Federated Algorithm With Bayesian Approach: Omni-Fedge

Sai Anuroop Kesanapalli [1] B. N. Bharath [2]

[1]Department of Computer Science and Engineering [2]Department of Electrical Engineering
Indian Institute of Technology Dharwad, Karnataka, India - 580011

Paper No. 4632
IEEE ICASSP 2021

Abstract

- Considered the problem of Federated Learning (FL) under non-i.i.d data setting
- Provided an improved estimate of the empirical loss at each node by using a weighted average of losses across nodes with a penalty term
- Assigned uneven weights to different nodes by taking a Bayesian approach to the problem where learning for each node is cast as maximizing the likelihood of a joint distribution of losses for a given neural network of a node, by using data across nodes
- Provided a PAC learning guarantee on the objective function which revealed that the true average risk is no more than the proposed objective and the error term
- Leveraged this guarantee to propose an algorithm called Omni-Fedge
- Using MNIST and Fashion MNIST data-sets, we showed that the performance of the proposed algorithm is significantly better than existing algorithms

Index Terms – Federated Learning, Neural Network, Bayesian Approach, Distributed Machine Learning, PAC Learning.

Introduction and Problem Setting

- We address the problem of improving FL performance with non-i.i.d data
- We consider a federated system with N edge-devices that communicate with one federating server (FS)
- We assume that the data points are independent but not necessarily identically distributed across edge-devices
- Further, we assume that the data at edge-device $i \in \{1, \ldots, N\}$ is sampled from a distribution $\mathcal{D}_i$
- Neural network weights are divided into two parts, viz, shared ($\boldsymbol{\theta}^{(sh)}$) and task-specific ($\boldsymbol{\theta}^{(i)}$)

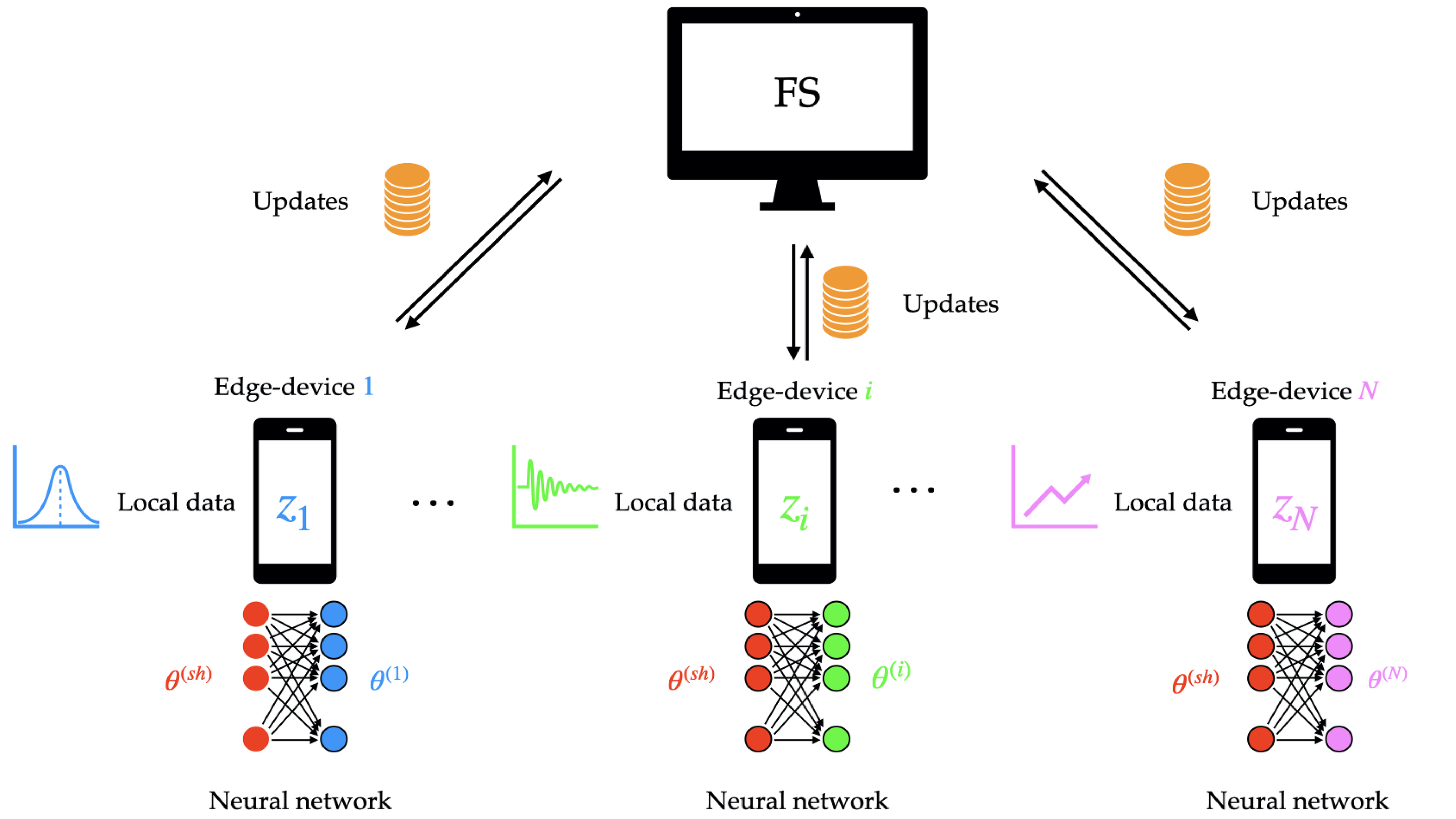


Figure: Federated Setup

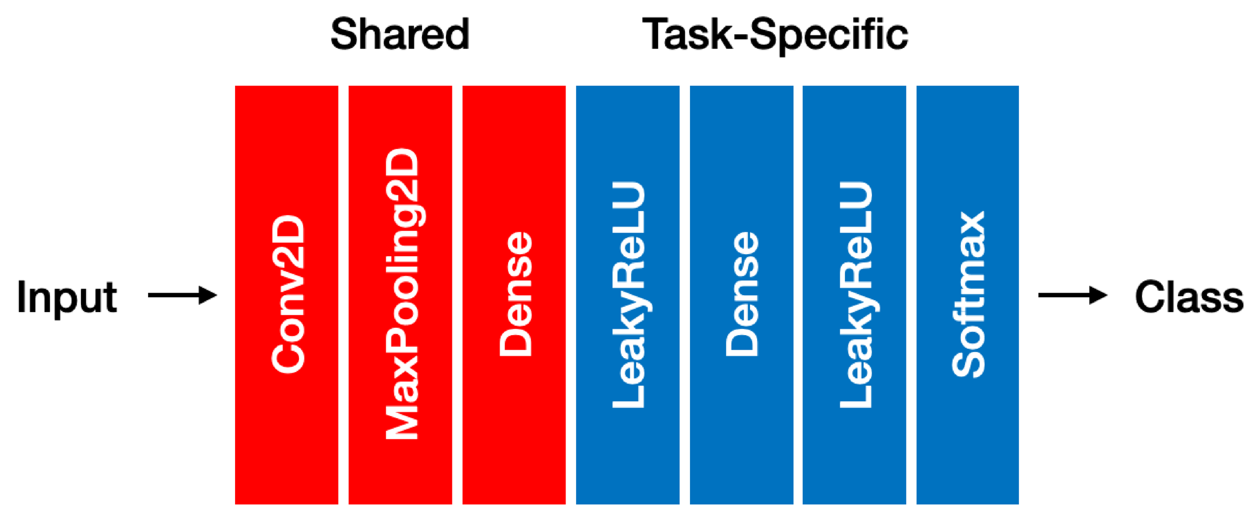


Figure: Neural Network

Motivation

$z_i \sim D_i$

non-IID local data sampled from distribution D_i

Finding a neural network that results in minimum average risk

$\min_{\theta} \mathbb{E}_{z_i \sim D_i}\{L_i(z_i, \theta)\}$

Minimise the expected value of the true loss with respect to the neural network weights

True loss unknown

$\min_{\theta} \hat{\mathbb{E}}L_i(z_i, \theta)$

Replacing the true loss with an empirical estimate of the true loss

Same neural network across all devices is not desirable

$\min_{\theta^{(i)},\theta^{(sh)}} \hat{\mathbb{E}}L_i(z_i, \theta^{(i)}, \theta^{(sh)})$

Splitting the neural network into shared and task-specific layers

Bayesian approach

$\min_{\omega_i,\theta^{(i)},\theta^{(sh)}} \sum_{j=1}^{N} \omega_{ij}\hat{\mathbb{E}}L_j(z_j, \theta^{(i)}, \theta^{(sh)}) - \sum_{j=1}^{N} \log \omega_{ij}$

Exploiting statistical similarity of data across edge-devices to get a better empirical estimate

Algorithm

Bayesian Approach

$$Q(\theta^{(i)}, \theta^{(sh)}) = (\Pi_{j=1}^{N}\omega_{ij})e^{-\sum_{j=1}^{N}\omega_{ij}\hat{\mathbb{E}}L_j(z_j,\theta^{(i)},\theta^{(sh)})}$$

Joint distribution of losses as assumed by edge-device i

$$P(\theta^{(i)}, \theta^{(sh)}) \quad (\neq Q(\theta^{(i)}, \theta^{(sh)}))$$

True joint distribution

$$\min_{\omega_i,\theta^{(i)},\theta^{(sh)}} \sum_{j=1}^{N} \omega_{ij}\hat{\mathbb{E}}L_j(z_j, \theta^{(i)}, \theta^{(sh)}) - \sum_{j=1}^{N} \log \omega_{ij}$$

Maximum likelihood estimate

Theoretical Guarantees

Definition: $\log - \exp$ **Complexity**

Let $\boldsymbol{\theta}^{(i)}$ and $\boldsymbol{\theta}^{(sh)}$ be a family of weights corresponding to task/edge specific and shared neural networks, respectively. The $\log - \exp$ complexity of the neural network with respect to the distribution $Q_{\boldsymbol{\theta}^{(i)},\boldsymbol{\theta}^{(sh)}}$ (Q for short) for $i = 1, 2, \ldots, N$ is defined as

$$\mathcal{R}_i(\boldsymbol{\theta}) := \log \mathbb{E}_Q \sup_{\boldsymbol{\theta}^{(i)},\boldsymbol{\theta}^{(sh)}} \frac{\exp\left\{\mathbb{E}_{z\sim\mathcal{D}_i} L_i(z, \boldsymbol{\theta}^{(i)}, \boldsymbol{\theta}^{(sh)})\right\}}{\Pi_{j=1}^{N}\hat{\mathbb{E}}L_j(z_j, \boldsymbol{\theta}^{(i)}, \boldsymbol{\theta}^{(sh)})}. \quad (1)$$

Theorem: PAC bound

For a given neural network $\boldsymbol{\theta}$, and the $\log - \exp$ complexity, the following bound holds with a probability of at least $1 - \delta$, $(\delta > 0)$

$$\inf_{\boldsymbol{\theta}} \mathbb{E}_{z_i\sim\mathcal{D}_i}\{L_i(z_i, \boldsymbol{\theta})\} \leq \inf_{\boldsymbol{\theta}^{(sh)}} \left[\texttt{Obj}_i(\boldsymbol{\theta}^{(sh)}) + \mathcal{R}_i(\boldsymbol{\theta}) + \sup_{\boldsymbol{\theta}^{(i)},\boldsymbol{\theta}^{(sh)},\boldsymbol{\omega}_i} \texttt{KL}(Q||P) + l_{max}\sqrt{\sum_{j=1}^{N}\frac{\omega_{ij}^2}{2n_j^2}\log\left(\frac{1}{\delta}\right)} - N\right],$$

where $\texttt{KL}(Q||P)$ is the KL-divergence between two joint distributions Q and P,

$$\texttt{Obj}_i(\boldsymbol{\theta}^{(sh)}) := \inf_{\boldsymbol{\omega}_i} \sum_{j=1}^{N}\left[\omega_{ij} \inf_{\boldsymbol{\theta}^{(i)}} \hat{\mathbb{E}}L_j(z_j, \boldsymbol{\theta}^{(i)}, \boldsymbol{\theta}^{(sh)}) - \log \omega_{ij}\right]. \quad (2)$$

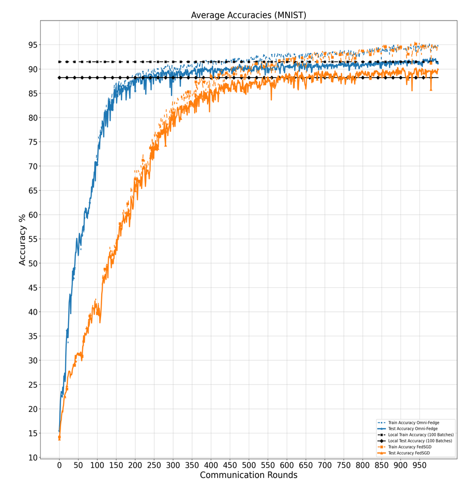
Please refer to the paper for proof

Algorithm

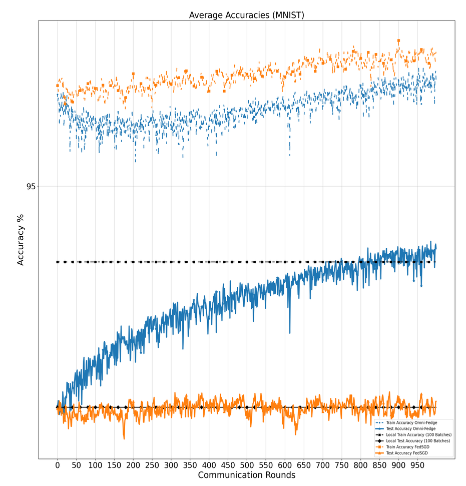
Algorithm 1: Omni-Fedge

```
1  Omni-Fedge():
2     INITIALIZE θ^sh and BROADCAST (BC) to all nodes
3     for t ∈ {1, 2, ......} do
4        for i = 1, 2, ..., N do
5           θ_t^(i) = arg min_θ^(i) ÊL_i(z_i, θ^(i), θ^(sh))
6           Each device i BCs θ_t^(i) to all other nodes
             l = 1, 2, ..., N through FS.
7           COMPUTE AND SEND ÊL_i(z_i, θ^(j), θ^(sh))
             to all nodes.
8           Minimize-Objective()
9              to get ω_i for all i.
10          At each node, COMPUTE
             Σ_{j=1}^N ω*_ji ∇_{θ_t^(sh)} ÊL_i(z_i, θ^(j), θ^(sh)) and
             BC it to all nodes through FS.
11          Perform GRADIENT UPDATE
             θ_{t+1}^(sh) := θ_t^(sh) − η^com γ_t^(i), where γ_t^(i) :=
             (1/N) ( Σ_{l=1}^N Σ_{j=1}^N ω*_jl ∇_{θ_t^(sh)} ÊL_l(z_l, θ^(j), θ^(sh)) )
12       GO TO step 3.
13 Minimize-Objective():
14    COMPUTE ω*_i =
       arg min_ω_i ( Σ_{j=1}^N ω_ij ÊL_j(z_j, θ^(i), θ^(sh)) −
       log Π_{j=1}^N ω_ij )
```

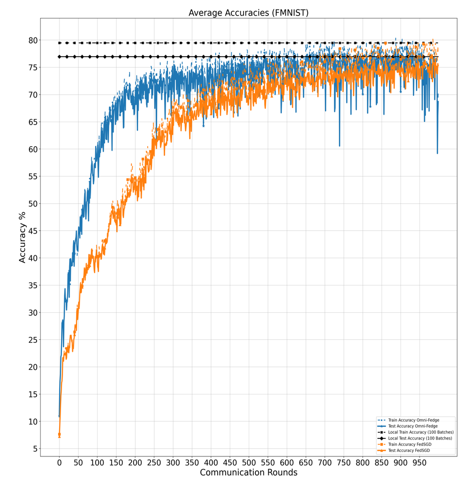
Experimental Results



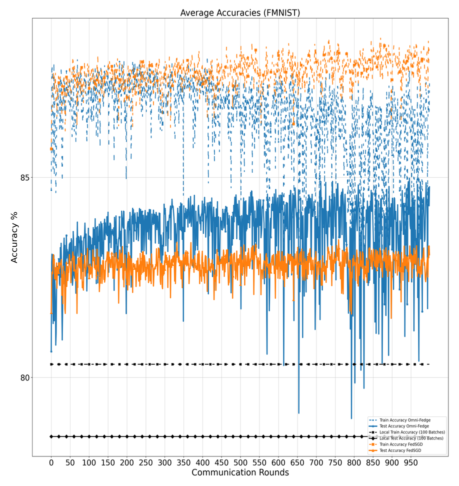
(a) MNIST non-i.i.d



(b) MNIST i.i.d



(c) FMNIST non-i.i.d



(d) FMNIST i.i.d

Figure: Plots of Average Accuracies vs Communication Rounds for Omni-Fedge and FedSGD

References

[1] Alex Kendall, Yarin Gal, and Roberto Cipolla.
Multi-task learning using uncertainty to weigh losses for scene geometry and semantics.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482--7491, 2018.

[2] Jakub Konečný, H Brendan McMahan, Daniel Ramage, and Peter Richtárik.
Federated optimization: Distributed machine learning for on-device intelligence.
arXiv preprint arXiv:1610.02527, 2016.

[3] Ozan Sener and Vladlen Koltun.
Multi-task learning as multi-objective optimization.
In *Advances in Neural Information Processing Systems*, pages 527--538, 2018.