Federated Learning with Noisy Labels

VASILEIOS TSOUVALAS, Eindhoven University of Technology, The Netherlands
AAQIB SAEED, Philips Research, The Netherlands
TANIR OZCELEBI, Eindhoven University of Technology, The Netherlands
NIRVANA MERATNIA, Eindhoven University of Technology, The Netherlands

Federated Learning (FL) is a distributed machine learning paradigm that enables learning models from decentralized private datasets, where the labeling effort is entrusted to the clients. While most existing FL approaches assume high-quality labels are readily available on users' devices; in reality, label noise can naturally occur in FL and follows a non-i.i.d. distribution among clients. Due to the "*non-iid-ness*" challenges, existing state-of-the-art centralized approaches exhibit unsatisfactory performance, while previous FL studies rely on data exchange or repeated server-side aid to improve model's performance. Here, we propose `FedLN`, a framework to deal with label noise across different FL training stages; namely, FL initialization, on-device model training, and server model aggregation. Specifically, `FedLN` computes per-client noise-level estimation in a single federated round and improves the models' performance by correcting (or limiting the effect of) noisy samples. Extensive experiments on various publicly available vision and audio datasets demonstrate a 24% improvement on average compared to other existing methods for a label noise level of 70%. We further validate the efficiency of `FedLN` in human-annotated real-world noisy datasets and report a 9% increase on average in models' recognition rate, highlighting that `FedLN` can be useful for improving FL services provided to everyday users.

Additional Key Words and Phrases: federated learning, noisy labels, label correction, deep learning, knowledge distillation

1 INTRODUCTION

Recent advances in smartphones, wearables, and the Internet of Things devices have led to continuous generation of massive amounts of data from both embedded sensors and logs obtained from users' interactions with various applications. The ubiquity of these contemporary devices and the exponential growth of the data produced by them present a significant opportunity to tackle critical problems for a wide range of application domains, such as healthcare, well-being, and manufacturing. Traditionally, machine learning (ML) approaches require the data from distributed devices to be stored or aggregated in a centralized cloud-based server before being further processed for solving a specific problem. However, the rapidly increasing volume of generated data, in combination with the high communication costs and possible bandwidth limitations make the accumulation of data in a centralized cloud-based server unfeasible [16]. Additionally, such centralized data aggregation schemes could also be restricted by privacy issues and regulations, e.g., General Data Protection Regulation (GDPR). Due to these factors and the advancement of computational and storage capabilities of distributed devices, it is appealing to leave the data on devices and utilize local computational resources for learning models.

To this end, The field of federated learning (FL) [13] is concerned with distributed training of machine learning models on decentralized data residing on personal devices like smartphones and wearables. The key idea behind FL is to bring the computation closer to where the data resides to extensively harness data locality. Specifically, in a federated learning regime, updates to the deep neural network models (e.g., their learnable parameters) are performed entirely on-device and communicated to the central server, which aggregates these updates from all participating devices to produce a unified global model. Unlike the standard centralized way of learning models, the salient differentiating factor of FL is that the data never leaves the user's device, which is an appealing property for privacy-sensitive data. This distributed ML strategy has been applied to a wide range of tasks in

Authors' addresses: Vasileios Tsouvalas, Eindhoven University of Technology, Eindhoven, The Netherlands, v.tsouvalas@tue.nl; Aaqib Saeed, Philips Research, Eindhoven, The Netherlands, aaqib.saeed@philips.com; Tanir Ozcelebi, Eindhoven University of Technology, Eindhoven, The Netherlands, t.ozcelebi@tue.nl; Nirvana Meratnia, Eindhoven University of Technology, Eindhoven, The Netherlands, n.meratnia@tue.nl.

recent years with great success [8, 15, 34, 42]. Nevertheless, a common limitation of existing supervised FL approaches is the implicit assumption that on-device data are perfectly annotated [36].

In reality, data samples in FL either cannot be labeled readily or labels quality cannot be guaranteed to the same extent as datasets that are collected and annotated in a centralized environment. Under federated setting, the data annotation can be performed either through user interaction or an automated approach via programmatic labeling functions, such as those used for keyboard query suggestions [42]. However, these labeling techniques do not provide guarantees on label correctness and may result in noisy or incorrect labels being assigned to the data samples. This is also the most case for automatic labeling systems, such as [26], where "*weak*" labels are constructed, which are inherently noisy. Furthermore, even if users themselves provide labels, there is no way to verify the quality of those labels due to private nature of on-device datasets. This is in contrast to centralized settings, where labels' validation can be performed to some extent by visually inspecting the data (even though this process remains unscalable for large-scale datasets). Therefore, in FL, the presence of mislabeled data samples, referred to as label noise or noisy labels, can naturally occur, while there is no straightforward way to perform label correction.

The problem of designing robust learning schemes in the presence of label noise has received noticeable attention in recent years [30], with various algorithms being proposed to train models using noisy labeled examples [1, 22, 23]. The majority of these centralized approaches are either based on filtering noisy samples or mitigating their effect through regularization techniques. However, due to locality and non-i.i.d. nature of data in FL, such centralized learning schemes are either infeasible or can negatively affect the performance of existing methods under federated setting. Apart from the "*non-iid-ness*" of data in FL [16], label noise can also follow a non-i.i.d. distribution, since noisy labels originates from a variety of client-dependent sources, such as the discrepancy between clients' labeling systems or the difference in clients' expertise and willingness to label data correctly. While label noise can naturally occur in FL and is a major practical problem when training FL models, it has received hardly any attention from the research community. The proposed solutions so far focus on detecting noisy labels through communicating private data [41].

To the best of our knowledge, this work is the first attempt at learning models in a federated setting for a variety of classification tasks under noisy labels, without relying on exchanging clients' sensitive information or introducing additional communication costs. We tackle the problem of label noise under federated setting in two stages: firstly, a per-client noise level estimation is been computed either at the initialization phase of FL (i.e., beginning of the training) or after performing a few federated training rounds with no overhead for the clients devices. This is motivated by the fact that in FL clients' noise profiles can drastically differ, while a well-annotated dataset can be present on a few clients, which we refer to as "*clean*" clients. Therefore, detecting those "*clean*" clients and the amount of label noise across the remaining ones can be beneficial for the training process and will ultimately result in a highly generalizable global model. Once a per-client noise level is established, in the second phase, we exploit this knowledge to efficiently train deep neural networks to improve performance of a given task, while correcting (or limiting the effect of) noisy labeled samples. Concisely, the main contributions of our work are as follows:

- We propose a framework, called `FedLN` (Federated Learning under Label Noise), to address the generalizability issues introduced when training federated models using noisy labeled data.
- We design simple, yet, effective approaches to accurately estimate a per-client label noise level and identify clients with clean or relatively high-quality labels.
- We devise various mechanisms to correct noisy labeled instances on a per-client basis, thus mitigating the need of user interaction for high-quality label acquisition. Further, these mechanisms can be equally useful when automated labeling techniques are used for label extraction that may otherwise result in noisy labels.

- We demonstrate through extensive evaluation that our framework is highly useful for learning generalizable models under a variety of federated and label noise settings on diverse public datasets from both vision and audio domains, namely CIFAR-10 [14], FashionMNIST [38], PathMNIST [40], and SpeechCommands [35].
- We show that FedLN with approximately 40% of labeled data correctly annotated, on average can improve recognition rate by 24% across all datasets compared to the fully-supervised federated model. Evaluating FedLN on real-world human annotated noisy datasets, namely CIFAR-10N and CIFAR-100N [36], we showcase an increase on recognition rate by 9% compared to the standard FL process.

2 BACKGROUND

In this section, we provide a brief overview of the statistical properties of label noise, the procedure of training a neural network with noisy labels, and federated learning to provide a foundation for our approach for training deep learning models with noisy labels in a federated setting.

2.1 Label Noise

Label noise refers to the misalignment between a ground truth label y^* and an observed label y in a given dataset. Specifically, in a C-way classification problem, where C is the number of label categories, label noise can be considered as a class-conditional label flipping process $f(\cdot)$, which projects $y^* \rightarrow y$, in a way that every label in class $j \in C$ may be independently mislabeled as class $i \in C$ with probability $p(y{=}i|y^*{=}j)$, written in a shorthand notation as $p(y|y^*)$. Hence, in our work we assume that the occurrences of label noise are data-independent, i.e., $p(y|y^*,x) = p(y|y^*)$, similar to [5]. From the definition of label noise function $f(y^*,C)$, a $C \times C$ noise distribution matrix denoted by $Q_{y|y^*}$ can be defined, where each column corresponds to the probability distribution for an input instance with ground truth label $y^*{=}i$ to be assigned to label j.

Given the definitions above, we can characterize label noise through two statistic parameters, i.e., noise level (denoted by n_l), and noise sparsity (denoted by n_s). Inspired by [23], we provide a formal definition for each of these parameters, as follows:

Definition 2.1 (Noise Level). Noise level (n_l) quantifies the amount of label noise present in a given dataset. It is defined as the reverse probability of the sum along the diagonal of $Q_{y|y^*}$, denoted as $n_s = 1 - \text{diag}\left(Q_{y|y^*}\right)$. Intuitively, noise level of zero corresponds to a "*clean*" dataset, where all observed labels match their ground truth labels, while a noise level of one can be considered as completely noisy dataset.

Definition 2.2 (Noise Sparsity). Noise sparsity (n_s) quantifies the shape of the label noise present in a dataset. It is defined by the fraction of zeros in the off-diagonals of noise distribution matrix $Q_{y|y^*}$. Thus, high noise sparsity values indicate a non-uniformity of label noise, which is common in most real-world datasets. For example, a high-sparsity noise can indicate a confusion between classes that are perceived to be related by humans, i.e., mislabeling of a cat as tiger or a lion, rather than a cat as a bird or dog. Alternatively, zero level of noise sparsity corresponds to completely random noise, where all instances belonging to one class can be confused with any other class. The special case of "*class-flipping*" can be constructed for $n_s = 1$, where instances that belong to a pair of two classes are confused. In this case, the noise probabilities between any pair of classes are equal, i.e., $p(y{=}i|y^*{=}j) = p(y{=}j|y^*{=}i)$.

Federated Label Noise. Considering the traditional centralized learning, noise distribution can be characterized by a single noise distribution matrix $Q_{y|y^*}$. However, in FL, where data is fragmented across multiple clients, distinct noise distributions among clients have to be considered, as noise resembles a non-i.i.d distribution in FL and is closely related to the clients' characteristics, i.e., user's expertise or preferences. Subsequently, in FL, noise distribution matrices among clients can differ significantly, i.e., $Q^i_{y|y^*} \neq Q^j_{y|y^*}$ with i,j indicating any pair

of clients. These naturally occurring differences in noise distributions among clients (i.e., clients' noise profiles) can introduce additional challenges for the FL process, especially during models' aggregation step.

2.2 Learning from Noisy Labels

The goal of a C-way supervised learning task is to learn a function that maps an input instance x to a corresponding ground truth label $y_i^* \, \epsilon \, \{1, \cdots, C\}$. Let $p_\theta \, (y \mid x)$ be a neural network that is parameterized by weights θ that predicts softmax outputs y for a given input x. In a typical classification problem, the model is provided with a training dataset $\mathcal{D} = \left\{\left(x_i, y_i^*\right)\right\}_{i=1}^{N}$, and aims to minimize the following objective function by learning the model's parameters θ:

$$\mathcal{L}_\theta \, (\mathcal{D}) = l \, (y^*, p_\theta \, (y|x)) \, , \tag{1}$$

where $\mathcal{L}_\theta \, (\mathcal{D})$ is the minimization function for supervised learning on $\mathcal{D}$, $l \, (\cdot)$ denotes a certain loss, and p_θ is the neural network that is parameterized by weights θ.

In real-world scenarios, in which data labels are often noisy, the neural network $p_\theta \, (y|x)$ is trained by noisy labels y, instead of the actual ground-truth labels y^*. Thus, the model is now provided with a noisy training dataset $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^{N_n}$. Similar to Equation 1, the objective is to minimize $\mathcal{L}_\theta$ on $\mathcal{D}_n$ by learning the model's parameters θ, as follows

$$\mathcal{L}_\theta \, (\mathcal{D}_n) = l \, (y, p_\theta \, (y|x)) \, , \tag{2}$$

Here, noisy labeled instances interference with the loss minimization process, since the computed loss is over the noisy dataset $\mathcal{D}_n$. Specifically, the neural network p_θ can easily memorize noisy labels and consequently degenerate network's generalization on unseen data.

3 METHODOLOGY

In this section, we present our federated learning framework, FedLN, for federated learning models under the presence of label noise. Firstly, we provide a formal definition of the underlying problem. Then, we discuss our proposed techniques for determining a per-client noise level estimation in detail. Finally, we provide a thorough description of our developed approached for handling and mitigating the impact of noisy labels during training of deep models in federated setting.

3.1 Problem Formulation

We focus on the problem of federated learning with noisy labels, where clients' data samples are often mislabeled either due to missing expertise of their users, users' mistakes, or error in the automated procedure for label inference. While noise can be present in input-space of data (e.g., background noise in audio or high brightness in the image), in this work, we solely focus on "*noise*" to be present in the label-space (i.e., categorize in case of classification problems). Under federated setting, label noise follows a highly non-i.i.d. distribution, which results in different noise profiles among clients. These distinct label noise profiles are introducing risks of overfitting to noisy data and suffering from parameters divergence during the server-side model aggregation. Additionally, a few clients that hold data with high-quality labels, i.e., $n_l \leqslant \varepsilon$ with $\varepsilon \rightarrow 0$, may be present in the FL process. With FedLN, we aim to eliminate the effect of on-device mislabeled samples in the training process and to improve the performance of FL models, alleviating the common assumption that clients hold well-annotated data. In this way, the generalization of a federated model can be substantially decoupled from the quality of labels utilized during training. Such separation means that models' performance does not depend on users' annotation effort or correctness of automatic labeling techniques. This is a highly desirable feature for most real-world applications, e.g., audio keyword spotting.

Formally in FL setting, we have a set of M clients, each holding a training set $\mathcal{D}^m$. Subsequently, each client's dataset, $\mathcal{D}^m$, can be divided into a correctly labeled set $\mathcal{D}_c^m = \left\{\left(x_i, y_i^*\right)\right\}_{i=1}^{N_c^m}$ and a noisy labeled set $\mathcal{D}_n^m = \{(x_i, y_i)\}_{i=1}^{N_n^m}$, where $N^m = N_c^m + N_n^m$ is the total number of data samples stored on the m^{th} client and $N = \sum_{i=0}^{M} N^m$ is the total number of samples present during training. The label noise level present in the m^{th} client's data is given by $n_l^m = \frac{N_c^m}{N^m}$. We aim to learn a global unified model G without clients sharing any of their local data ($\mathcal{D}^m$), while minimizing the effect of noisy label set $\mathcal{D}_n^m$ on the training process. Specifically, the objective function we aim to minimize is the following:

$$\min_{\theta} \mathcal{L}_\theta = \sum_{m=1}^{M} \gamma_m \mathcal{L}_m(\theta), \text{ where } \mathcal{L}_m(\theta) = \mathcal{L}_\theta(\Phi(\mathcal{D}^m)) = \mathcal{L}_\theta\left(I\left(\mathcal{D}_c^m\right) + \Psi\left(\mathcal{D}_n^m\right)\right), \tag{3}$$

where $\mathcal{L}(\mathcal{D}^m)$ is the supervised loss term of the m^{th} client, $\Phi(\cdot)$ is a correction mechanism aiming at reducing the impact of noisy samples of the m^{th} client on the training procedure by either masking or correcting the label y of $\mathcal{D}_n^m$, and $I(\cdot)$ is the identity function. With γ_m, we denote the relative impact of the m^{th} client on the generation of the global model G. For FedAvg [21] algorithm, parameter γ_m is equal to the ratio of client's local data N_m over all training samples, i.e., $\gamma_m = \frac{N_m}{N}$.

3.2 Label Noise Estimation in Federated Setting

In federated setting, the label noise is directly affected by discrepancies on clients' labeling systems or their users expertize. These discrepancies result in varying label noise profiles on a client basis, where a few '*clean*' clients may exist, holding high-quality labels. Here, we address the problem of noisy labels by firstly estimating a per-client label noise level and correctly identifying any '*clean*' client. To this end, we propose two methods for determining a per-client noise level estimation, i.e., (i) an embeddings-based discovery, in which noise is computed from '*noise-tolerant*' embeddings, and (ii) a model's confidence based approach, in which noise is estimated using a scoring function based on models outputs (or logits). By establishing a per-client noise level, we can then efficiently train deep neural networks to improve the performance on a given task, limiting the effect of noisy samples on model's generalizability.

Embedding-based Discovery of Noisy Labels. In supervised learning, a deep model can easily memorize corrupted labels, which can lead to poor generalization when exposed to unseen data. On the contrary, learning from embeddings extracted from a self-supervised pre-trained model can help avoid such problems. Specifically, due to the fact that pre-training is performed with large-scale unlabeled data, the quality of embeddings remain unaffected by the presence of noisy labels [45].

For this purpose, we utilize a self-supervised pre-trained model as a feature extractor $g(\cdot)$ to produce embeddings e_i for every input instance $x_i \in \mathcal{D}_m$. With the generation of embeddings, we can then utilize a k-Nearest Neighbor (kNN) approach to identify noisy samples, which corresponds to outliers in the embeddings space (i.e., data points belonging to the same neighborhood with different labels). Specifically, for the neighbourhood of k points surrounding e_i, we assign a new label using a majority voting mechanism, with random tie-breaking, as:

$$y_i^{vote} = \underset{j \in [k]}{arg\,max}\ \widehat{y_i}[j] = \max \sum_{j=1}^{k} \left\{ y_j \in \mathcal{D}_k : \left|\min\left\{\left\|g(x_i) - g(x_l)\right\|, \forall x_l \in \mathcal{D}\right\}\right| < k \right\}, \tag{4}$$

where y_j is the predicted kNN label for embedding vector e_j, extracted using the feature extractor $g(\cdot)$ from an input instance x_j. One should note that local neighbourhood surrounding embedding e_i, denoted as $\mathcal{D}_k$ in Equation 4, is formulated by computing the euclidean distance of e_i with all other embedding's vectors in $\mathcal{D}$.

Finally, to detect if label y_i is noisy, we check whether y_i^{vote} matches the original label y_i [45]. Extending this process across all participating clients, $\mathcal{D}^m$, we can compute a per-client noise level estimation , by:

$$n_l^m = \frac{\sum_{i=0}^{N_m} y_i^{vote} \neq y_i}{N_m}, \tag{5}$$

where N_m is the number of samples available in the m^{th} client's local dataset, $\mathcal{D}^m$. It is important to note that the aforementioned procedure is performed locally on each client during the initialization phase of FL (i.e., beginning of the training). Thus, noise level estimation via embeddings does not introduce additional computation or communication costs to the FL training process.

Model Confidence as a Proxy for Label Noise. To provide an alternative to utilizing an external module (i.e.,a pre-trained model) for the detection of noisy labeled clients, we propose the use of a scoring based method that can directly be applied to the outputs (or logits) of the trained neural network. The intuition behind our approach is to utilize a scoring function to rank input instances, such that a low score indicates a high probability of having a noisy label. Therefore, the critical components required to facilitate this approach include a *scoring function*, which ranks samples based on model's predictions confidence, and a *threshold* to distinguish between low-score (noisy labeled) and high-score (correctly labeled) data.

To this end, we utilize energy score [19] as our *scoring function*. While energy score is leveraged for detecting out-of-domain samples [19], in this work, we propose to use it as a proxy to identify clients with noisy data. In contrast to the softmax score, energy score has proven to be less susceptible to modern's neural networks overconfidence issues [6, 19]. Accordingly, we compute the energy score for an input instance x as:

$$s = \mathcal{E}(x, p_\theta) = -T \cdot log\left(\sum_i^C e^{z/T}\right), \tag{6}$$

where z is the logits produced by the neural network p_θ for a C-way classification problem, and T is a temperature scalar equal to 1. In essence, $\mathcal{E}(\cdot)$ is the *logsumexp* operator over the logits. Applying $\mathcal{E}(\cdot)$ operator over each client's dataset in federated round r, where the locally trained model ($p_{\theta_m^r}$) is used to compute logits, we can acquire a scoring set, $\mathcal{S}_{\theta_m^r}$, which contains a score for each locally stored sample.

Next, to differentiate between wrongly and correctly labeled instances, we utilize a thresholding mechanism based on the energy scores obtained from the aggregated model in the same federated round r (G^r). The threshold value is computed from the v^{th} percentile over the obtained score set, $\mathcal{S}_{\theta_G^r}$, though other statistical measures, e.g., median or mean, can also be used. Specifically, we compute the threshold τ_v as:

$$\tau_v = \mathcal{P}_v\left(\mathcal{S}_{\theta^G}\right) = \underset{[S_{\theta^G}]}{arg\,max}\ \left(\frac{v}{100} \cdot |\mathcal{S}_{\theta^G}|\right), \tag{7}$$

where $[\mathcal{S}_{\theta^G}]$ and $|\mathcal{S}_{\theta^G}|$ are the ordered set and cardinality of $\mathcal{S}_{\theta^G}$, respectively. Intuitively, "*clean*" clients, holding high-quality labels, produce drastically different scores from clients containing noisy labeled data, while computing a threshold from the same global unified model G provides a "*common*" ground to ensures a clear separation between them. With both the scoring set $\mathcal{S}_{\theta_m^r}$ and the threshold τ_v computed, we can estimate a per-client noise level by counting the percentage of m^{th} client's local instances that are below the obtained τ_v as:

$$n_l^m = \frac{1 - \sum_{i=0}^{N_m} u_{\tau_v}(s_i)}{N_m} = \frac{1 - \sum_{i=0}^{N_k} s_i > \tau_v}{N_m}, \tag{8}$$

where $u_{\tau_v}(\cdot)$ is a "τ-*shifted*" Heaviside function, which produces 1 for all inputs above a threshold τ.

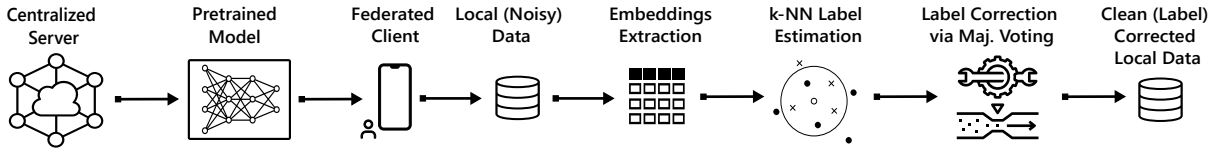


Fig. 1. Illustration of NNC for identifying and correcting noisy labeled instances via embeddings. The process is depicted for one federated client for the sake of simplicity.

Concisely, we estimate the noise level using the model confidence by computing locally, on each client, a pair of computationally inexpensive scoring sets in a single federated round r ($\mathcal{S}_{\theta_m^r}, \mathcal{S}_{\theta_G^r}$). While the produced noise level can be a rough estimation of actual noise level present in clients' data, as opposed to the embedding based one, it can be used effectively as a proxy to identify and exploit "*clean*" clients with noisy data, as we showcase in Section 3.3.

3.3 Federated Learning under Presence of Label Noise

The objective of supervised training in presence of label noise is to learn a model, where the effect of such labeled instances on model's performance is minimal. In a centralized setting, the detection of noisy labeled samples can be performed by directly applying a data-centric approach to detect outliers, or based on a model decision [23, 43, 45]. Under federated learning regime, however, a more complicated analysis is required, as clients' local data follow a non-i.i.d. distribution both in terms of data samples and noise profiles, while no global entity can directly access the complete pool of data. This leaves most of centralized approaches for handling noisy labels infeasible or ineffective, when applied in FL. To tackle this issue, we present approaches that utilize a per-client noise level estimates (as described in Section 3.2) to mitigate the effect of noisy labels on the federated model's performance.

3.3.1 ***Nearest Neighbor-based Correction (NNC).*** Input embeddings (i.e., feature extracted from a specific layer of a neural network) computed from a pre-trained model can be exploited to perform label correction. As discussed in Section 3.2, we estimate the label for each input instance x by "*looking*" at the labels of a neighbourhood examples in the embedding space. Thus, apart from predicting a per-client noise estimation using the kNN predictions, we can also perform label correction [45]. Specifically, when we detect a label mismatch between the predicted (with kNN) and current label, we consider the predicted label as the true label to be used during the training phase. This way, instead of discarding noisy instances altogether, their labels are modified on a per-client basis at the initialization phase of FL, after which the FL training process continues as usual. Mathematically, the "*corrected*" local datasets across all clients in FL can be written as:

$$\mathcal{D}^{m^*} = \left\{x_i, y_i^{vote}\right\}_{i=0}^{N_m} = \left\{x_i, \Psi\left(g, x_i\right)\right\}_{i=0}^{N_m}, \tag{9}$$

where $\Psi(\cdot)$ is a mapping function that adjusts y_i to y_i^{vote} (as computed in Equation 4) using the feature extractor $g(\cdot)$. To learn from the "*corrected*"datasets across all clients, we apply the cross-entropy loss as:

$$\mathcal{L}_\theta(\mathcal{D}^{m*}) = \mathcal{L}_{CE}\left(y^{vote}, p_{\theta^m}(y|x)\right) = -\frac{1}{N_m}\sum_{i=1}^{N_m}\sum_{j=1}^{C} y_i^{votej}\log(f_i^{\theta^m}(x_j)), \tag{10}$$

It is important to highlight that our approach remains unaffected by the level of label noise present at each client, whereas it largely depends on the quality of the extracted embeddings from a pre-trained model. However, we do not see it as a major problem as several self-supervised pre-training approaches [25, 28, 29] provide useful embeddings for broad spectrum of tasks. In addition, clients are required to hold a pre-trained model only for

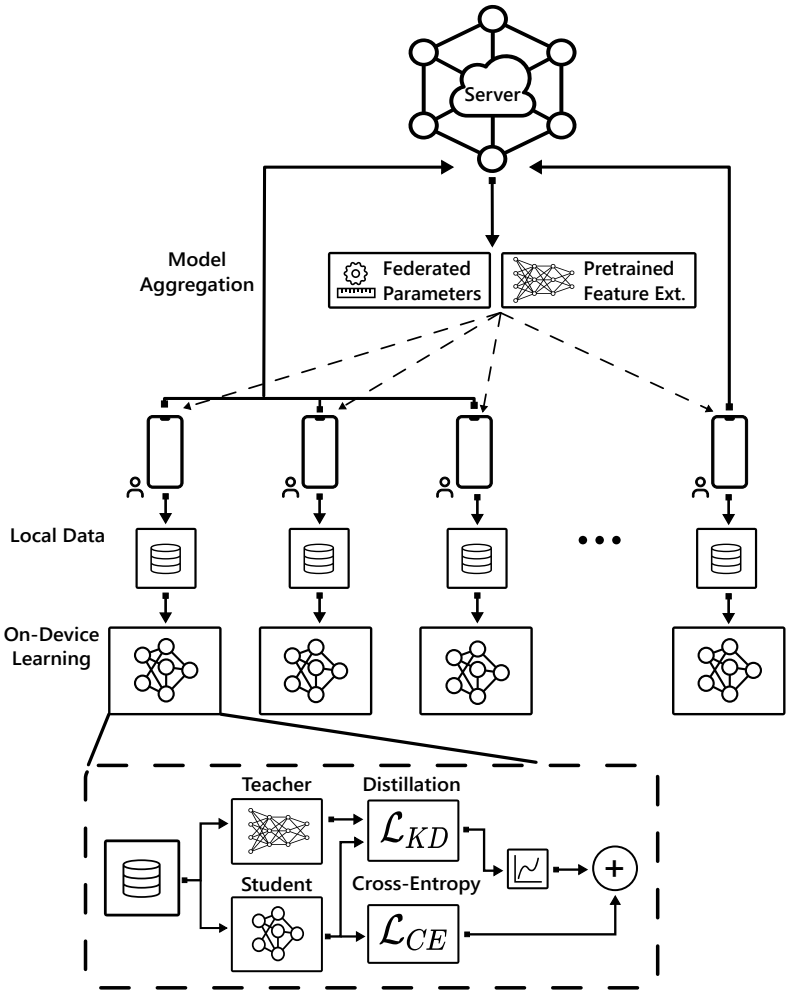


Fig. 2. Illustration of our AKD approach for training federated models under the presence of noisy labels with "*adaptive*" Knowledge Distillation (see Section 3.3.2). Embeddings are extracted from a pre-trained model, as an extra source of supervision for local models, while KD is activated on a per client-basis based on global noise threshold (computed on server-side) in case label noise is detected.

a single forward-pass during the initialization phase of FL. Further details and an overview of our proposed NNC federated approach for learning models under the presence of label noisy can be found in Algorithm 1, highlighted in red color.

3.3.2 ***Adaptive Knowledge Distillation (AKD).*** Rather than directly altering the labels, we can implicitly guide the model not to memorize noisy labeled samples during the FL training phase through means of an additional loss term. Specifically, in addition to the standard cross-entropy loss, we propose to use a knowledge distillation loss [10, 27] that requires each client to also learn to mimic embeddings or output of a teacher model.

In contrary to the NCC, in which we proposed a label correction process for training models under the presence of noisy labeled instances, with AKD, we utilize soft-labels or embeddings from a teacher model as ground-truth.

To this end, we propose to utilize embeddings computed with a pre-trained model or the globally aggregated model's outputs as a source of supervision. Specifically, after the noise level being estimated on a per-client basis, as discussed in Section 3.2, we incorporate a distillation loss term only on clients with noisy labeled data, leaving "*clean*" clients to directly train on their locally stored data. Thus, we present an "*adaptive*" knowledge distillation (AKD) during FL training process, where noisy clients' models are guided with additional supervision to learn noise-robust models. To learn from the datasets of all clients, $\mathcal{D}^m$, we apply the cross-entropy loss and an knowledge distillation loss as:

$$\mathcal{L}_\theta(\mathcal{D}^m) = \mathcal{L}_{CE}\left(y, p_{\theta^m}\left(y \mid x\right)\right) + \beta \cdot u_\epsilon(n_l^m) \cdot \mathcal{L}_{KD}\left(e, p_{\theta^m}\left(e|x\right)\right)$$
$$\text{where , } \mathcal{L}_{KD} = \begin{cases} \mathcal{L}_{MAE} = \frac{1}{N_m}\sum_{i=1}^{N_m}|e_i - \tilde{e}_i| & \text{, if } e \text{ are embeddings of } g(\cdot) \\ \mathcal{L}_{KL} = T^2 \sum_{j=1}^{C} p_{\theta^m}^T log\frac{p_{\theta^m}^T}{p_{\theta^G}^T} & \text{, if } e \text{ are "}\textit{temperature-scaled}\text{" logits of } G \end{cases} \tag{11}$$

Here, $\mathcal{L}_{CE}(\cdot)$ indicates the standard cross-entropy loss, $\mathcal{L}_{MAE}(\cdot)$ corresponds to the mean absolute error loss between the embeddings computed from a feature extractor $g(\cdot)$ and the local model of the m^{th} client (noted as $\tilde{e}_i$), and $\mathcal{L}_{KL}(\cdot)$ refers to the Kullback-Leibler divergence loss [10] between the "*temperature-scaled*" logits of m^{th} client's local model and the globally aggregated model G, where the temperature scalar T is equal to 2. With the help of a "ϵ-*shifted*" Heaviside function, u_ϵ, we apply the $\mathcal{L}_{KD}$ loss only to clients, whose estimated label noise exceeds ϵ, while we add the scalar β (β=10, which we found to be working well during our initial exploration) to control the contribution of these losses on model optimization. Further details and an overview of our proposed AKD federated approach can be found in Algorithm 1, highlighted with green color.

3.3.3 ***Noise-aware Federated Averaging (NA-FedAvg)***. As an alternative to the previously proposed approaches to deal with noisy labels, we also aim to directly tackle the effect of noisy labels at a later stage of the training process. Specifically, during the server-side aggregation process of *FedAvg* [21], we propose to utilize the estimated client's noise level to perform a "*noise-aware*" *FedAvg* (*NA-FedAvg*) aggregation step, which considers both the number of samples and the number of noisy labels in the client's data. Particularly, for a client m, the estimated noise level and the impact of the clients model on the construction of the global model G in a given federated round are inversely related, such that "*clean*" clients, holding well-annotated data contribute more to the FL process in contrast to clients with noisy labeled instances. To minimize the computation overhead in our approach, we compute a noise estimation directly from model's predictions confidence. Thus, *NA-FedAvg* is ideal for clients with minimal computational resources, as clients are only required to compute computationally inexpensive scores for each locally stored sample, while the bulk of the work is now off-loaded to the server that computes a noise level estimation for each client, as discussed in Section 3.2.

Concisely, *NA-FedAvg* begins with standard FL process up to a certain federated round R_c, which is considered a hyperparameter of *NA-FedAvg*. In round R_c, after receiving globally aggregated weights $\theta_{R_c}^G$, each client performs a local training step on the global model weights and now possesses a local model with weights $\theta_{R_c}^m$. Using these two local and global models, and only for round R_c, the scoring sets $\mathcal{S}_{\theta_{R_c}^G}^m$ and $\mathcal{S}_{\theta_{R_c}^m}^m$ are computed on a per-client basis and communicated back to the central server, together with each model's parameters $\theta_{R_c}^m$. Using the acquired scoring sets, we compute a per-client noise estimation at the server-side, using Equation 8, and the estimated noise level is introduced in the aggregation step of the FL, while the remaining FL training process continues as usual. Specifically, in the "*Noise-Aware*" FedAvg process, we have:

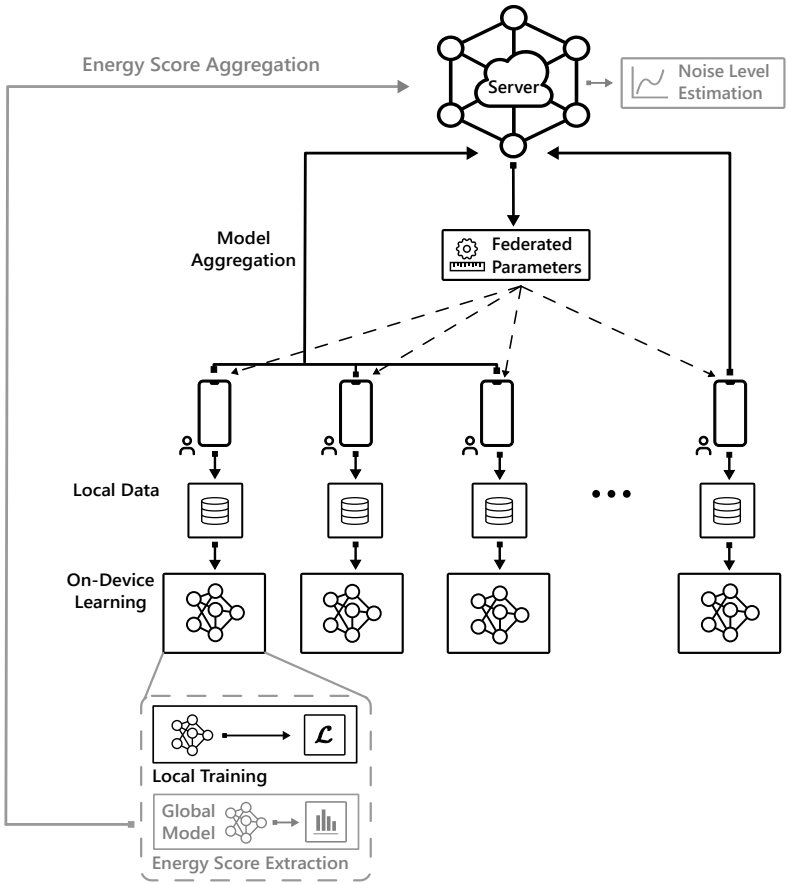


Fig. 3. Illustration of NA-FedAvg method for mitigating the effect of label noise on model's generalizability. In addition to the standard FedAvg process, a noise estimation process is introduced once, indicated with grey colors. In this round, energy scores are computed and communicated to server. Afterwards, the global model and a noise estimation per client is calculated on server. For the remaining FL training rounds, a typical local train step is performed, while server performs a noise-aware weighted model aggregation, incorporating clients' estimated noise level.

$$\min_{\theta} \mathcal{L}_{\theta} = \sum_{m=1}^{M} \left(1 - n_l^m\right) \cdot \gamma_m \mathcal{L}_m\left(\theta\right) \text{ where } \mathcal{L}_m\left(\theta\right) = \mathcal{L}_{CE}\left(y, p_{\theta^m}\left(y|x\right)\right), \tag{12}$$

where $\mathcal{L}_{CE}(\cdot)$ is the cross-entropy loss for the m^{th} client's local data ($\mathcal{D}_m$), γ_m denotes the relative impact of the m^{th} client on the construction of the global model G (e.g., $\gamma_m = \frac{N_m}{N}$) due to the amount of locally stored data, and n_l^k is the estimated noise level of the m^{th} client's local data $\mathcal{D}_m$. Further details and an overview of our "*Noise-Aware*" FedAvg approach can be found in Algorithm 1, highlighted in blue color.

4 EXPERIMENTS

In this section, we describe our extensive performance evaluation for FedLN. We use various publicly available datasets to determine efficacy of our approaches in learning generalizable models under a variety of federated

Algorithm 1 FedLN: Federated learning under Label Noise. We develop three distinct approaches to deal with label noise in learning models from decentralized data, whereas FedAvg [21] is the base algorithm and it is to indicate the key contributions of our proposed approaches. In the algorithm, scalar R_c indicate the activation round for NA-FedAvg algorithm and η is the learning rate.
FedAvg , NNC , AKD , NA-FedAvg

Server initialization of model G with model weights θ_0^G, n_s^k=0, $\forall k \in K$
for each client $k \in K$ **in parallel do**
  for $(x_i, y_i) \in \mathcal{D}^k$ **do**
    $\tilde{y_i} \leftarrow \Phi\left(g, x_i\right)$
    $e_i \leftarrow g\left(x_i\right)$
  end for
  $n_l^m = \frac{\sum_{i=0}^{N_m} y_i^{vote} \neq y_i}{N_m}$
end for
for $r = 1, \ldots, R$ **do**
  Randomly select M clients to participate in round i
  for each client $m \in M$ **in parallel do**
    $\theta_r^m \leftarrow \theta_r^G$
    θ_{r+1}^m, $(S_{\theta^G}^m, S_{\theta^{m+1}}^m) \leftarrow$ ClientUpdate(θ_r^m,r,n_l^m)
  end for
  if $r = R_c$ **then** $n_l^m = \frac{1-\sum_{i=0}^{N_m} u_{\tau_v}\left(S_{\theta^{m+1}}\right)}{N_m}$ **end if**
  $\theta_{r+1}^G \leftarrow \sum_{m=1}^{M} \frac{N_m}{N} \theta_{r+1}^m$
  $\theta_{r+1}^G \leftarrow \sum_{m=1}^{M} \left(1 - n_{sl}^m\right) \cdot \frac{N_m}{N} \theta_{r+1}^m$
end for
procedure ClientUpdate(θ, r, n_l)
  for epoch $e = 1, 2, \ldots, E$ **do**
    for batch $b \in \mathcal{D}^k$ **do**
      $\acute{\theta} \leftarrow \theta - \eta \nabla_\theta \left(\mathcal{L}_{CE}\left(y^{vote}, p_\theta\left(y|x_b\right)\right)\right)$
      $\acute{\theta} \leftarrow \theta - \eta \nabla_\theta \left(\mathcal{L}_{CE}\left(y, p_\theta\left(y|x_b\right)\right) + \beta \cdot u_\epsilon(n_l) \cdot \mathcal{L}_{KL}\left(e_b, p_\theta\left(y|x_b\right)\right)\right)$
      $\acute{\theta} \leftarrow \theta - \eta \nabla_\theta \left(\mathcal{L}_{CE}\left(y, p_\theta\left(y|x_b\right)\right)\right)$
      if $r = R_c$ **then** $s_{b,\acute{\theta}} \leftarrow \mathcal{E}\left(x_b, p_{\acute{\theta}}\right)$, $s_{b,\theta} \leftarrow \mathcal{E}\left(x_b, p_\theta\right)$ **end if**
    end for
  end for
  return $\acute{\theta}$, $(S_\theta, S_{\acute{\theta}})$
end procedure

leaning and label noise settings. Firstly, the utilized datasets are presented, followed by a detailed description of the neural network architectures and FL framework used in our experiments. Next, we present our evaluation

strategy, including all considered federated parameters and baselines used to evaluate/compare against our methods. Finally, we provide our finding about performance of FedLN across a wide range of label noise scenarios.

4.1 Datasets

We use publicly available datasets for performance evaluation on a range of classification tasks from both the vision and audio domains. For all datasets, we use standard training/test splits for comparability purposes, as provided with the original datasets. From the vision domain, we use the CIFAR-10 [14] and FashionMNIST [38] datasets, where the tasks of interests are object detection and clothes classification, respectively. By utilizing these datasets, we facilitate the ease of benchmarking for future research. In addition, we perform experiments with the PathMNIST[40] dataset from the medical imaging. By doing this, we investigate the performance of FedLN in a domain, where noisy labels may occur due to incorrect diagnosis. From the audio domain, we use SpeechCommands (v2) dataset [35], where the learning objective is to detect when a particular keyword is spoken out of a set of twelve target classes. Apart from these datasets, we extend our evaluation to a real-world, human annotated version of popular CIFAR-10/100 [36] datasets, namely CIFAR-10N/100N, where label noise presents varied (or biased in some manner) patterns based on users' preferences.

Table 1. Details of the datasets used in our experiments

Dataset	Task	Classes
CIFAR-10 [14]	Object detection	10
FashionMNIST [38]	Clothing classification	10
PathMNIST [40]	Pathology reporting	9
SpeechCommands [35]	Audio keyword spotting	12
CIFAR-10N [36]	Object detection	10
CIFAR-100N [36]		100

For all image classification tasks, we perform standard augmentations, such as random flipping and cropping, followed by Cutout [4] transformation. For the SpeechCommands audio dataset, we extract log-Mel spectrograms from raw waveforms as our model input. We compute this by applying a short-time Fourier transform on the one-second audio segment with a window size of 25 *ms* and a hop size equal to 10 *ms* to extract 64 Mel-spaced frequency bins for each window. In order to make an accurate prediction on an audio clip, we average over the model predictions of non-overlapping segments of an entire audio clip.

4.2 Implementation Details and Evaluation Strategy

In this subsection, we provide details regarding the experimental settings under which we assess FedLN , together with the utilized models and their optimizations.

Models and Optimization. We use different model architectures, one for the vision and another one for the audio domain. For our image classification tasks, we choose a ResNet-20 [9] model architecture due to its relatively compact model size, which makes it ideal for on-device learning, where devices have medium to low computational resources. Here, we utilize an SGD optimizer with a learning rate of 0.1 and momentum of 0.9 for both CIFAR-10 and FashionMNIST, while the Adam optimizer with the default learning rate of 0.001 was used for the PathMNIST.

For the audio domain, the network architecture of our global model is inspired by [32] for mobile devices. Our convolutional neural network architecture consists of four blocks. In each block, we perform two separate

convolutions: one on the temporal and another one on the frequency dimension, outputs of which we concatenate afterwards in order to perform a joint 1×1 convolution. Using this scheme, the model can capture fine-grained features from each dimension and discover high-level features from their shared output. Furthermore, we apply L2 regularization with a rate of 0.0001 in each convolution layer and group normalization [37] after each layer. Between model blocks, we utilize max-pooling to reduce the time-frequency dimensions by a factor of 2 and use a spatial dropout rate of 0.1 to avoid over-fitting. We apply ReLU as a non-linear activation function and use Adam optimizer with the default learning rate of 0.001 to minimize the loss function.

For our methods as discussed in Sections 3.2 and 3.3.1, which relies on embeddings, we use off-the-self pretrained models trained on large-scale datasets in an unsupervised manner. For vision models, we use ViT-B/32 from CLIP [25] and for audio we leverage TRILLsson (v3, EfficientNetv2-B3) [29], which has a same audio front-end as we used for our audio model. These publicly available models can be downloaded directly on client devices and we run them once (i.e., forward-pass only) to compute embeddings from client's local storage.

Federated Environment. To simulate a federated environment, we use the Flower framework [3] and utilize FedAvg [21] as the optimization algorithm to construct the global model from clients' local updates. Additionally, a number of primary parameters, listed in Table 2, were selected to control the federated setting in our experiments. In all `FedLN` experiments, where noise level estimation is computed based on a model's confidence, we use a fixed scoring threshold percentile ν to 75%, while for embedding-based noise discovery the neighborhood size in kNN to 100, which we found to be working well during our initial exploration. Further, we set the clients' participation rate in each federated round (q) to be equal to 80%. It is important to note that we employ uniform random sampling for the clients' selection strategy, as other approaches for adequate clients election are outside the scope of current work.

Table 2. Primary Experiment Parameters.

Name	Parameter	Range
Number of Clients	M	30
Number of Federated Rounds	R	1—200
Number of Local Train Steps	E	1
Clients' Participation Rate	q	80%
Noise Level	n_l	0—100%
Noise Sparsity	n_s	0—100%
Percentage of Noisy Clients	F	0—100%
Data Distribution Variance across Clients	σ	25%

For the data distribution process in our federated experiments, we randomly partitioned the datasets across the available clients in a non-overlapping fashion, controlling the amount of data across clients through parameter σ. With the σ set to 25% and a random partitioning of data among clients, the data distribution across clients resembles a non-i.i.d. setting. It is worth mentioning that even if the meaning of non-i.i.d. is generally straightforward, data can be non-i.i.d in many ways. In our work, the term non-i.i.d describes a data distribution with both a label distribution skew and a quantity skew (data samples imbalance in terms of classes across clients). This type of data distribution is common across clients' data in federated setting [12], where each client frequently corresponds to a particular user (affecting the label distribution), and the application usage across clients can differ substantially (affecting the label distribution).

To generate label noise in the datasets, we constructed a noise matrix $Q_{y|y^*}$ based on parameters n_l and n_s, similar to [23]. While in centralized settings a single noise matrix was considered, in FL we constructed a unique

noise matrix per client, thus introducing distinct noise profiles across clients. We note that the noise injection process has been performed after the data partioning process. With parameter F, we controlled the number of "*noisy*" clients, i.e., clients' holding noisy labeled instances in their datasets. Lastly, for an accurate comparison between our experiments, we manage any randomness during data partitioning, label noise injection and training procedures by using a seed alongside the parameters as presented in Table 2.

Benchmark Baselines. As the effect of label noise remains mostly unexplored in federated setting, we perform a thorough evaluation across a wide range of existing techniques used in centralized setting to handle noisy labels . From the perspective of regularization techniques, we consider Label Smoothing [22], which "*softens*" the labels by taking a weighted average of the hard targets and the uniform distribution over labels. Such "*smoothing*" of labels aims to accounts for the fact that datasets could contain mislabeled instances, thus maximizing the likelihood of $p\,(y,x)$ directly can be harmful. In our experiments, we adjusted the Label Smoothing rate, α, to be equal to 0.2 [20]. Additionally, we conduct experiments, where we use Bi-Tempered [1] loss instead of cross-entropy. This loss replaces softmax function with a high temperature generalization and uses a low temperature logarithm. In this way, Bi-Tempered loss is able to construct a decision boundary less susceptible to noisy labels and learn from outliers (which are considered noisy labeled instances) present in the data.

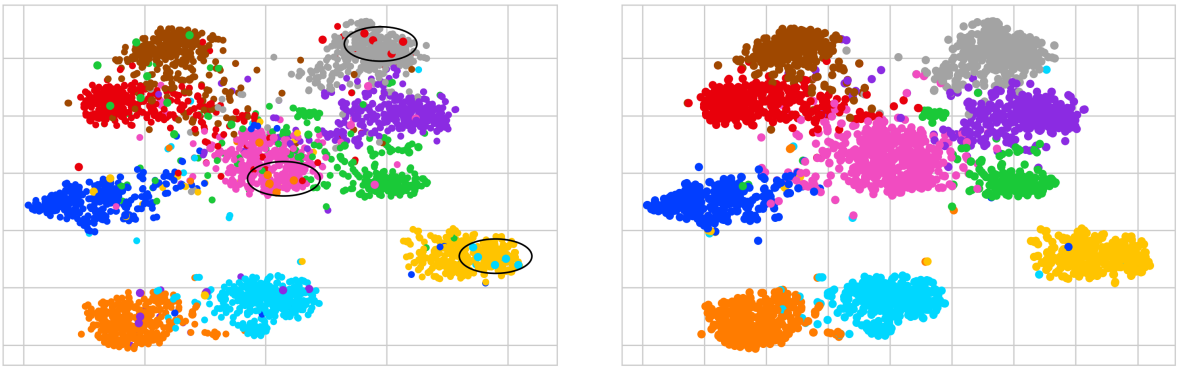
Furthermore, from a noisy instances detection perspective directly from data, we consider Confidence Learning (CL) [23]. It prunes noisy instances from a dataset based on probabilistic thresholds from a neural network's predictions. To compute these probabilistic thresholds, we utilize the globally unified model G during the initialization phase of FL, which we train locally for a fixed number of epochs (E=20) on all clients. Once the data pruning was performed using these thresholds, we discard the trained model and resume the FL training process to train a model from scratch. Apart from these methods, we perform preliminary experiments under standard centralized and federated setting with clean data. With the centralized experiments, we aim to establish an upper bound of performance for our FL approach, while the FL experiments serve a baseline to evaluate the performance improvement we can obtain with `FedLN` , when noisy labels are present. Lastly, for a rigorous evaluation, we perform three distinct trials (i.e., running an entire federated experiment) in each setting, and the average accuracy over three runs is reported across the results of Section 4.3.

4.3 Results

In this subsection, we discuss our findings on the effectiveness of `FedLN` to deal with label noise in FL. First, we show the efficacy of our noise detection mechanisms, Next, we study the performance of `FedLN` across a wide range of classification tasks. Finally, we explore how our approaches handle different model architectures, distinct noise profiles, and real-life label noise settings.

4.3.1 ***Mechanisms for label noise detection in FL.***

The estimation of noise level on a per-client basis and the detection of "*clean*" clients, holding well-annotated data, is a fundamental block of our `FedLN`framework. By exploiting this information, we can compose learning schemes to mitigate the effect of label noise during training. For this purpose, we first perform experiments, where we evaluate the performance of our proposed label noise detection mechanisms, namely *embedding-based* and *model's confidence-based*, on CIFAR-10 and SpeechCommands for different noise profiles. Additionally, we conduct experiments, where, model's predictions (i.e., softmax scores), instead of energy scores, were used to provide a noise level estimation. From our considered baselines, we used CL's [23] data pruning process as an approximation to noise level, where the noise level correspond to the percentage of pruned data. Furthermore, we perform experiments with identical data partitioning and noise injection processes are across methods to ensure an unbiased evaluation.



(a) t-SNE on embeddings for n_l=40% and n_s=0%.

(b) t-SNE on embeddings for n_l=40% and n_s=0% after NNC label correction process.

Fig. 4. Intuition behind FedLN embedding-based approaches for noise level estimation. Figures (a) and (b) show t-SNE on computed embeddings using [25] for CIFAR-10 before and after the label correction using NNC (see Section 3.3.1). Few cases of noisy labeled instances are highlighted with black circles in (a), where examples of one class are incorrectly assigned label from another class. Embedding-based discovery of noisy labels relies on the detection of outlier in a given neighborhood.

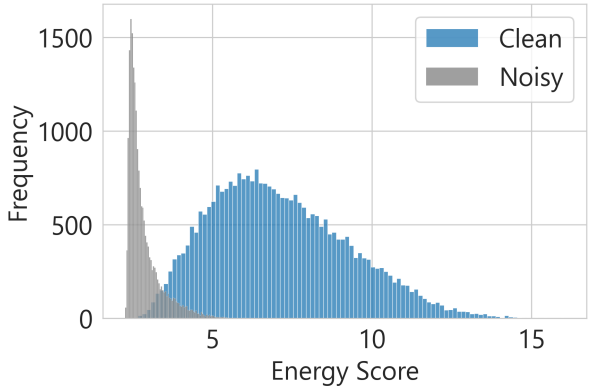


Fig. 5. Energy scores computed for CIFAR-10 on R=30 using client's local models. Clean and noisy terms refers to the presence of label noise in each client's dataset. A clear separation between the scores computed from clean and noisy clients is evident. Note that labels are not explicitly used to compute these scores.

While the embedding-based discovery of noisy labels does not depend on the clients' models, the remaining *model-exploitation* approaches require clients' models to be trained for certain number of federated rounds (referred to as R_c in Section 3.3). To this end, we initially experiment with all model-based approaches to derive a suitable training point, R_c, where noisy labeled instances can be effectively detected across clients. Figure 6 provide obtained AUC score on detection of noisy labeled instances across all clients in different federated rounds. We observe that energy scores provide an effective mechanism for detection of noisy labeled instances, following a steep curve and outperforming CL across all considered noise profiles. In particular, after 30 federated rounds,

Table 3. AUC scores on detection of noisy labeled data in CIFAR-10 and SpeechCommands on R=30. CL refer to Confidence Learning [23], while Softmax and Energy [19] correspond to our model confidence approach, when the corresponding scoring function is used. Embeddings account for the proposed embedding-based discovery of noisy labels during the initialization phase of FL. The remaining federated parameters are set to M=30, F=80%, E=1, q=80%, and σ=25%.

Noise Profile (n_l% , n_s%)		CL	Softmax	Energy	Embeddings
n_l=40	n_s=0	60.37	57.25	68.80	**93.55**
	n_s=40	58.13	55.27	65.07	**93.43**
n_l=70	n_s=0	52.23	52.81	57.71	**91.63**
	n_s=70	53.73	51.89	54.40	**62.29**

(a) CIFAR-10

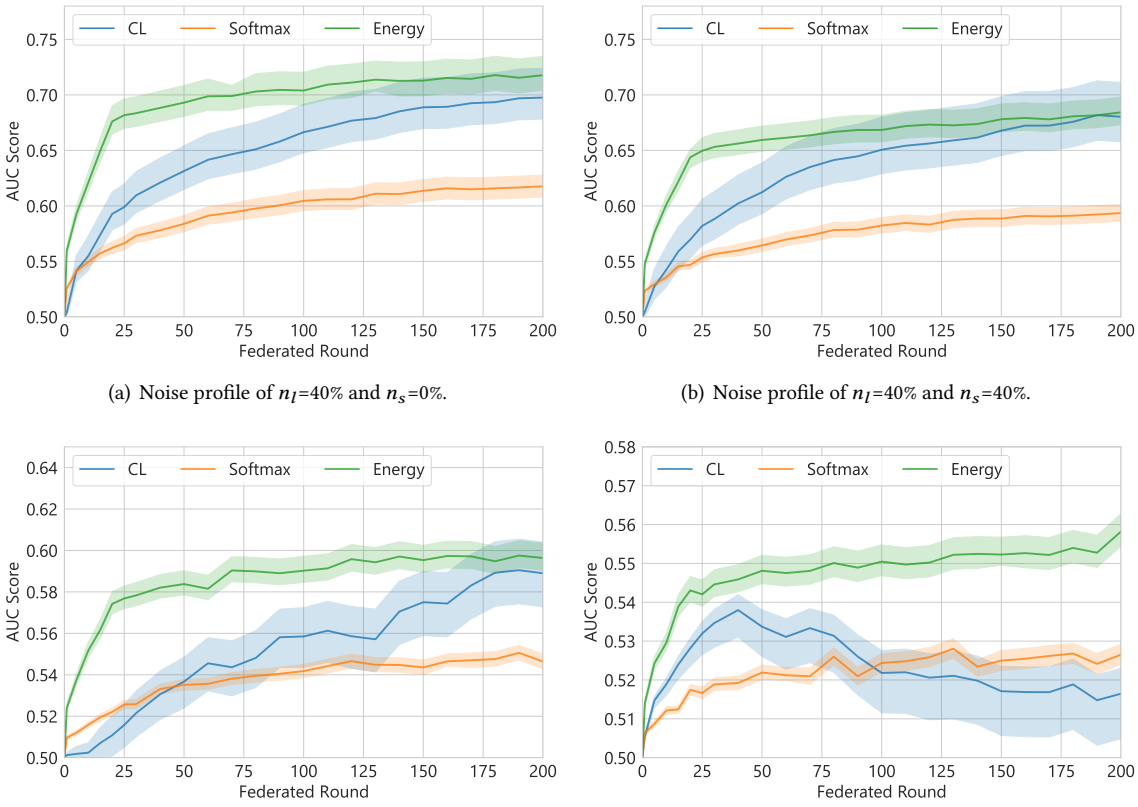
Noise Profile (n_l% , n_s%)		CL	Softmax	Energy	Embeddings
n_l=40	n_s=0	62.37	59.12	70.21	**83.16**
	n_s=40	61.85	57.98	69.59	**83.07**
n_l=70	n_s=0	54.37	52.93	63.01	**82.57**
	n_s=70	54.22	52.02	59.33	**60.11**

(b) SpeechCommands

energy score is within a close proximity of it's maximum AUC score across all federated rounds. Therefore, for the results reported in the remaining of Section 4.3, we use a fixed R_c=30 for all model-based approaches to identification of noisy clients.

In Table 3, the AUC scores of all considered approaches is reported for CIFAR-10 and SpeechCommands. Comparing the model's energy scores approach with embedding-based ones, we note that embeddings performance is superior. This is due to the difference in the quality of the noise-free *source* that is utilized to detect noisy labeled instances on the proposed methods. While embeddings are extracted from a pre-trained model, completely unrelated to clients' own models or their label noise. For model-based approaches, "*clean* clients' confidence is used to detect noisy instances. Thus, with an increase on label noise across federated setting, a certain degree of noise may implicitly be introduced to "*clean*" clients during the model aggregation procedure. We can also observe the impact of noise level on the noisy labeled instances detection rate of all model-based methods from the decreasing AUC score by an average of 7% across the two datasets, when n_l is increased from 40% to 70%. However, even if samples with label noise might be problematic to detect in high levels of label noise, the noisy and clean clients' models can still differ significantly in terms of generalization capability. To clearly illustrate this point, we present a histogram of the energy scores obtained at R=30 in Figure 5, where, we distinguish between the scores obtained from *clean* and *noisy* clients. We observe a clear separation between their scores' distributions, one for each *group* of clients.

On the contrary, embedding-based discovery is more robust to the client's noise profile. An exception to this, is the case of high noise with high sparsity (n_l=70%, n_s=70/100%), where, labels that belong to groups of classes are easily confused. Specifically, we note from Table 3 that a AUC score decline of approximately 44% occurs on average when n_l=70% and n_s increases from 40% to 70%. While the computed embeddings are unrelated to the noise profile (because a pretrained model is used to compute them) on the labels, embeddings are amplifying the noise level through faulty label estimation. To understand why this is the case, we illustrate the intuition behind how noisy labeled instances are identified using our embedding-based approach in Figure 4, we visualize t-SNE. The t-SNE is is computed for embeddings (extracted using [25]) of CIFAR-10. We distinguish example areas of noisy labeled instances with a black circle in Figure 4(a). As discussed in Section 3.3.1, discovery of noisy labels using NNC relies on the detection of outlier in a given neighborhood via majority vote, as shown in Figure 4(b). However, a high noise sparsity results in labels from a group of classes to be mixed. Thus, the label estimation can be seen from the concentration of noisy labels from a particular class in a given neighborhood, amplifying the noise level. For instance, consider the neighborhood of a cat's image (ground truth label) in the embedding space, where 7 out of the 10 samples are mislabeled as dogs due to high noise sparsity between cats and dogs. NNC computes the estimated label based on the dominant label in the neighborhood; thus, a dog's label will be assigned.



(a) Noise profile of n_l=40% and n_s=0%.

(b) Noise profile of n_l=40% and n_s=40%.

(c) Noise profile of n_l=70% and n_s=0%.

(d) Noise profile of n_l=70% and n_s=70%.

Fig. 6. Performance evaluation of model confidence based mechanisms for detection of noisy labeled instances. CL refer to Confidence Learning [23], while Softmax and Energy [19] correspond our approach, when the corresponding scoring function is been utilized. The mean AUC score across clients in different federated rounds is reported on CIFAR-10, while variance is indicated from the shaded line. Federated parameters are set to R=200, M=30, F=80%, E=1, q=80%, and σ=25%.

To overcome this problematic behavior of embedding-based label estimation, AKD (see Section 3.3.2) can be used to exploit embeddings for distillation rather than fixing labels directly, as we demonstrate in Section. 4.3.2.

4.3.2 *Comparison of* FedLN *against existing techniques.*

Here, we compare FedLN to determine the achieved improvements versus other considered baselines, and assess FedLN effectiveness on a wide range of tasks from both vision and audio domains. To this end, we perform experiments on all datasets for a diverse number of noisy profiles, varying both noise level (n_l) and sparsity (n_s). For a fair comparison, we utilize identical data partitioning and noisy injection schemes in all related experiments. Table 4 provides the accuracy scores on test sets averaged across three independent runs to be robust against differences in randomness involve in training deep neural networks. In the experiments we conducted in centralized setting, models are trained until convergence to obtain the resulting accuracy on a test set, which is presented in the centralized rows of Table 4. Additionally, for ease of comparison and benchmarking,

we include the obtained accuracy when no label noise is present in data for all considered approaches and datasets thus, acting as an upper bound of models' performance. In the case of AKD, we report the performance for the case of embeddings as source of supervision in Table 4, while the accuracies for identical experiments performed for AKD with globally aggregated model's outputs (i.e., logits) as supervisory signal are provided in Table 8 of the Appendix.

In Table 4, we observe our approaches can improve the model's performance compared to standard FedAvg across all datasets significantly. Consequently, we can conclude that FedLN can be applied in a federated environment with noisy labels to boost the performance, independent of the learning task or inputs' modality. In particular, comparing the rows for n_l=70% we note an increase of 24.82% in accuracy on average using FedLN across the considered tasks compared to the standard federated model.

Observing the results obtained with the baselines, we notice that both LS and Bi-Tempered loss performance are inconsistent for a wide range of noise profiles, with LS being effective for low sparsity label noises and Bi-Tempered loss produce desirable results only on large levels of label noise. The CL performance is more stable across diverse label noise settings; however, it's overall improvement does not exceed 4% on average across the tasks. On the contrary, FedLN performance is stable across a wide range of noise profiles and provide significant improvement in model's generalization capability. From our proposed approaches, we note that NNC provides the largest improvement on model's performance across distinct noise profiles, with the exception of the special cases of "*class-flipping*" on high levels of noise (n_l=40/70% and n_s=100%). In such cases, NNC is unable to use the computed embeddings to perform the label correction process, as discussed in Section 4.3.1, while AKD can adequately handle these cases to properly utilize the embeddings and overcome the effect of label noise. Apart from the particular "*class-flipping*" noise, AKD's embedding-based distillation approach could possibly surpass the performance of NNC, in case FL runs for a longer period (R>200) at the expense of increased computational overhead. This is evident when no noise is present, where AKD surpass all other approaches in federated setting for all datasets from the vision domain. Furthermore, we observe that NA-FedAvg can remain effective across all considered noise profiles, while introducing minimal computational overhead and client-side modifications during the FL training process. Therefore, NA-FedAvg can be an ideal for clients with minimal computational and storage resources.

4.3.3 *Evaluation of* FedLN *with different model architectures.*

Our proposed methods for dealing with label noise in FL attain high performance across a wide range of classification tasks. While in NNC the label correction process does not utilize clients' model's, both AKD and NA-FedAvg approaches involves clients' models. To ensure that FedLN efficacy is not related to a specific network architecture or model optimization, we perform experiments on CIFAR-10, where, we replace the ResNet-20 model with a recent convolutional-based neural network, named ConvMixer [33]. In particular, we choose ConvMixer-128/4, with a kernel size of 2, a patch size of 5 and approximately 0.1M parameters. Such small memory footprint and low complexity, render ConvMixer-128/4 ideal for on-device learning. In our experimentation with ConvMixer, we use Adam optimizer with a default learning rate of 0.001, as suggested in the original paper [33].

From the results presented in Table 5, we note that FedLN retains high efficacy, even if a different architecture is used. In particular, for n_l=70%, we notice an increase of 5.25% in accuracy on average using FedLN compared to the standard "*FedAvg*", with all three of proposed approaches following similar performance gains to the ones observed in Table 4. This indicates that both of our embeddings and model confidence based methods can be employed in FL to mitigate the effect of label noise and improve model's generalizability, irrespective of the architecture of federated model to be learned.

4.3.4 *Effectiveness of* FedLN *across diverse label noise settings.*

So far, in our evaluation, we considered diverse noise settings, assuming that"*clean*" clients, holding well-annotated

Table 4. Performance evaluation of FedLNon different datasets against a range of baselines. Average accuracy over three distinct trials on test set is reported. Supervised refers to standard FedAvg [21] training process, while LS and Bi-Temp denote the use of Label Smoothing [22] regularization and Bi-Tempered loss [1], respectively. CL corresponds to the Confidence Learning [23] technique. In AKD, we report the performance for the case of embeddings as source of supervision. The accuracies for AKD with globally aggregated model's outputs (i.e., logits) as supervisory signals are provided in Table 8 of the Appendix. Federated parameters are set to R=200, M=30, F=80%, E=1, q=80%, and σ=25%.

Noise (n_l)			0.0	0.4				0.7			
Sparsity (n_s)			0.0	0.0	0.4	0.7	1.0	0.0	0.4	0.7	1.0
CIFAR-10	Centralized		91.52	76.61	76.98	76.91	69.47	58.83	58.33	57.42	45.06
	FedAvg	Supervised	78.52	68.77	67.05	67.31	67.92	57.65	56.94	56.81	63.54
		LS	73.91	68.63	64.91	64.02	64.68	58.08	56.53	56.21	62.04
		Bi-Temp.	75.29	66.18	65.66	66.58	67.71	56.75	57.61	58.39	63.74
		CL	73.84	68.96	67.65	68.98	68.03	59.25	60.28	61.21	64.99
	FedLN	NNC	75.29	**73.68**	**73.64**	**74.23**	**71.44**	**74.76**	**72.63**	64.49	54.59
		AKD	**83.55**	69.72	69.18	70.26	70.55	68.43	68.06	69.74	**68.39**
		NA-FedAvg	76.41	69.52	70.73	70.61	71.01	65.34	66.04	**68.11**	67.92
Fashion MNIST	Centralized		91.85	83.56	86.76	86.18	78.26	63.87	60.96	61.95	40.32
	FedAvg	Supervised	86.43	82.05	83.24	83.35	81.07	58.55	56.06	57.11	59.37
		LS	84.86	82.08	82.07	81.88	79.84	59.38	56.24	56.95	55.66
		Bi-Temp.	84.61	81.93	81.15	81.84	81.46	57.93	57.35	58.31	59.24
		CL	83.91	82.82	83.29	83.76	81.91	59.48	57.97	57.33	60.89
	FedLN	NNC	80.29	**86.81**	**87.22**	**87.77**	**83.13**	**85.38**	**84.91**	**76.88**	44.48
		AKD	**86.91**	83.97	84.06	83.53	82.35	80.77	78.63	80.41	78.64
		NA-FedAvg	86.39	83.59	84.28	84.45	83.02	78.22	78.68	76.17	**79.46**
Path MNIST	Centralized		90.65	81.16	80.92	81.02	78.05	58.33	59.82	57.75	47.89
	FedAvg	Supervised	87.05	78.82	77.06	76.68	77.03	54.74	52.49	53.22	58.61
		LS	84.13	79.62	76.96	74.57	74.9	56.17	52.06	52.31	53.46
		Bi-Temp.	83.09	78.03	77.23	77.61	76.67	55.89	55.74	56.15	58.54
		CL	84.31	78.97	79.09	77.26	81.02	59.69	55.88	54.29	60.21
	FedLN	NNC	84.45	**82.76**	**82.97**	**82.01**	78.97	**80.13**	**82.74**	78.78	41.53
		AKD	**87.82**	**86.42**	82.83	81.46	**83.26**	79.94	78.63	78.31	**76.02**
		NA-FedAvg	85.96	80.52	81.06	81.36	78.35	76.69	75.85	**79.64**	72.84
Speech Commands	Centralized		96.68	90.33	90.31	90.84	84.84	84.95	83.31	82.65	60.41
	FedAvg	Supervised	**96.31**	81.83	82.53	82.44	80.33	72.34	70.34	70.89	72.39
		LS	94.64	91.13	84.77	80.11	79.35	77.06	71.28	68.13	69.71
		Bi-Temp.	96.21	82.31	81.35	82.76	82.78	73.27	71.41	72.57	70.98
		CL	87.12	85.45	87.97	84.34	85.54	78.34	72.92	70.07	72.81
	FedLN	NNC	95.79	**95.91**	**95.95**	**95.97**	**96.24**	**96.09**	**96.11**	**96.13**	46.07
		AKD	84.82	86.42	84.83	85.46	83.26	79.94	76.63	78.31	76.02
		NA-FedAvg	96.07	89.49	90.35	92.72	94.09	79.12	81.91	82.33	**80.37**

data, are present during the FL procedure. In this subsection, we assess the efficacy of FedLN , while we relax our assumption regarding "*clean*" clients. To this end, we conduct further experiments on CIFAR-10, where the percentage of noisy clients, and the amount of well-annotated data present on "*clean*" clients are varied.

Table 5. Performance evaluation of FedLN with ConvMixer-128/4 [33] on CIFAR-10. Average accuracy over three distinct trials on test set is reported. In AKD, we report the performance for the case of embeddings as source of supervision. Federated parameters are set to R=200, M=30, F=80%, E=1, q=80%, and σ=25%.

Noise (n_l)	0.0	0.4				0.7			
Sparsity (n_s)	0.0	0.0	0.4	0.7	1.0	0.0	0.4	0.7	1.0
Centralized	91.26	76.61	76.98	76.31	69.47	68.83	68.33	67.42	55.06
FedAvg [21]	82.04	74.64	73.26	74.28	74.41	63.46	63.94	64.36	67.25
NNC	77.73	**76.53**	**77.44**	**77.82**	**76.33**	**76.82**	**76.59**	70.46	30.14
AKD	**82.25**	74.31	74.18	75.04	75.18	71.89	71.63	**72.64**	70.09
NA-FedAvg	81.90	75.85	74.79	76.92	75.74	68.61	68.97	67.66	**70.18**

Varying number of noisy clients: With the client's labeling systems and user's willingness (or ability) to perform a correct annotation process significantly varies in a federated environment, the number of noisy (and clean) clients present in the FL process can fluctuate drastically. In this ablation study, we conduct experiments to determine the effect of the number of noisy clients on FedLN performance. To this end, we perform experiments by varying the percentage of noisy clients (F) from 25% up to 100% for n_l=40% utilizing both NNC and NA-FedAvg. In this way, we are able to assess the performance of our methods for estimating label noise (namely, using embeddings and model's predictions confidence). It is important to note that F=100% corresponds to a scenario, where all clients contain n_l% of label noise in their locally stored data (not to be confused with a completely noisy dataset).

Table 6. Performance evaluation of FedLN when varying the percentage of noisy clients (F) for n_l=40% on CIFAR-10. Average accuracy over three distinct trials on test set is reported. Federated parameters are set to R=200, M=30, E=1, q=80%, and σ=25%.

% Noisy Clients (F)	***NNC***				***NA-FedAvg***			
	n_s=0%	n_s=40%	n_s=70%	n_s=100%	n_s=0%	n_s=40%	n_s=70%	n_s=100%
25	73.80	74.84	74.18	72.84	**75.99**	**75.97**	**76.59**	**76.02**
50	**74.54**	**73.59**	**74.97**	74.23	73.27	73.45	72.99	**74.36**
75	**73.50**	**74.49**	**74.96**	**73.95**	70.99	70.74	71.48	70.80
100	**73.76**	**73.41**	**73.14**	**72.49**	64.02	62.62	62.72	64.22

We present our findings in Table 6, where, we note that the number of noisy clients has a relatively low impact on NNC's ability to perform the label correction process, as embeddings are unaffected from the noisy clients. On the contrary, when trained with NA-FedAvg, model's performance deteriorates once all clients become noisy (F=100%). However, with as few as 20% of clients being "*clean*", the model's performance reaches approximately 70% across all noise profiles, as we have seen from Table 4. Furthermore, in cases, where label noise is sparse across clients (F=25%), we observe that NA-FedAvg performance is superior to NNC. Because the overall number of noisy labeled instances has negligible effect on the FL process and a small amount of label noise is introduced to "*clean*" clients due to slight imperfection in label estimation via NNC.

Varying label noise distribution across devices: Apart from varying the number of noisy devices in federated

Table 7. Performance evaluation of FedLN when noise is introduced on "*clean*" clients for n_l=40% in CIFAR-10. Average accuracy over three distinct trials on test set is reported. Federated parameters are set to R=200, M=30, F=80%, E=1, q=80%, and σ=25%.

% of n_l on clean clients	*NNC*				*NA-FedAvg*			
	n_s=0%	n_s=40%	n_s=70%	n_s=100%	n_s=0%	n_s=40%	n_s=70%	n_s=100%
0	**73.68**	**73.64**	**74.23**	**71.44**	69.52	70.73	70.61	71.01
5	**73.53**	**74.02**	**75.45**	**72.24**	69.17	70.46	70.02	70.87
10	**74.39**	**74.57**	**74.70**	**72.29**	68.12	68.13	67.18	67.53
25	**73.49**	**72.94**	**73.68**	**72.82**	67.01	66.17	64.11	66.11
50	**74.25**	**73.68**	**72.57**	**72.98**	64.88	64.02	63.32	64.57
75	**74.72**	**74.08**	**73.48**	**72.45**	63.79	63.97	63.57	64.14
100	**74.01**	**73.08**	**74.14**	**73.87**	64.02	62.62	62.72	64.22

setting, the quality of labels present on "*clean*" clients can also fluctuate in most pragmatic federated setting. This can happen due to imperfect labeling processes, which frequently occur in real-life, either due to human error or unforeseen mistake in a automated labeling system [26]. Therefore, in this ablation study we aim to assess FedLN performance, where, in addition to noisy clients, "*clean*" clients also hold a small amount of label noise. For this purpose, we perform experiments with both NCC and NA-FedAvg for n_l=40%, where we introduce a percentage of label noise to "*clean*" clients. Note that this percentage is considered a percentage of label noise n_l; thus a value of 100% corresponds to n_l% of label noise injected in data of "*clean*" clients.

From the results provided in Table 7, we observe that the performance of NA-FedAvg starts to deteriorate, when label noise is injected to "*clean*" clients data. In particular, with a noise addition of $0.1 \times n_l$, we note a 3% drop in accuracy on average across the noise profiles. This is due to the fact that NA-FedAvg assumes availability of a set of *clean* clients, where, this "*source*" enables detection of noisy labeled instances. While the overall label noise present in the FL process is similar to that of the experiments reported in Table 6; here, we add label noise to all "*clean*" clients, instead of flipping a "*clean*" client to noisy ones. In essence, by introducing noise to the "*clean*" clients, the clear separation between the energy scores obtained from the two groups of clients, i.e., clean and noisy, as illustrated in Figure 5 start to diminish. Therefore, NA-FedAvg ability to distinguish between clean/noisy clients sharply deteriorates, resulting in poor generalizability for the obtained model. On the contrary, the ability of NNC to perform label correction and produce highly generalizable models remains effective, even under the injection of n_l of noise to "*clean*" clients. Therefore, NNC can be a preferable approach in cases, where strong assumptions are made about noise profile, i.e., no clients, holding quality labels. When even a few these "*clean*" clients are present, NA-FedAvg can produce highly accurate models, as we have seen in Table 6

4.3.5 ***Real-world human annotation errors***.

Since synthetic noise often mimic clear structures to enable statistical analyses, it may fails to model complex real-world noise patterns or biases, which impose additional challenges as compared to synthetic label noise. In an effort to evaluate FedLN performance in a more realistic label noise setting, we use the re-annotated versions of the CIFAR-10/100 datasets, which contains real-world human annotation errors, namely CIFAR-10/100N [36]. In this way, we are able to study FedLN performance with label noise in-the-wild. During the labeling process by human annotators, with the help of Amazon Mechanical Turk, a noise level of approximately 40% (n_l=40%) was observed. This is evident that human labeling efforts inevitably result in considerable amount of label noise being introduced in the data. Considering the federated setting, this posses major challenges for user's, who are required

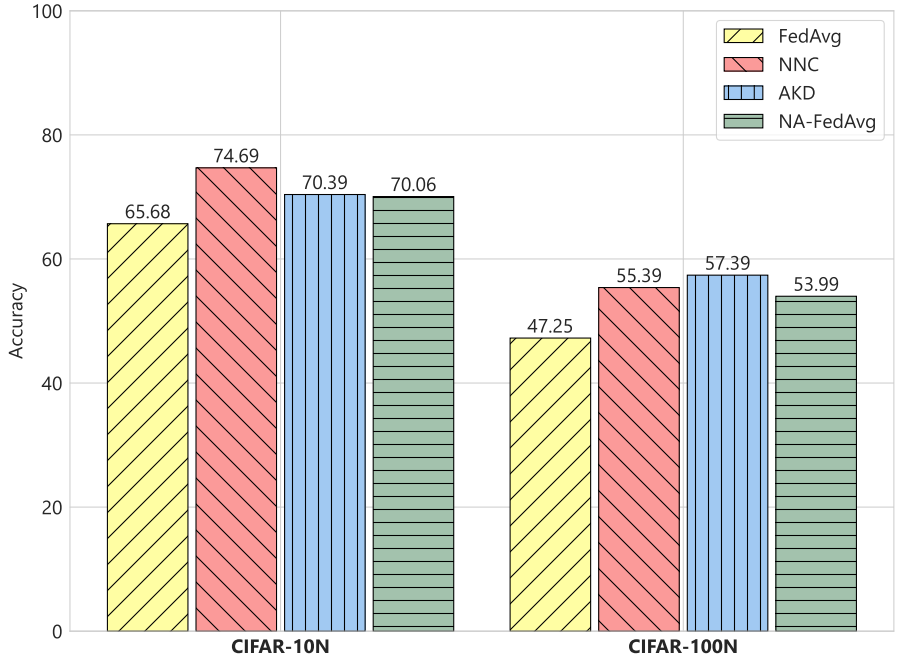


Fig. 7. Evaluation of FedLN in real-world label noise patterns from CIFAR-10/100N. Average accuracy over three distinct trials and on test set is reported for R=200/500 for CIFAR-10/100N, respectively. In AKD, we report the performance for the case of embeddings as source of supervision. Remaining federated parameters are set to M=30, E=1, q=80%, and σ=25%.

to provide well-annotated data to enjoy high-quality FL services, and further necessitates the development of approaches to adequately deal with label noise in FL.

We perform experiments on both CIFAR-10/100N datasets utilizing FedLN approaches, and include standard FedAvg with same training configuration, to clearly illustrate performance gain of our approach. It is important to note that we utilize the "worst" case version of human-annotated labels provided in [36], where ground truth label are chosen only when all annotator classify an instance correctly. While the train set of CIFAR-10/100N was annotated by humans, the original *noise-free* test set of both datasets is used for evaluation. Furthermore, we randomly distribute the data across clients; thus no clear group of "*clean*" clients is considered in this case, which make the learning even more challenging. From the CIFAR-10N results presented in Figure 7, we note that FedLN retains it's effectiveness, while moving from synthetic to real-world noise patterns. In particular, model's recognition rate remains within 2% to the ones reported in Table 4 for n_l=40%. In the case of CIFAR100N, where the complexity of the task is increased due to large number of classes, we train all models for 500 federated rounds (R=500). In this case, we observe that embedding-based supervision via AKD outperforms the remaining approaches, validating that AKD's performance can be especially beneficial if longer federated training is possible. Subsequently, we note that FedLN is able to largely address the challenges introduced from real-life noise patterns and is able to improve the recognition rate by 9%, compared to the standard FL process. Therefore, our approaches can be useful for everyday users, who utilize FL services and are requested to provide labels for their data, either explicitly or through their own activities (e.g., the next word prediction in Gboard [15]), which are inherently noisy.

5 RELATED WORK

With the advent of deep learning, a wide-range of studies has been conduced on the robustness of neural networks from various aspects, importantly against noisy labels in supervised learning [31], Since deep networks have sufficient capacity to memorize noisy labeled instances [7]; it can severely reduce their generalizability on unseen test data [36]. From the developed approaches over the years, a general family of methods to deal with label noise revolves around regularization techniques in the loss function, where the loss score is adapted directly based on the model's confidence in the provided score [24]. In this realm, the Bi-Tempered loss [1] offers a generalized softmax cross-entropy loss, which bounds loss value per sample and offers a heavy-tail softmax probability function via two tunable parameters. In [20], the effectiveness of Label Smoothing against label noise was investigated, showing that smoothing is competitive with most loss-correction techniques. Co-teaching [31] deals with noisy labels by simultaneously training two deep neural networks, which provide supervision to each other to avoid memorizing on noisy labels. As opposed to the aforementioned approaches that deal with label noise during training, Confidence Learning [23] deals with label noise by directly estimating noisy labeled instances and removing them prior to training. This pruning process is based on probabilistic thresholds from a (often, pre-trained) neural network's predictions. Apart these methods, many studies consider robust architectures, which aim to to model the noise transition matrix of a noisy dataset [39]. Nevertheless, none of the discussed approaches focuses on federated setting, where noise patterns vastly vary among clients; thus introducing additional challenges across all considered approaches; modeling noise profiles, assessing model's confidence on it's loss values, and producing appropriate thresholds to detect noisy instances. For these reasons, in our work we perform a thorough evaluation of centralized approaches that deal with label noise in federated setting, enabling future research to advance on the, yet unexplored, area of FL under the presence of noisy labels. Based on the findings, we develop `FedLN` , a framework consisting of various distinct approaches to deal with label noise in FL, each tackling the label noise on a difference phase of the training process.

Federated Learning (FL) is a collaborative learning paradigm that aims to learn a single, global model from data stored on remote (decentralized) clients with no need to share their data with a central server. Under federated setting, *FedAvg* [21] has been the "*de facto*" approach for the construction of a unified model from received clients' model updates (i.e., weights or parameters), performing weighted averaging among clients. In [44], the performance of *FedAvg* under non-i.i.d. data distribution has been studied, reporting a substantial decrease in performance compared to i.i.d. settings. From the perspective of noise, [2] has explored the presence of noise during communication. While other practical challenges of FL have need widely studied, such as sparsity of labels [34] and communication efficiency [17], label noise remains largely unexplored. Recently, authors in [41] have considered noise in the labels, and utilized the communication of class-wise data centroids among clients to construct decision boundaries among local data classes to identify noise instances. However, such learning scheme introduce additional communication costs, centroids need to be re-computed and communicated upon the introduction of new participants. More importantly, communication of user-related information may violate the privacy aspect of FL and perform poorly on clients with a unique noise profile. Furthermore, while Knowledge Distillation (KD) has been employed in FL to deal with communication efficiency [11] and non-i.i.d. data [18], it's effect on overcoming label noise on the FL train process remain unexplored. Nonetheless, there is no efficient approach to identify noisy instances decentralized data at present, while label correction under federated setting remain unexplored. To address the aforementioned problems, our proposed `FedLN` , a framework provides simple, yet effective approaches to accurately estimate label noise on a per-client basis, correct noisy labeled instances, and offer robust learning schemes to learn better generalizable federated models under the presence of label noise.

6 CONCLUSIONS

We study the pragmatic problem of label noise under federated setting. In the distributed scenario, clients' well-annotated examples are sparse due to flaws in the labeling process, which originate either from deficient automatic-labeling techniques or users' mistakes. To aggravate this problem, label noise in federated setting can follow a non-i.i.d. distribution, as it is closely coupled to a number of client-dependent sources, such as the discrepancy between clients' labeling systems or the difference in clients' expertise and willingness to label data correctly. This "*non-iid-ness*" of noise, results in centralized approaches' (to handle noisy labels) performance to deteriorate in federated setting, while the limited FL approaches deal with noisy labels introducing extensive communication overhead and relying on the exchange of user-related data. To address the lack of computationally efficient ways to deal with label noise during learning on-device models, we present FedLN framework, where we propose three different approaches. They operate in distinct phases of the FL process (FL initialization, clients' local model training, and server-side model aggregation). Apart from identifying noisy labeled instances and mitigating their effect during training, in FedLN , we provide a mechanism to perform label correction. Despite the simplicity of our approaches, namely NNC, AKD and NA-FedAvg, we demonstrate that they can address noisy labeled instances across a wide range of label noise settings. We exhaustively evaluate FedLN on several publicly available datasets, from both vision and audio domain, while comparing its performance with several baselines both in centralized and federated settings. The models' generalization we achieve is consistently superior to the considered baselines, while an evaluation of FedLN with in-the-wild label noise data, showcase that it is valuable for improving the FL services provided to respective users.

ACKNOWLEDGMENTS

The work presented in this paper is partially performed in the context of the Distributed Artificial Intelligent Systems (DAIS) project supported by the ECSEL Joint Undertaking (JU). JU receives support from the European Union's Horizon 2020 research and innovation programme and Sweden, Netherlands, Germany, Spain, Denmark, Norway, Portugal, Belgium, Slovenia, Czech Republic, Turkey.

Various icons used in the figures are created by Nanda Diga, Soremba, Andriwidodo, Product Pencil, Weltenraser, Kamin, Trevor, Olena, and David from the Noun Project.

REFERENCES

[1] Ehsan Amid, Manfred K. Warmuth, Rohan Anil, and Tomer Koren. 2019. Robust Bi-Tempered Logistic Loss Based on Bregman Divergences. (2019). https://doi.org/10.48550/ARXIV.1906.03361

[2] Fan Ang, Li Chen, Nan Zhao, Yunfei Chen, Weidong Wang, and F. Richard Yu. 2020. Robust Federated Learning With Noisy Communication. *IEEE Transactions on Communications* 68, 6 (2020), 3452–3464. https://doi.org/10.1109/TCOMM.2020.2979149

[3] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D Lane. 2020. Flower: A Friendly Federated Learning Research Framework. *arXiv preprint arXiv:2007.14390* (2020).

[4] Terrance DeVries and Graham W. Taylor. 2017. Improved Regularization of Convolutional Neural Networks with Cutout. https://doi.org/10.48550/ARXIV.1708.04552

[5] Jacob Goldberger and Ehud Ben-Reuven. 2017. Training deep neural-networks using a noise adaptation layer. In *ICLR*.

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. https://doi.org/10.48550/ARXIV.1706.04599

[7] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels. https://doi.org/10.48550/ARXIV.1804.06872

[8] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated Learning for Mobile Keyboard Prediction. *CoRR* abs/1811.03604 (2018). arXiv:1811.03604 http://arxiv.org/abs/1811.03604

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. https://doi.org/10.48550/ARXIV.1503.02531

[11] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. 2018. Communication-Efficient On-Device Machine Learning: Federated Distillation and Augmentation under Non-IID Private Data. *ArXiv* abs/1811.11479 (2018).

[12] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. 2021. Advances and Open Problems in Federated Learning. arXiv:1912.04977 [cs.LG]

[13] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2017. Federated Learning: Strategies for Improving Communication Efficiency. arXiv:1610.05492 [cs.LG]

[14] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images.* Technical Report.

[15] David Leroy, Alice Coucke, Thibaut Lavril, Thibault Gisselbrecht, and Joseph Dureau. 2019. Federated Learning for Keyword Spotting. arXiv:1810.05512 [eess.AS]

[16] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine* 37, 3 (2020), 50–60. https://doi.org/10.1109/MSP.2020.2975749

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated Optimization in Heterogeneous Networks. https://doi.org/10.48550/ARXIV.1812.06127

[18] Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. 2020. Ensemble Distillation for Robust Model Fusion in Federated Learning. https://doi.org/10.48550/ARXIV.2006.07242

[19] Weitang Liu, Xiaoyun Wang, John D. Owens, and Yixuan Li. 2020. Energy-based Out-of-distribution Detection. https://doi.org/10.48550/ARXIV.2010.03759

[20] Michal Lukasik, Srinadh Bhojanapalli, Aditya Krishna Menon, and Sanjiv Kumar. 2020. Does label smoothing mitigate label noise? https://doi.org/10.48550/ARXIV.2003.02819

[21] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv:1602.05629 [cs.LG]

[22] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. 2019. When Does Label Smoothing Help? https://doi.org/10.48550/ARXIV.1906.02629

[23] Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. 2019. Confident Learning: Estimating Uncertainty in Dataset Labels. (2019). https://doi.org/10.48550/ARXIV.1911.00068

[24] Giorgio Patrini, Alessandro Rozza, Aditya Menon, Richard Nock, and Lizhen Qu. 2016. Making Deep Neural Networks Robust to Label Noise: a Loss Correction Approach. https://doi.org/10.48550/ARXIV.1609.03683

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 8748–8763.

[26] Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel. *Proceedings of the VLDB Endowment* 11, 3 (nov 2017), 269–282. https://doi.org/10.14778/3157794.3157797

[27] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. FitNets: Hints for Thin Deep Nets. https://doi.org/10.48550/ARXIV.1412.6550

[28] Aaqib Saeed, David Grangier, and Neil Zeghidour. 2021. Contrastive learning of general-purpose audio representations. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3875–3879.

[29] Joel Shor and Subhashini Venugopalan. 2022. TRILLsson: Distilled Universal Paralinguistic Speech Representations. *arXiv preprint arXiv:2203.00236* (2022).

[30] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2020. Learning from Noisy Labels with Deep Neural Networks: A Survey. https://doi.org/10.48550/ARXIV.2007.08199

[31] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2022. Learning from Noisy Labels with Deep Neural Networks: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[32] Marco Tagliasacchi, Beat Gfeller, Félix de Chaumont Quitry, and Dominik Roblek. 2019. Self-supervised audio representation learning for mobile devices. *arXiv preprint arXiv:1905.11796* (2019).

[33] Asher Trockman and J. Zico Kolter. 2022. Patches Are All You Need? https://doi.org/10.48550/ARXIV.2201.09792

[34] Vasileios Tsouvalas, Aaqib Saeed, and Tanir Ozcelebi. 2022. Federated Self-Training for Semi-Supervised Audio Recognition. *ACM Trans. Embed. Comput. Syst.* (feb 2022). https://doi.org/10.1145/3520128

[35] P. Warden. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *ArXiv e-prints* (April 2018). arXiv:1804.03209 [cs.CL] https://arxiv.org/abs/1804.03209

[36] Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. 2021. Learning with Noisy Labels Revisited: A Study Using Real-World Human Annotations. *CoRR* abs/2110.12088 (2021). arXiv:2110.12088 https://arxiv.org/abs/2110.12088

[37] Yuxin Wu and Kaiming He. 2018. Group Normalization. arXiv:1803.08494 [cs.CV]

[38] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR* abs/1708.07747 (2017). arXiv:1708.07747 http://arxiv.org/abs/1708.07747

[39] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. 2015. Learning from massive noisy labeled data for image classification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2691–2699. https://doi.org/10.1109/CVPR.2015.7298885

[40] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. 2021. MedMNIST v2: A Large-Scale Lightweight Benchmark for 2D and 3D Biomedical Image Classification. *arXiv preprint arXiv:2110.14795* (2021).

[41] Seunghan Yang, Hyoungseob Park, Junyoung Byun, and Changick Kim. 2022. Robust Federated Learning With Noisy Labels. *IEEE Intelligent Systems* 37, 2 (2022), 35–43. https://doi.org/10.1109/MIS.2022.3151466

[42] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. 2018. Applied Federated Learning: Improving Google Keyboard Query Suggestions. arXiv:1812.02903 [cs.LG]

[43] Jing Zhang, Victor S. Sheng, Tao Li, and Xindong Wu. 2018. Improving Crowdsourced Label Quality Using Noise Correction. *IEEE Transactions on Neural Networks and Learning Systems* 29, 5 (2018), 1675–1688. https://doi.org/10.1109/TNNLS.2017.2677468

[44] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated Learning with Non-IID Data. (2018). https://doi.org/10.48550/ARXIV.1806.00582

[45] Zhaowei Zhu, Zihao Dong, and Yang Liu. 2021. Detecting Corrupted Labels Without Training a Model to Predict. https://doi.org/10.48550/ARXIV.2110.06283

APPENDIX

Table 8. Performance evaluation on different datasets for FedLN's AKD approach, while globally aggregated model's outputs are exploited as source of supervision. Average accuracy over three distinct trials on test set is reported. Federated parameters are set to R=200, M=30, F=80%, E=1, q=80%, and σ=25%. Emb. denotes embedding.

Noise (n_l)			0.0	0.4				0.7			
Sparsity (n_s)			0.0	0.0	0.4	0.7	1.0	0.0	0.4	0.7	1.0
CIFAR-10	FedAvg		78.52	68.77	67.05	67.31	67.92	57.65	56.94	56.81	63.54
	FedLN	AKD (Emb.)	**83.55**	**69.72**	**69.18**	**70.26**	**70.55**	**68.43**	**68.06**	**69.74**	**68.39**
		AKD (Logits)	78.54	63.42	62.79	67.45	68.63	54.09	53.38	59.14	64.73
Fashion MNIST	FedAvg		86.43	82.05	83.24	83.35	81.07	58.55	56.06	57.11	59.37
	FedLN	AKD (Emb.)	**86.91**	**83.97**	**84.06**	**83.53**	**82.35**	**80.77**	**78.63**	**80.41**	**78.64**
		AKD (Logits)	85.82	79.11	80.82	80.28	80.82	73.39	68.57	71.94	78.61
Path MNIST	FedAvg		87.05	78.82	77.06	76.68	77.03	54.74	52.49	53.22	58.61
	FedLN	AKD (Emb.)	**87.82**	**86.42**	**82.83**	**81.46**	**83.26**	**79.94**	**78.63**	**78.31**	**76.02**
		AKD (Logits)	85.98	77.77	79.94	75.38	80.87	57.56	59.36	49.36	60.04
Speech Commands	FedAvg		96.31	81.83	82.53	82.44	80.33	72.34	70.34	70.89	72.39
	FedLN	AKD (Emb.)	84.82	**86.42**	**84.83**	**85.46**	**83.26**	**79.94**	**76.63**	**78.31**	**76.02**
		AKD (Logits)	95.69	80.39	81.07	82.56	81.64	73.11	72.74	74.14	69.67