL, i.e., with $N = 1$ and $\lambda_{KD} = \lambda_{disc} = 0$), independent learning (or IL, i.e., with $\lambda_{KD} = \lambda_{disc} = 0$), federated learning using `FedAvg` (FL) and federated knowledge distillation (FD). We use the default learning rate $\eta = 10^{-3}$ for all experiments and we perform 1 local epoch of training per communication round. For CL, IL, FD and our framework, we use the `Adam` stochastic optimization algorithm [16]. Finally, supported by Fig. 2, we set $\lambda_{KD} = 10$ and $\lambda_{disc} = 1$ in our final objective (Eq. (6)).

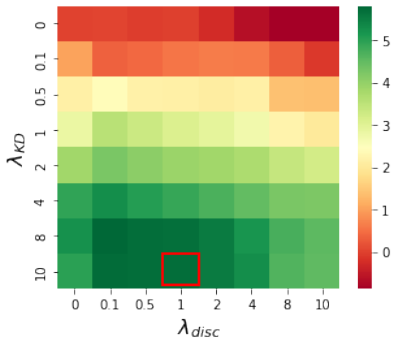
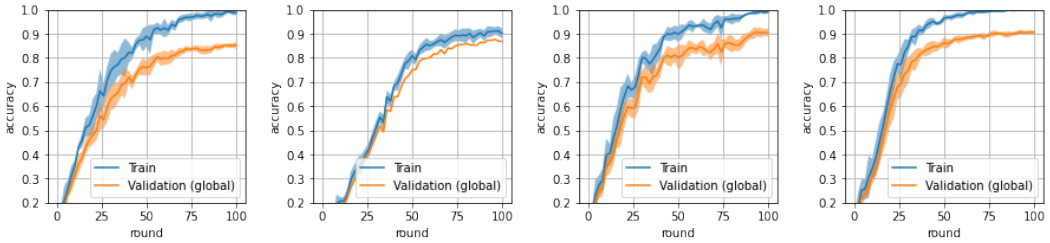


Figure 2: Ablation study for λ_{KD} and λ_{disc}. Average test accuracy improvement [%] w.r.t. IL (upper left corner) when different combinations of λ_{KD} and λ_{disc} are used (MNIST/LeNet5 experiment with 5 users and 100 communication rounds). The red square indicates the value of λ_{KD} and λ_{disc} used in all our experiments.

Network emulation For our contrastive objective $\mathcal{L}_{disc}$, we use $n_{avg} = 10$ (i.e., for every class, each user selects n_{avg} samples of that class, computes and averages their feature representations, uploads them to the server and downloads the representations of another user chosen at random). For the feature-based KD objective $\mathcal{L}_{KD}$, each client average the feature representation of all the samples of a same class and uploads these averaged representations to the server, which in turn averages them to obtain one global representation per class. For simplicity, we use $M_\uparrow = M_\downarrow = 1$ (clients upload and download one observation of $\mathbf{\Phi}_t$ for each class).

Performance As seen in Table 1, our framework outperforms every other framework for when a small model is used (MNIST/LeNet5), especially when the number of clients grows, which is typically the kind of configuration that would be relevant for a cross-device application). In that



(a) IL (b) FL (c) FD (d) Ours

Figure 3: Comparison of the train and test accuracy during 100 communication rounds of IL, FL, FD and our framework on the MNIST dataset with LeNet5 architectures and $N = 5$ users. The shaded areas represent $\pm$ the standard deviation of the metric across clients. For the validation, each model is tested using the entire test dataset.

Table 1: Average test accuracy over clients $[\%]$ of the different frameworks after r communication rounds when the same amount of data is divided uniformly at random between N users. We use 1200 training samples for MNIST, 6000 for Fashion-MNIST and 10000 for CIFAR10. The validation is done using the full test dataset (10000 samples for each task).

	MNIST ($r = 100$)			Fashion-MNIST ($r = 20$)			CIFAR10 ($r = 20$)		
CL	94.00			87.77			66.15		
	$N = 2$	$N = 5$	$N = 10$	$N = 2$	$N = 5$	$N = 10$	$N = 2$	$N = 5$	$N = 10$
FL	92.64	86.79	70.06	89.79	89.28	88.21	67.99	59.18	51.05
IL	91.46	85.26	72.86	86.04	83.61	80.52	59.85	46.46	38.51
FD	94.45	90.55	77.90	87.17	83.32	79.44	56.75	44.91	31.43
Ours	94.19	90.63	82.07	87.91	84.44	80.77	63.49	47.28	37.78

setup, FL performs particularly poorly as it struggles to find a low-capacity model that matches the data distribution of each client. This is particularly visible on Fig. 3b. Although in that configuration, FD shows similar performance for small number of clients ($N = 2, 5$), it still exhibits a lower rate of convergence (compare Fig. 3c and Fig. 3d). Our framework even outperforms centralized learning (CL) when $N = 2$, suggesting that the added objectives can also be seen as regularizers. For the Fashion-MNIST dataset, our framework shows significant improvement over IL and FD (i.e., frameworks where there are no global model), but is not able to compete with FL anymore, which in that case even outperforms CL. However, the comparison between FL and our framework is unfair for large models due to the amounts of shared information. Similar conclusions can be drawn for the CIFAR10 experiments. However, in that case, even IL outperforms our framework for $N = 10$ after $r = 20$ rounds. Indeed, since our objective function is highly complex (the global minimum depends on the data of other peers), the algorithm struggles to converge to the optimal model when the number of parameters is very large, which is a clear limitation that needs to be addressed.

5 Conclusion

We introduce a new collaborative learning algorithm that enables tunable collaboration in cross-device applications and whose uplink and downlink communication does not scale with the model size (as in FL) or the dataset size (as in SL). We prove that our objective is well posed from the point of view of collaboration, as it maximizes a lower bound on the mutual information between the feature representations of different users across the network. Then, we show empirically that it is particularly relevant in setups where the number of clients is large and when each of them have limited computational resources.

References

[1] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *CoRR*, abs/1912.00818, 2019. URL http://arxiv.org/abs/1912.00818.

[2] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2019.

[3] Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 535–541, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933395. doi: 10.1145/1150402.1150464. URL https://doi.org/10.1145/1150402.1150464.

[4] Hongyan Chang, Virat Shejwalkar, Reza Shokri, and Amir Houmansadr. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer, 2019. URL https://arxiv.org/abs/1912.11279.

[5] Hong-You Chen and Wei-Lun Chao. Feddistill: Making bayesian model ensemble applicable to federated learning. *CoRR*, abs/2009.01974, 2020. URL https://arxiv.org/abs/2009.01974.

[6] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[7] Alexandros G. Dimakis, Soummya Kar, José M. F. Moura, Michael G. Rabbat, and Anna Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010. doi: 10.1109/JPROC.2010.2052531.

[8] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[9] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[10] 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.

[11] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[12] 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. *CoRR*, abs/2008.06180, 2020. URL https://arxiv.org/abs/2008.06180.

[13] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *CoRR*, abs/1811.11479, 2018. URL http://arxiv.org/abs/1811.11479.

[14] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[15] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL http://www.cs.toronto.edu/~kriz/cifar.html.

[18] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, 12 1989. ISSN 0899-7667. doi: 10.1162/neco.1989.1.4.541. URL https://doi.org/10.1162/neco.1989.1.4.541.

[19] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *CoRR*, abs/1910.03581, 2019. URL http://arxiv.org/abs/1910.03581.

[20] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[21] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30, 2017.

[22] Xianfeng Liang, Shuheng Shen, Jingchang Liu, Zhen Pan, Enhong Chen, and Yifei Cheng. Variance reduced local SGD with lower communication complexity. *CoRR*, abs/1912.12844, 2019. URL http://arxiv.org/abs/1912.12844.

[23] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[24] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[25] Umberto Michieli and Mete Ozay. Prototype guided federated learning of visual feature representations. *CoRR*, abs/2105.08982, 2021. URL https://arxiv.org/abs/2105.08982.

[26] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization. *CoRR*, abs/2003.00295, 2020. URL https://arxiv.org/abs/2003.00295.

[27] EK Sannara, François Portet, Philippe Lalanda, and VEGA German. A federated learning aggregation algorithm for pervasive computing: Evaluation and comparison. In *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE, 2021.

[28] Devavrat Shah. Gossip algorithms. *Found. Trends Netw.*, 3(1):1–125, jan 2009. ISSN 1554-057X. doi: 10.1561/1300000014. URL https://doi.org/10.1561/1300000014.

[29] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkgpBJrtvS.

[30] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. URL http://arxiv.org/abs/1807.03748.

[31] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. *CoRR*, abs/1610.05202, 2016. URL http://arxiv.org/abs/1610.05202.

[32] Praneeth Vepakomma, Abhishek Singh, Otkrist Gupta, and Ramesh Raskar. Nopeek: Information leakage reduction to share activations in distributed deep learning. In *2020 International Conference on Data Mining Workshops (ICDMW)*, pages 933–942. IEEE, 2020.

[33] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris S. Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *CoRR*, abs/2002.06440, 2020. URL https://arxiv.org/abs/2002.06440.

[34] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

[35] Stefanie Warnat-Herresthal, Hartmut Schultze, Krishnaprasad Lingadahalli Shastry, Sathyanarayanan Manamohan, Saikat Mukherjee, Vishesh Garg, Ravi Sarveswara, Kristian Händler, Peter Pickkers, N Ahmad Aziz, et al. Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594 (7862):265–270, 2021.

[36] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation". *Nat Commun*, 13(1):2032, apr 2022.

[37] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747.

A Appendix

A.1 Notation

- N: Number of participating users/clients/peers/data owners.
- d: Raw input data dimension (e.g., $d = 3072$ for 32×32 RBG images).
- d': Latent feature (or feature representation) space dimensionality (i.e., width of last hidden layer).
- C: Number of class.
- $(\mathbf{x}, y) \in \mathbb{R}^d \times \{0, ..., C-1\}$: Labeled data sample.
- $p_{\mathbf{X},Y,U} : \mathbb{R}^d \times \{1, ..., C\} \times \{1, ..., U\} \to \mathbb{R}_+$: Joint probability density function across users.
- $p_u(\mathbf{x}, y) := p_{\mathbf{X},Y|U=u}(\mathbf{x}, y)$: Data distribution of user u.
- $\mathcal{D}_u := \{(\mathbf{x}_i, y_i) \overset{iid}{\sim} p_u\}_{i=1}^{n_i}$: Dataset of user u.
- $\mathbf{w}_u := \{\boldsymbol{\theta}_u, \boldsymbol{W}_u, \mathbf{b}_u\}$ with $\boldsymbol{\theta}_u \in \Theta_u, \boldsymbol{W}_u \in \mathbb{R}^{C \times d'}, \mathbf{b}_u \in \mathbb{R}^C$: Parameters of the neural network of u, with Θ_u the achievable model parameters for user u.
- $\phi_u : \mathbb{R}^d \to \mathbb{R}^{d'},\ \mathbf{x} \mapsto \mathbf{x}' = \phi(\mathbf{x}; \boldsymbol{\theta}_u)$: Neural network of u (or similar parameterized function) that maps a raw input into a latent feature space.
- $\tau_u : \mathbb{R}^{d'} \to \mathbb{R}^C,\ \mathbf{x}' \mapsto \mathbf{z} = \tau(\mathbf{x}'; \mathbf{W}_u, \mathbf{b}_u) := \mathbf{W}_u\mathbf{x}' + \mathbf{b}_u$: Linear classifier of u.
- $\lambda_{KD}, \lambda_{disc}, n_{avg}, M_\uparrow, M_\downarrow$: Hyperparameters.
- η: Learning rate.
- $\ell_{CE}, \ell_{KD}, \ell_{disc}$: Cross-entropy, feature-based KD and discriminator loss functions, respectively.
- $\mathcal{L}_{CE}, \mathcal{L}_{KD}, \mathcal{L}_{disc}$: Expected value (over the data) of ℓ_{CE}, ℓ_{KD} and ℓ_{disc}, respectively.
- L_{CE}, L_{KD}, L_{disc}: Mini-batch estimates of $\mathcal{L}_{CE}, \mathcal{L}_{KD}$ and $\mathcal{L}_{disc}$, respectively.
- $\boldsymbol{\Phi}_s, \boldsymbol{\Phi}_t$: Random vectors (feature representations) of the student (user u) and the teacher (random user U).
- $p_{s,t}, p_s, p_t$: Joint and marginal distributions of $\boldsymbol{\Phi}_s$ and $\boldsymbol{\Phi}_t$, respectively.
- q: Joint distribution of $\boldsymbol{\Phi}_s, \boldsymbol{\Phi}_t$ and I, where I is a binary random variable indicating if $\boldsymbol{\Phi}_s, \boldsymbol{\Phi}_t$ has been drawn from $p_{s,t}$ ($I = 1$) or $p_s p_t$ ($I = 0$).
- h: Binary discriminator with Bernoulli parameter $\hat{h}$ (i.e., learnable estimate of $q_{I|\boldsymbol{\Phi}_s,\boldsymbol{\Phi}_t}$).
- $\mathbf{s}, \mathbf{t}$: Observation/realization of $\boldsymbol{\Phi}_s$ and $\boldsymbol{\Phi}_t$, respectively.
- $\bar{\mathbf{t}}^c, \mathbf{t}^c$: Global (i.e., using all the samples across users) and local (i.e., using n_{avg} samples) average feature representations of class c, respectively.

A.2 Proof of Theorem 1

Recall that $q(\mathbf{s}, \mathbf{t}, i)$ is the joint distribution of $(\mathbf{\Phi}_s, \mathbf{\Phi}_t, I)$ such that $q(\mathbf{s}, \mathbf{t}|i = 1) = p_{s,t}(\mathbf{s}, \mathbf{t})$ and $q(\mathbf{s}, \mathbf{t}|i = 0) = p_s(\mathbf{s})p_t(\mathbf{t})$ and suppose that the prior $q(i)$ satisfy $q(i = 1) = \frac{1}{K+1}$ and $q(i = 0) = \frac{K}{K+1}$, i.e., for each sample from the distribution $p_{s,t}$, we draw K samples from the distribution $p_s p_t$. We have:

$$\mathcal{I}(\mathbf{\Phi}_s, \mathbf{\Phi}_t) = \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[-\log \frac{p_s(\mathbf{\Phi}_s)p_t(\mathbf{\Phi}_t)}{p_{s,t}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)} \right] \tag{7}$$

$$= \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[-\log \left(K \frac{p_s(\mathbf{\Phi}_s)p_t(\mathbf{\Phi}_t)}{p_{s,t}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)} \right) \right] + \log(K) \tag{8}$$

$$\geq \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[-\log \left(1 + K \frac{p_s(\mathbf{\Phi}_s)p_t(\mathbf{\Phi}_t)}{p_{s,t}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)} \right) \right] + \log(K) \tag{9}$$

$$= \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[\log q(i = 1|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] + \log(K) \tag{10}$$

where the last equality is obtained using the Bayes' rule on the posterior $q(i = 1|\mathbf{\Phi}_s, \mathbf{\Phi}_t)$:

$$q(i = 1|\mathbf{\Phi}_s, \mathbf{\Phi}_t) = \frac{q(\mathbf{\Phi}_s, \mathbf{\Phi}_t|i = 1)q(i = 1)}{q(\mathbf{\Phi}_s, \mathbf{\Phi}_t|i = 0)q(i = 0) + q(\mathbf{\Phi}_s, \mathbf{\Phi}_t|i = 1)q(i = 1)} \tag{11}$$

$$= \frac{p_{s,t}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)}{K p_s(\mathbf{\Phi}_s)p_t(\mathbf{\Phi}_t) + p_{s,t}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)} \tag{12}$$

$$= \left(1 + K \frac{p_s(\mathbf{\Phi}_s)p_t(\mathbf{\Phi}_t)}{p_{s,t}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)} \right)^{-1}. \tag{13}$$

Hence, by optimizing $\mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[\log q(i = 1|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right]$ with respect to the model parameters $\boldsymbol{\theta}_u$ of the student, we optimize a lower bound on the mutual information between $\mathbf{\Phi}_s, \mathbf{\Phi}_t$. By noting that $\log q(i = 0|\mathbf{\Phi}_s, \mathbf{\Phi}_s) \leq 0$, we can further bound the expectation term in (10) as follows:

$$\mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[\log q(i = 1|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \geq \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q|I=1} \left[\log q(i = 1|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right]$$
$$+ K \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q|I=0} \left[\log q(i = 0|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \tag{14}$$

$$= (K + 1) \sum_i q(i) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q|I=i} \left[\log q(i|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \tag{15}$$

$$= (K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t, I) \sim q} \left[\log q(I|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \tag{16}$$

$$= (K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q} \left[\mathbb{E}_{I \sim q|\mathbf{\Phi}_s, \mathbf{\Phi}_t} \left[\log q(I|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \right] \tag{17}$$

However, similar to Tian et al. [29], the Bernoulli distribution $q(i|\mathbf{\Phi}_s, \mathbf{\Phi}_t)$ is unknown and must therefore be approximated by training a discriminator $h : \{0, 1\} \times \mathbb{R}^{d'} \times \mathbb{R}^{d'} \to [0, 1]$. Using Gibbs' inequality, we obtain

$$-\mathbb{E}_{I \sim q|\mathbf{\Phi}_s, \mathbf{\Phi}_t} \left[\log q(I|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \leq -\mathbb{E}_{I \sim q|\mathbf{\Phi}_s, \mathbf{\Phi}_t} \left[\log h(I, \mathbf{\Phi}_s, \mathbf{\Phi}_t) \right], \tag{18}$$

where the right-hand term is the expected negative log-likelihood loss of the discriminator for a particular set $(\mathbf{\Phi}_s, \mathbf{\Phi}_t)$. Hence, Eq. (17) is proportional to minus the expected loss of the discriminator. Let $\hat{h}(\mathbf{s}, \mathbf{t}) \in [0, 1]$ denote the Bernoulli parameter of h given the data $(\mathbf{s}, \mathbf{t})$ (i.e., $h(i, \mathbf{s}, \mathbf{t}) = \hat{h}(\mathbf{s}, \mathbf{t})^i (1 - \hat{h}(\mathbf{s}, \mathbf{t}))^{1-i}$), we define our learning objective for the discriminator as follows:

$$-\mathcal{L}_{disc}(h, \phi_u) := (K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q} \left[\mathbb{E}_{I \sim q|\mathbf{\Phi}_s, \mathbf{\Phi}_t} \left[\log h(I, \mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \right] \tag{19}$$

$$= (K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t, I) \sim q} \left[\log h(I|\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] \tag{20}$$

$$= (K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t, I) \sim q} \left[\log \left(\hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)^I (1 - \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t))^{(1-I)} \right) \right] \tag{21}$$

$$= +(K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q|I=1} \left[\log \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] q(i = 1)$$
$$(K + 1) \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim q|I=0} \left[\log \left(1 - \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right) \right] q(i = 0) \tag{22}$$

$$= \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_{s,t}} \left[\log \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \right] + K \mathbb{E}_{(\mathbf{\Phi}_s, \mathbf{\Phi}_t) \sim p_s p_t} \left[\log(1 - \hat{h}(\mathbf{\Phi}_s, \mathbf{\Phi}_t)) \right], \tag{23}$$

which concludes the derivation.

A.3 Algorithms

With the standard assumption that the datasets are composed of IID samples drawn from their corresponding distributions, the expected losses $\mathcal{L}_{CE}, \mathcal{L}_{KD}$ and $\mathcal{L}_{disc}$ can be approximated by their unbiased mini-batch estimators L_{CE}, L_{KD} and L_{disc}, respectively. For one given sample, the loss functions are given by $\ell_{CE}(\mathbf{z}, c) := -\log\left(\text{softmax}(\mathbf{z})\right)_c$, $\ell_{KD}(\mathbf{x}', \mathbf{x}'') := \|\mathbf{x}' - \mathbf{x}''\|^2$ and finally

$$\begin{aligned}\ell_{disc} : \{0,1\} \times \mathbb{R}^{d'} \times \mathbb{R}^{d'} &\to \mathbb{R}_+ \\ (i, \mathbf{s}, \mathbf{t}) &\mapsto -i \log\left(\hat{h}(\mathbf{s}, \mathbf{t})\right) - (1-i)\log\left(1 - \hat{h}(\mathbf{s}, \mathbf{t})\right). \end{aligned} \tag{24}$$

Finally, for the sampling procedure of the contrastive objective, we use the most intuitive scheme: for every training sample $(\mathbf{x}_i, y_i)$ with feature representation $\mathbf{s}_i = \phi_u(\mathbf{x}_i)$, we use one observation of $\mathbf{\Phi}_t$ sampled using y_i (i.e., $I = 1$), and one observation of $\mathbf{\Phi}_t$ sampled using each $c \neq y_i$ (i.e., $I = 0$). Thus, we have $K = C - 1$. With these considerations, a detailed description of our collaborative learning framework can be found in the supplementary material.

Algorithm 1: GLOBALUPDATE

Input: Server S, N users with local datasets $\{\mathcal{D}_u\}_{u=1}^N$.

S initializes randomly $\{\bar{\mathbf{t}}^c\}_{c=1}^C$ and random observations $\{\{\mathbf{t}_m^c\}_{c=1}^C\}_{m=1}^{N \cdot M_\uparrow}$
Each client u initializes $\mathbf{w}_u^0 := \{\boldsymbol{\theta}_u^0, \mathbf{W}_u^0, \mathbf{b}_u^0\}$
$r \leftarrow 0$
while *training:*
  $r \leftarrow r + 1$
  for *each client* u*:*
    u downloads $\{\bar{\mathbf{t}}^c\}_{c=1}^C$ and $M_\downarrow$ random observations $\{\{\mathbf{t}_m^c\}_{c=1}^C\}_{m=1}^{M_\downarrow}$
    $\mathbf{w}_u^r \leftarrow$ LOCALUPDATE$(\mathcal{D}_u, \mathbf{w}_u^{r-1}, \{\bar{\mathbf{t}}^c\}_{c=1}^C, \{\{\mathbf{t}_m^c\}_{c=1}^C\}_{m=1}^{M_\downarrow})$
    u computes and uploads its local averaged representations $\{\bar{\mathbf{t}}_u^c\}_{c=1}^C$
    u computes (using n_{avg}) and uploads $M_\uparrow$ observations $\{\{\mathbf{t}_{u,m}^c\}_{c=1}^C\}_{m=1}^{M_\downarrow}$
  S aggregates $\{\{\bar{\mathbf{t}}_u^c\}_{c=1}^C\}_{u=1}^N$ to obtain $\{\bar{\mathbf{t}}^c\}_{c=1}^C$
  S stores (and shuffles) $\{\{\mathbf{t}_{u,m}^c\}_{c=1}^C\}_{m=1}^{M_\downarrow}$ in their corresponding class buffers
return $\{\mathbf{w}_u^r\}_{u=1}^N$

Algorithm 2: LOCALUPDATE

Input: Local dataset $\mathcal{D}_u$, model parameters $\mathbf{w}_u = \{\boldsymbol{\theta}_u, \mathbf{W}_u, \mathbf{b}_u\}$, global features $\{\bar{\mathbf{t}}^c\}_{c=1}^C$ and observations $\{\{\mathbf{t}_m^c\}_{c=1}^C\}_{m=1}^{M_\downarrow}$, number of local training rounds E, averaging parameter n_avg.

Buffer initialization (per class): $\{\hat{\mathbf{\Phi}}_u^c \leftarrow \mathbf{0}\}_{c=1}^C$
for $e \in [1, ..., E]$*:*
  for *mini-batch* $\mathcal{B} \in \mathcal{D}_u$*:*
    $L_{KD}, L_{CE}, L_{disc} \leftarrow 0$
    for $(\mathbf{x}_i, y_i) \in \mathcal{B}$*:*
      $\mathbf{s}_i \leftarrow \phi_u(\mathbf{x}_i)$
      $L_{CE} \leftarrow L_{CE} + \frac{1}{|\mathcal{B}|}\ell_{CE}(\tau_u(\mathbf{s}_i), y_i)$
      $L_{KD} \leftarrow L_{KD} + \frac{1}{|\mathcal{B}|}\ell_{KD}(\mathbf{s}_i, \bar{\mathbf{t}}^{y_i})$
      Sample $m \sim$ UNIFORM$(1, ..., M_\downarrow)$
      $L_{disc} \leftarrow L_{disc} + \frac{1}{|\mathcal{B}|}\left(\ell_{disc}(1, \mathbf{s}_i, \mathbf{t}_m^{y_i}) + \sum_{c \neq y_i} \ell_{disc}(0, \mathbf{s}_i, \mathbf{t}_m^c)\right)$
    $L \leftarrow L_{CE} + \lambda_{KD} L_{KD} + \lambda_{disc} L_{disc}$
    $\mathbf{w}_u \leftarrow \mathbf{w}_u - \eta \nabla_{\mathbf{w}_u} L$
return $\mathbf{w}_u$