Federated Learning with Noisy Labels: Achieving Generalization in the Face of Label Noise

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

Abstract

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, and server-side model aggregation. Extensive experiments on various publicly available vision and audio datasets demonstrate an improvement of 24% on average compared to state-of-the-art methods for a label noise level of 70%.

1 Introduction

Federated Learning (FL) has been attracting growing attention thanks to its privacy-appealing characteristics [5]. In FL, the Federated Averaging (FedAvg) algorithm [9] is the "*de facto*" approach for the construction of a unified model from clients' model updates, showcasing great success in a wide range of tasks in recent years [7, 23, 3]. Nevertheless, a common limitation of existing FL approaches is the implicit assumption that on-device data are perfectly annotated [19]. In reality, data samples in FL either cannot be labeled readily or label quality cannot be guaranteed to the same extent as datasets that are collected and annotated in a centralized environment. Under federated setting, 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 [23]. 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 case for automatic labeling systems, such as [13] where "*weak*" labels are constructed, which are inherently noisy. Therefore, in FL, the presence of mislabeled samples, referred to as noisy labels, can naturally occur, while there is no straightforward way to perform label correction.

In centralized regimes, the problem of designing robust learning schemes in the presence of label noise has received noticeable attention recently [16], with various algorithms proposed to train models using noisy labeled samples [10, 11, 1]. 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 the non-i.i.d. nature of data in FL, such learning schemes are ineffective under federated setting. To overcome label noise in FL, authors in [22] utilized the communication of class-wise data centroids among clients to construct decision boundaries among local data classes to identify noise labeled instances. However, such learning scheme introduces additional communication costs and may violate the privacy aspect of FL. Nonetheless, there is no efficient approach to detect noisy instances in decentralized data, while label correction in FL remains unexplored.

Here, we propose a federated framework, named `FedLN` (Federated Learning with Label Noise), to provide simple, yet effective approaches to accurately estimate label noise on a per-client basis, correct noisy labeled instances, and offer robust learning scheme to learn generalizable models under the presence of label noise. This way, `FedLN` mitigates the need of user interaction for high-quality label acquisition, and is equally useful when automated labeling techniques are used for label extraction, which are inherently noisy. Our experimentation on diverse public datasets from both vision and audio domains shows that `FedLN` with only 30% of labeled data correctly annotated, on average improves recognition rate by 24% across all datasets compared to the standard FL strategy (FedAvg). To the best of our knowledge, `FedLN` is the first FL framework that learns 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.

2 Methodology

Problem Formulation: To define a label noise profile, we utilize two variables as defined in [11]: noise level (n_l), quantifying the percentage of mislabeled data in a given dataset, and noise sparsity (n_s), indicating the amount of confusion between classes. 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 $\mathcal{D}_c^m = \{(x_i, y_i^*)\}_{i=1}^{N_c^m}$ 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 $N = \sum_{i=1}^{M} N^m$ is the total number of samples present during training. Both $\mathcal{D}_c^m$ and $\mathcal{D}_n^m$ are not known apriori and 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.

Nearest Neighbor-based Correction (NNC): Learning from embeddings extracted from a self-supervised pre-trained model can help avoid poor generalization due to noisy labels, as embeddings' quality remain unaffected by the presence of noisy labels [24]. 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 $x_i \in \mathcal{D}^m$. We can then utilize a k-Nearest Neighbor (kNN) approach to identify and correct 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 labels using a majority voting mechanism, with random tie-breaking, as:

$$y_i^{vote} = \Phi(g, x_i) = \arg\max \sum_{j=1}^{k} \{y_j \in \mathcal{D}_k : |\text{sorted}\{\|g(x_i) - g(x_l)\|, \forall x_l \in \mathcal{D}\}| < k\}, \quad (1)$$

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, and k is the neighbourhood size of kNN. Afterwards, we perform the label correction process by utilizing the predicted label (with kNN) as the true label during the training phase, when a label mismatch between the predicted and current label is detected.

We note that NNC is performed locally on each client during the initialization phase of FL, where clients are required to hold a pre-trained model only for a single forward-pass. Additionally, NNC 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. Recently, several self-supervised pre-training approaches, such as [12, 14, 15], provide useful embeddings for broad spectrum of tasks. Further details on NNC can be found in Algorithm 1 of the Appendix.

Noise-aware Federated Averaging (NA-FedAvg): With NA-FedAvg, we aim to tackle the effect of noisy labels during the server-side aggregation process of naive *FedAvg* [9]. To this end, we

Table 1: Performance evaluation of `FedLN` on different datasets against a range of baselines. Average accuracy over three distinct trials on test set is reported. Supervised refers to standard FedAvg [9] training process, while LS and Bi-Temp denote the use of Label Smoothing [10] regularization and Bi-Tempered loss [1], respectively. CL corresponds to the Confidence Learning [11] technique. 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
		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
		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
		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
		NA-FedAvg	96.07	89.49	90.35	92.72	94.09	79.12	81.91	82.33	**80.37**

propose to utilize an estimation of client's noise level to perform a "*noise-aware*" aggregation, which considers both the number of samples and the number of noisy labels in the client's data. We use a scoring-based method, namely energy score [8], which can directly be applied to the outputs (or logits) of the neural network to estimate a per-client noise level. Each client applies the energy score fuction over their dataset in federated round r, where the locally trained model ($p_{\theta^r_m}$) is used to compute logits. We can acquire a scoring set, $\mathcal{S}_{\theta^r_m}$, which contains a score for each locally stored sample. Next, to differentiate between noisy and correctly labeled instances, we utilize a thresholding mechanism based on the scores obtained from the global aggregated model in the same federated round r (G^r). The threshold value τ_ν is computed from the ν^{th} percentile over the obtained score set, $\mathcal{S}_{\theta^r_G}$. With $\mathcal{S}_{\theta^r_m}$ and τ_ν computed, we can estimate a per-client noise level by computing the percentage of m^{th} client's local instances that are below the obtained τ_ν, as:

$$n_l^m = \frac{\sum_{i=1}^{N^m} u_{\tau_\nu}(s_i)}{N^m} \tag{2}$$

where $u_{\tau_\nu}(\cdot)$ is a "τ*-shifted*" Heaviside function, which produces 1 for all inputs above a threshold τ_ν. Finally, in the "*Noise-Aware*" FedAvg process, we introduce n_l^m in the relative impact of the m^{th} client on the construction of the global model G. It is important to highlight that even though *NA-FedAvg* produces a rough estimation of actual noise level present in clients' data by computing a pair of computationally inexpensive scoring sets in a single federated round, it can still be used effectively as a proxy to identify and exploit "*clean*" clients. Further details on *NA-FedAvg* can be found in Algorithm 1 of the Appendix.

3 Experiments

Datasets: We use publicly available datasets from both the vision and audio domains with their standard training/test splits. Specifically, we use the CIFAR-10 [6], FashionMNIST [20] and

PathMNIST[21] datasets, where the tasks of interests are object detection, clothes classification, and pathology reporting, respectively. Likewise, we use SpeechCommands (v2) dataset [18] for audio-based keyword spotting (12 classes in total). Additionally, we extend our evaluation to a real-world, human annotated version of CIFAR-10/100 [19] datasets, namely CIFAR-10N/100N, where label noise presents varied (or biased in some manner) patterns based on users' preferences.

Models Architectures: We use ResNet-20 [4] for the vision domain, while [17] was utilized for the considered audio recognition task. These models were chosen due to their relatively compact model size, which makes them ideal for on-device learning, where devices have medium to low computational resources. As a feature extractor for NNC, we use off-the-self pretrained models trained on large-scale datasets in an unsupervised manner. For vision tasks, we use ViT-B/32 from CLIP [12] and for audio we leverage TRILLsson (v3, EfficientNetv2-B3) [15], which has the same audio front-end as 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 for client's local data.

Experimental Setup: To simulate a federated environment, we use Flower framework [2] with FedAvg [9] to construct the global model from clients' local updates. The detailed federated parameters utilized in our experiments are presented in Table 2 of the Appendix. Furthermore, we fix ν=75% and k=100, which we determine during our initial exploration. Additionally, we randomly partitioned the datasets across the available clients in a non-overlapping fashion, after which we performed label noise injection using a noise matrix (unique among client) based on parameters n_l and n_s, similar to [11]. 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 aforementioned parameters.

Results: We performed experiments on all datasets for a diverse number of noisy profiles, varying both noise level (n_l) and sparsity (n_s), and compared `FedLN` performance with other considered baselines in Table 1. For a fair comparison, we utilized identical data partitioning and noisy injection schemes in all related experiments. From Table 1, we observe that `FedLN` can improve the model's performance compared to standard FedAvg across all datasets significantly. 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 baseline results, we notice that both LS [10] and Bi-Tempered loss [1] performance are inconsistent for a wide range of noise profiles, while CL [11] is more stable across diverse label noise settings with improvement not exceeding 4% on average across all tasks. On the contrary, `FedLN` performance is stable across majority of noise profiles and provides significant improvement in model's generalization capability. 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 the majority vote during kNN label estimation points to the noisy class; thus, amplifying the label noise. NA-FedAvg performance remains effective across all considered noise profiles, with no computational overhead and client-side modifications during the FL training process, which makes it ideal for clients with minimal computational and storage resources.

Next, we evaluated `FedLN` performance in the re-annotated versions of the CIFAR-10/100 datasets, namely CIFAR-10/100N [19], which contain a real-world human annotation error level of approximately 40% (n_l=40%); thus studying `FedLN` performance with label noise in-the-wild. We randomly distributed the data across clients; thus no clear group of "*clean*" clients is considered in this case, which makes the learning process even more challenging. From the results presented in Figure 1, we note that `FedLN` retains its 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 1 for n_l=40% for CIFAR10 dataset, while `FedLN` is able to improve the recognition rate by 8% on average compared to the standard FL process.

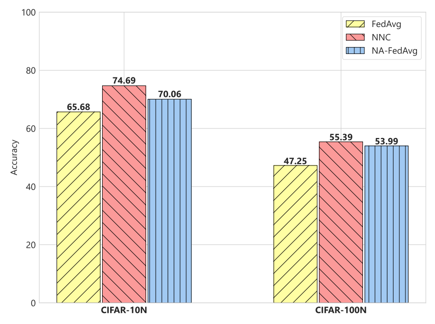

Figure 1: Evaluation of `FedLN` in real-world label noise patterns from CIFAR-10/100N. Average accuracy over three distinct trials on test set is reported for R=200/500 for CIFAR-10/100N, respectively. Federated parameters are set to M=30, E=1, q=80%, and σ=25%.

4 Conclusions

We propose a robust and effective framework to deal with label noise during learning on-device models, which operate in distinct phases of the FL process. Despite its simplicity, the model generalization we achieve is consistently superior to the considered baselines, while an evaluation of `FedLN` with in-the-wild label noise data showcases 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.

References

[1] E. Amid, M. K. Warmuth, R. Anil, and T. Koren. Robust bi-tempered logistic loss based on bregman divergences. 2019. doi: 10.48550/ARXIV.1906.03361. URL `https://arxiv.org/abs/1906.03361`.

[2] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, and N. D. Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[3] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. *CoRR*, abs/1811.03604, 2018. URL `http://arxiv.org/abs/1811.03604`.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. URL `https://arxiv.org/abs/1512.03385`.

[5] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency, 2017.

[6] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[7] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau. Federated learning for keyword spotting, 2019.

[8] W. Liu, X. Wang, J. D. Owens, and Y. Li. Energy-based out-of-distribution detection, 2020. URL `https://arxiv.org/abs/2010.03759`.

[9] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data, 2017.

[10] R. Müller, S. Kornblith, and G. Hinton. When does label smoothing help?, 2019. URL `https://arxiv.org/abs/1906.02629`.

[11] C. G. Northcutt, L. Jiang, and I. L. Chuang. Confident learning: Estimating uncertainty in dataset labels. 2019. doi: 10.48550/ARXIV.1911.00068. URL `https://arxiv.org/abs/1911.00068`.

[12] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[13] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. Snorkel. *Proceedings of the VLDB Endowment*, 11(3):269–282, nov 2017. doi: 10.14778/3157794.3157797. URL `https://doi.org/10.14778%2F3157794.3157797`.

[14] A. Saeed, D. Grangier, and N. Zeghidour. Contrastive learning of general-purpose audio representations. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3875–3879. IEEE, 2021.

[15] J. Shor and S. Venugopalan. Trillsson: Distilled universal paralinguistic speech representations. *arXiv preprint arXiv:2203.00236*, 2022.

[16] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee. Learning from noisy labels with deep neural networks: A survey, 2020. URL `https://arxiv.org/abs/2007.08199`.

[17] V. Tsouvalas, A. Saeed, and T. Ozcelebi. Federated self-training for semi-supervised audio recognition. *ACM Trans. Embed. Comput. Syst.*, feb 2022. ISSN 1539-9087. doi: 10.1145/3520128. URL `https://doi.org/10.1145/3520128`.

[18] P. Warden. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *ArXiv e-prints*, Apr. 2018. URL `https://arxiv.org/abs/1804.03209`.

[19] J. Wei, Z. Zhu, H. Cheng, T. Liu, G. Niu, and Y. Liu. Learning with noisy labels revisited: A study using real-world human annotations. *CoRR*, abs/2110.12088, 2021. URL `https://arxiv.org/abs/2110.12088`.

[20] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL `http://arxiv.org/abs/1708.07747`.

[21] J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, and B. Ni. Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *arXiv preprint arXiv:2110.14795*, 2021.

[22] S. Yang, H. Park, J. Byun, and C. Kim. Robust federated learning with noisy labels. *IEEE Intelligent Systems*, 37(2):35–43, 2022. doi: 10.1109/MIS.2022.3151466.

[23] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays. Applied federated learning: Improving google keyboard query suggestions, 2018.

[24] Z. Zhu, Z. Dong, and Y. Liu. Detecting corrupted labels without training a model to predict, 2021. URL `https://arxiv.org/abs/2110.06283`.

Appendix

Algorithm 1 `FedLN`: Federated learning under Label Noise. We develop two distinct approaches to deal with label noise in learning models from decentralized data, whereas FedAvg [9] 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 , NA-FedAvg

```
Server initialization of model $G$ with model weights $\theta_0^G$, $n_l^k$=0, $\forall k \in K$
**for** each client $k \in K$ **in parallel do**
    **for** $(x_i, y_i) \in \mathcal{D}^k$ **do**
        $\tilde{y_i} \leftarrow \Phi(g, x_i)$
    **end for**
**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**
        $\theta_r^m \leftarrow \theta_r^G$
        $\theta_{r+1}^m$, $(S_{\theta^G}^m, S_{\theta^m+1}^m)$ $\leftarrow$ ClientUpdate($\theta_r^m$,$r$,$n_l^m$)
    **end for**
    **if** $r = R_c$ **then** $n_l^m = \frac{1-\sum_{i=0}^{N_m} u_{\tau_\nu}\left(S_{\theta^m+1}\right)}{N_m}$ **end if**
    $\theta_{r+1}^G \leftarrow \sum_{m=1}^M \frac{N_m}{N} \theta_{r+1}^m$
    $\theta_{r+1}^G \leftarrow \sum_{m=1}^M (1 - n_l^m) \cdot \frac{N_m}{N} \theta_{r+1}^m$
**end for**
**procedure** ClientUpdate($\theta$, $r$, $n_l$)
    **for** epoch $e = 1, 2, \ldots, E$ **do**
        **for** batch $b \in \mathcal{D}^k$ **do**
            $\acute{\theta} \leftarrow \theta - \eta \nabla_\theta \left(\mathcal{L}_{CE}\left(y^{vote}, p_\theta\left(y|x_b\right)\right)\right)$
            $\acute{\theta} \leftarrow \theta - \eta \nabla_\theta \left(\mathcal{L}_{CE}\left(y, p_\theta\left(y|x_b\right)\right)\right)$
            **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**
```

Table 2: Primary Experiment Parameters used during evaluation. 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.

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%