OPTIMAL IMPORTANCE SAMPLING FOR FEDERATED LEARNING

Elsa Rizk, Stefan Vlaski, Ali H. Sayed

ABSTRACT

Federated learning involves a mixture of centralized and decentralized processing tasks, where a server regularly selects a sample of the agents and these in turn sample their local data to compute stochastic gradients for their learning updates. The sampling of both agents and data is generally uniform; however, in this work we consider non-uniform sampling. We derive optimal importance sampling strategies for both agent and data selection and show that under convexity and Lipschitz assumptions, non-uniform sampling without replacement improves the performance of the original FedAvg algorithm. We run experiments on a regression and classification problem to illustrate the theoretical results.

Index Terms— federated learning, importance sampling, asynchronous SGD, non-IID data, heterogeneous agents

1. INTRODUCTION AND RELATED MATERIAL

Most current machine learning applications work on large datasets, and sometimes this data is distributed accross several locations, making learning algorithms computationaly and communicatively expensive. Therefore, designers of these learning algorithms move toward stochastic implementations where only a subset of the data is used, such as in stochastic gradient descent (SGD). The most prominent selection schemes use uniform sampling. While it has proven to be a good approximator of the original solution, a greater advantage exists in non-uniform sampling that prioritizes samples and results in a faster learning process. For SGD, importance sampling attributes to each data sample a probability proportional to the gradient evaluated at that sample. Such probabilities result in faster convergence and reduced stochastic gradient variance [1–5].

A prominent machine learning architecture is federated learning [6] where non-IID data is distributed accross heterogeneous agents, and a server acts as a master node and controller of the learning process. The goal is to find a model that fits all the data across the agents. If we consider K agents each with a non-IID dataset $\{x_{k,n}\}$ of size N_k, where subscript k denotes the agent and subscript n indicates the sample number, the global optimization problem becomes as expressed in (1), where the cost function $P_k(\cdot)$ is defined as the sample average of some loss function $Q_k(\cdot; x_{k,n})$.

$$w^o \triangleq \underset{w}{\operatorname{argmin}} \frac{1}{K} \sum_{k=1}^{K} \left\{ P_k(w) \triangleq \frac{1}{N_k} \sum_{n=1}^{N_k} Q_k(w; x_{k,n}) \right\} \quad (1)$$

In standard federated learning, at each iteration i the server selects L agents, and we denote by $\mathcal{L}_i$ the set of indices of the chosen agents. In turn, each participating agent runs E_k epochs, during which it selects a mini-batch $\mathcal{B}_{k,i,e}$ of size B_k. The sampling occurs uniformly with replacement. Thus, unlike standard learning algorithms, two-level sampling exists in federated learning: first at the agent level and second at the data level.

In the literature on federated learning, researchers have studied the effects of non-uniform selection of agents, i.e., the selection of agents with unequal probabilities. The selection schemes can be split into those maximizing accuracy and those prioritizing fairness. Of the works maximizing accuracy, the authors in [7] introduce a new agent selection scheme called FedCS. The selection protocol consists of two steps. In the first step, the server asks the available agents for information on their resources. In the second step, based on the recieved information, the server selects the most number of agents that are capable of finishing an interation by a set deadline. Reference [8] broadens the previous work to include non-IID data, and they allow the server to act as one of the agents; the server collects from the selected agents some of their data and runs SGD on them. In [9], a non-uninform sampling scheme of agents is considered; the probability distribution of the sampling of agents is calculated by maximizing the inner product between the global and local gradients. They provide an approximate solution, since the actual calculations of the sampling probabilities is non-trivial. In [10, 11], the goal is to ensure fairness. In [10] the authors introduce agnostic federated learning where they model the data distribution as a non-uniform mixture of the local distributions. Thus, they solve a minimax problem to select the agents. Such a solution leads to minimizing the worst loss function. Building on the previous work, q-FFL [11] reweights the cost function by assigning higher weights to agents that have larger loss.

None of these works appears to consider the problem of incorporating importance samples at two levels: one for selecting the subset of agents and another for the agents to sample their own local data. In what follows, we implement non-uniform sampling of agents and data without replacement. To do so, we first introduce the notion of *normalized inclusion probability*. We define p_k as the normalized inclusion probability of agent k and $p_n^{(k)}$ as the normalized inclusion probability of data sample n of agent k [12]:

$$p_k \triangleq \frac{\mathbb{P}(k \in \mathcal{L}_i)}{L}, \qquad p_n^{(k)} \triangleq \frac{\mathbb{P}(n \in \mathcal{B}_{k,i,e})}{B_k}. \quad (2)$$

We briefly mention the difference between the *inclusion probabilities* and the *sampling probabilities*, since it is important to understand the distinction for the remainder of this work. The former indicates the probability of a sample being chosen after the sampling scheme is done, while the latter is used to choose the samples. The sampling probability is not affected by the sampling scheme, while the inclusion probability is. When sampling with replacement, the two are equal up to a multiplicative factor. When sampling without replacement, a non-trivial relationship exists between them. For instance, if we require to sample 2 out of 4 balls with sampling probabilitis denoted by π_n, the normalized inclusion probability of ball 1 is given by:

$$p_1 = \frac{1}{2} \sum_{n=2}^{4} \mathbb{P}\big(1 \text{ chosen on the } 1^{st} \text{ trial \& } n \text{ chosen on the } 2^{nd} \text{ trial}\big)$$

$$+ \mathbb{P}\big(n \text{ chosen on the } 1^{st} \text{ trial \& } 1 \text{ chosen on the } 2^{nd} \text{ trial}\big)$$

$$= \frac{1}{2} \sum_{n=2}^{4} \left(\pi_1 \frac{\pi_n}{1 - \pi_1} + \pi_n \frac{\pi_1}{1 - \pi_n} \right). \quad (3)$$

School of Engineering, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland (e-mail:{elsa.rizk, stefan.vlaski, ali.sayed}@epfl.ch).

In standard importance sampling, we optimize over the sampling probabilities, while in our implementation we optimize over the normalized inclusion probabilities, since we sample without replacement.

2. ALGORITHM DESCRIPTION

We introduce the ISFedAvg algorithm, which is a variation of the original FedAvg [6]. The goal of the original algorithm is to approximate the true gradient by some stochastic gradient. During each iteration i of FedAvg algorithm, agent k runs E_k steps of SGD:

$$\boldsymbol{w}_{k,e} = \boldsymbol{w}_{k,e-1} - \mu \frac{1}{B_k} \sum_{b \in \mathcal{B}_{k,i,e}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b}), \quad (4)$$

where $\boldsymbol{w}_{k,0} = \boldsymbol{w}_{i-1}$. After each agent finishes its epochs, the server combines the new parameter vector as follows:

$$\boldsymbol{w}_i = \frac{1}{L} \sum_{k \in \mathcal{L}_i} w_{k,E_k}. \quad (5)$$

The solution provided by FedAvg algorithm leads to the following estimation of the gradient at any given iteration i:

$$\frac{1}{L} \sum_{k \in \mathcal{L}_i} \frac{1}{B_k} \sum_{e=1}^{E_k} \sum_{b \in \mathcal{B}_{k,i,e}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b}). \quad (6)$$

However, as argued in [13], we modify the estimator by scaling it by the epoch size, so that the resulting estimator is unbiased. Furthermore, under more general sampling schemes, the unbiased estimator is in fact [1]:

$$\frac{1}{L} \sum_{k \in \mathcal{L}_i} \frac{1}{Kp_k} \frac{1}{E_k B_k} \sum_{e=1}^{E_k} \sum_{b \in \mathcal{B}_{k,i,e}} \frac{1}{N_k p_b^{(k)}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b}). \quad (7)$$

Thus, the new algorithm operates as follows: At each iteration i, the server selects L participating agents such that each agent's probability of inclusion is $\mathbb{P}(k \in \mathcal{L}_i) = Lp_k$. Then, each agent k initializes its model with the previous model sent by the server $\boldsymbol{w}_{i-1}$ and performs E_k SGD steps. During each step, agent k chooses its mini-batch such that a data point's probability of inclusion is $\mathbb{P}(b \in \mathcal{B}_{k,i,e}) = B_k p_b^{(k)}$. It then uses the following stochastic gradient to update its local model:

$$\frac{1}{Kp_k E_k B_k} \sum_{b \in \mathcal{B}_{k,i,e}} \frac{1}{N_k p_b^{(k)}} \nabla_{w^\mathsf{T}} Q_k(w; \boldsymbol{x}_{k,b}). \quad (8)$$

After all the agents are done and send back their new model, the server aggregates them according to (5). We call this new algorithm Importance Sampling FedAvg (ISFedAvg).

3. ALGORITHM CONVERGENCE

We now examine the convergence behavior of the proposed ISFedAvg implementation. The analysis will suggest ways to optimize the selection of the sampling probabilities which we discuss in the next section.

3.1. Modeling Conditions

Certain standard assumptions on the nature of the cost functions and the local optimizers must be made to ensure a tractable convergence analysis. Thus, in this work we assume convexity of the cost functions and smoothness of their gradients.

Algorithm 1 (Importance Sampling Federated Averaging)
> **initialize** w_0
> **for** each iteration $i = 1, 2, \cdots$ **do**
>> Select the set of participating agents $\mathcal{L}_i$ by sampling L times from $\{1, \ldots, K\}$ without replacement according to the probabilities p_k.
>> **for** each agent $k \in \mathcal{L}_i$ **do**
>>> **initialize** $\boldsymbol{w}_{k,0} = \boldsymbol{w}_{i-1}$
>>> **for** each epoch $e = 1, 2, \cdots E_k$ **do**
>>>> Find indices of the mini-batch sample $\mathcal{B}_{k,i,e}$ by sampling B_k times from $\{1, \ldots, N_k\}$ without replacement according to the probabilities $p_n^{(k)}$.
>>>> $\boldsymbol{g} = \dfrac{1}{Kp_k E_k B_k} \sum\limits_{b \in \mathcal{B}_{k,i,e}} \dfrac{1}{N_k p_b^{(k)}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b})$
>>>> $\boldsymbol{w}_{k,e} = \boldsymbol{w}_{k,e-1} - \mu \boldsymbol{g}$
>>> **end for**
>> **end for**
>> $\boldsymbol{w}_i = \dfrac{1}{L} \sum\limits_{k \in \mathcal{L}_i} \boldsymbol{w}_{k,E_k}$
> **end for**

Assumption 1. *The functions $P_k(\cdot)$ are $\nu-$strongly convex, and $Q_k(\cdot; x_{k,n})$ are convex, namely:*

$$P_k(w_2) \geq P_k(w_1) + \nabla_{w^\mathsf{T}} P_k(w_1)(w_2 - w_1) + \frac{\nu}{2} \|w_2 - w_1\|^2, \quad (9)$$

$$Q_k(w_2; x_{k,n}) \geq Q_k(w_1; x_{k,n}) + \nabla_{w^\mathsf{T}} Q_k(w_1; x_{k,n})(w_2 - w_1). \quad (10)$$

Also, the functions $Q_k(\cdot; x_{k,n})$ have $\delta-$Lipschitz gradients:

$$\|\nabla_{w^\mathsf{T}} Q_k(w_2; x_{k,n}) - \nabla_{w^\mathsf{T}} Q_k(w_1; x_{k,n})\| \leq \delta \|w_2 - w_1\|. \quad (11)$$

□

We further require the local optimizers $w_k^o = \operatorname{argmin}_w P_k(w)$ not to drift too far from the global optimizer. Such an assumption is necessary for good performance since without it, agents would have very different local models. The resulting average model would not perform well locally.

Assumption 2. *The distance of each local model w_k^o to the global model w^o is bounded uniformly:*

$$\|w_k^o - w^o\| \leq \xi. \quad (12)$$

□

3.2. Main Convergence Result

Instead of writing the recursion as in (4) and (5) and using the more general estimator (7), we can combine the multiple epochs and the aggregation into one SGD step:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \frac{1}{L} \sum_{k \in \mathcal{L}_i} \frac{1}{Kp_k E_k B_k} \sum_{e=1}^{E_k} \sum_{b \in \mathcal{B}_{k,i,e}} \frac{1}{N_k p_b^{(k)}}$$
$$\times \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b}). \quad (13)$$

For simplicity of notation, we introduce the stochastic gradient:

$$\widehat{\nabla_{w^\mathsf{T}} P_k}(w) \triangleq \frac{1}{E_k B_k} \sum_{e=1}^{E_k} \sum_{b \in \mathcal{B}_{k,i,e}} \frac{1}{N_k p_b^{(k)}} \nabla_{w^\mathsf{T}} Q_k(w; \boldsymbol{x}_{x,b}). \quad (14)$$

Then, by introducing $\widetilde{\boldsymbol{w}}_i = w^o - \boldsymbol{w}_i$ and defining two error terms:

$$\boldsymbol{s}_i \triangleq \frac{1}{L} \sum_{k \in \mathcal{L}_i} \frac{1}{Kp_\ell} \widehat{\nabla_{w^\top} P_k}(\boldsymbol{w}_{i-1}) - \frac{1}{K} \sum_{k=1}^{K} \nabla_{w^\top} P_k(\boldsymbol{w}_{i-1}),$$

$$(15)$$

$$\boldsymbol{q}_i \triangleq \frac{1}{L} \sum_{\ell \in \mathcal{L}_i} \frac{1}{Kp_\ell E_\ell B_\ell} \sum_{e=1}^{E_\ell} \sum_{b \in \mathcal{B}_{\ell,i,e}} \frac{1}{N_\ell p_b^{(\ell)}}$$
$$\times \left(\nabla_{w^\top} Q_k(\boldsymbol{w}_{\ell,e-1}; \boldsymbol{x}_{\ell,b}) - \nabla_{w^\top} Q_k(\boldsymbol{w}_{i-1}; \boldsymbol{x}_{\ell,b}) \right), \quad (16)$$

we can write the following error recursion:

$$\widetilde{\boldsymbol{w}}_i = \widetilde{\boldsymbol{w}}_{i-1} + \mu \boldsymbol{s}_i + \mu \boldsymbol{q}_i + \mu \frac{1}{K} \sum_{k=1}^{K} \nabla_{w^\top} P_k(\boldsymbol{w}_{i-1}). \quad (17)$$

We call the first error term *gradient error* (15), which quantifies the error due to the approximation of the gradient by using subsets of agents and data; it is the error that results when central SGD is ran at the server. We call the second error term *incremental error* (16), since it captures the error induced by running multiple epochs locally. We show that both errors have bounded second order moments; furthermore, we show that the gradient noise is zero-mean, which implies that the gradient estimate is unbiased. We summarize the results in the below two theorems.

Theorem 1 (Estimation of first and second order moments of the gradient noise). *The gradient noise $\boldsymbol{s}_i$ defined in (15) is zero-mean with bounded variance:*

$$\mathbb{E}\|\boldsymbol{s}_i\|^2 \leq \beta_s^2 \mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \sigma_s^2, \quad (18)$$

where:

$$\beta_s^2 \triangleq 3\delta^2 + \frac{1}{K^2} \sum_{k=1}^{K} \frac{1}{p_k} \left(\beta_{s,k}^2 + 3\delta^2 \right), \quad (19)$$

$$\sigma_s^2 \triangleq \frac{1}{K^2} \sum_{k=1}^{K} \frac{1}{p_k} \left\{ \sigma_{s,k}^2 + \left(3 + \frac{6}{E_k B_k} \right) \|\nabla_{w^\top} P_k(w^o)\|^2 \right\}, \quad (20)$$

$$\beta_{s,k}^2 \triangleq \frac{3\delta^2}{E_k B_k} \left(1 + \frac{1}{N_k^2} \sum_{n=1}^{N_k} \frac{1}{p_n^{(k)}} \right), \quad (21)$$

$$\sigma_{s,k}^2 \triangleq \frac{6}{E_k B_k N_k^2} \sum_{n=1}^{N_k} \frac{1}{p_n^{(k)}} \|\nabla_{w^\top} Q_k(w^o; x_{k,n})\|^2. \quad (22)$$

Proof. Proof omitted due to space limitations. $\square$

We observe that the gradient noise is controlled by the normalized inclusion probabilities. The $\sigma_{s,k}^2$ term captures the average local *data variability*, which is controlled by the batch size B_k. To overcome its effect, agents must increase their batch size. Furthermore, the second term in σ_s^2 captures the *model variability*, since it quantifies the suboptimality of the global model w^o. With Assumption 2, we can bound its influence on the convergence of the algorithm.

Theorem 2 (Estimation of the second order moments of the incremental noise). *The incremental noise $\boldsymbol{q}_i$ defined in (16) has bounded variance:*

$$\mathbb{E}\|\boldsymbol{q}_i\|^2 \leq O(\mu)\mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + O(\mu)\xi^2 + O(\mu^2)\sigma_q^2, \quad (23)$$

where:

$$\sigma_q^2 \triangleq \frac{1}{K} \sum_{k=1}^{K} \frac{3}{B_k N_k^2} \sum_{n=1}^{N_k} \frac{1}{p_n^{(k)}} \|\nabla_{w^\top} Q_k(w^o; x_{k,n})\|^2, \quad (24)$$

and the $O(\cdot)$ terms depend on epoch sizes, local convergence rates, total number of data samples, number of agents, Lipschitz constant, and data and agent inclusion probabilities.

Proof. Proof omitted due to space limitations. $\square$

Similarly to the bound on the variance of the gradient noise, we observe a *data variablity* term σ_q^2 and a *model variability* term ξ^2. However, here the effect of the latter is greater than that of the former, since it is multiplied by an $O(\mu)$ term as opposed to $O(\mu^2)$.

Thus, using the above results, we show under general probability sampling schemes that the ISFedAvg algorithm converges to an $O(\mu)-$neighbourhood of the global model. The result is summarized in the below theorem.

Theorem 3 (Mean-square-error convergence of federated learning under importance sampling). *Consider the iterates $\boldsymbol{w}_i$ generated by the importance sampling federated averaging algorithm. For sufficiently small step-size μ, it holds that the mean-square-error converges exponentially fast:*

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq O(\lambda^i) + O(\mu)\left(\sigma_s^2 + \xi^2\right) + O(\mu^3)\sigma_q^2, \quad (25)$$

where $\lambda = 1 - O(\mu) + O(\mu^2) \in [0,1)$.

Proof. Proof omitted due to space limitations. $\square$

In fact, we observe that the only significant contribution from the incremental noise is the ξ term that captures the model variability accross the agents.

4. IMPORTANCE SAMPLING: PROBABILITY DERIVATION

Since agents and data are heterogeneous, increasing the chances of a subset being choosen is generally beneficial. In the previous section, we established a performance bound under general probabilities; the probabilities appear in the σ_s^2 term. Thus, to improve the performance bound we will choose these probabilities by minimizing σ_s^2.

4.1. Optimal Probabilities

Importance sampling occurs at two levels: at the agent level when selecting the data, and at the server level when selecting the agents. During each epoch of every iteration, agent k has to select a mini-batch in accordance with its normalized inclusion probabilities $p_n^{(k)}$. To calculate the optimal probabilities, we minimize $\sigma_{s,k}^2$, and we find:

$$p_n^{(k),o} \triangleq \frac{\|\nabla_{w^\top} Q_k(w^o; x_{k,n})\|}{\sum_{m=1}^{N_k} \|\nabla_{w^\top} Q_k(w^o; x_{k,m})\|}, \quad (26)$$

which is proportional to the data variability. The more uniform the data is, the closer the distribution is to a uniform distribution. Similarly, during each iteration the server selects a subset of participating agents. Thus, we minimize σ_s^2, and find the optimal probabilities:

$$p_k^o \triangleq \frac{\sqrt{\sigma_{s,k}^2 + \alpha_k \|\nabla_{w^\top} P_k(w^o)\|^2}}{\sum_{\ell=1}^{K} \sqrt{\sigma_{s,\ell}^2 + \alpha_\ell \|\nabla_{w^\top} P_\ell(w^o)\|^2}}, \quad (27)$$

where we define the constants $\alpha_k = \left(3 + \frac{6}{E_k B_k}\right)$. The probabilities are proportional to the two variability terms, data and model. Thus, the more homogeneous the agents are or the local data, the closer the probabilites are to a uniform distribution. Furthermore, both expressions of the probabilities are proportional to the gradient. Thus, we attribute larger probabilities to agents and samples with higher gradients resulting in a steeper gradient step during an iteration and converging faster.

4.2. Practical Problems

The expressions of the optimal probabilities are in terms of the true model w^o, which we do not know. Furthermore, they require the true gradient, which is costly to calculate. Thus, we modify the expressions by writing them in terms of the mini-batch gradient evaluated at the previous model $\boldsymbol{w}_{i-1}$. In addition, while each agent has access to all of its data and can calculate the denominator in (26), the server does not have acces to all the agents and cannot calculate the denominator in (27). Therefore, we suggest the following scheme to calculate the probabilities: during the initial iteration we assume the normalized inclusion probability of the agents is uniform. Then, during the succeeding iterations, the server updates the probabilities of the participating agents after they have sent their current stochastic gradients:

$$\widehat{p}_k^o = \frac{\sqrt{\sigma_{s,k}^2 + \alpha_k \left\| \widehat{\nabla_{w^\top} P_k}(\boldsymbol{w}_{i-1}) \right\|^2}}{\sum_{\ell \in \mathcal{L}_i} \sqrt{\sigma_{s,\ell}^2 + \alpha_\ell \left\| \widehat{\nabla_{w^\top} P_\ell}(\boldsymbol{w}_{i-1}) \right\|^2}} \left(1 - \sum_{\ell \in \mathcal{L}_i^c} \widehat{p}_\ell^o \right).$$

(28)

The multiplicative factor is to ensure all inclusion probabilities accross the agents sum to 1. Similarly, we implement the same scheme to approximate the normalized data inclusion probabilities at the agents:

$$\widehat{p}_n^{(k),o} = \frac{\| \nabla_{w^\top} Q_k(\boldsymbol{w}_{i-1}; x_{k,n}) \|}{\sum_{b \in \mathcal{B}_{k,i,e}} \| \nabla_{w^\top} Q_k(\boldsymbol{w}_{i-1}; x_b) \|} \left(1 - \sum_{b \in \mathcal{B}_{k,i,e}^c} \widehat{p}_b^{(k),o} \right).$$

(29)

Given the inclusion probabilities, there are many works in the literature that devise schemes to deduce from them what the sampling probabilities should be. We choose to implement the scheme introduced in [14]. If given $k = 1, 2, \cdots, K$ agents to choose from with normalized inclusion probabilities p_k, and if we require to sample L agents, we first introduce the *progressive totals* $\Pi_k = \sum_{\ell=1}^{k} L p_\ell$ and set $\Pi_0 = 0$. Next, we uniformly choose at random a variable $d \in [0, 1)$. Finally, we select the L agents such that $\Pi_{k-1} \leq d + i \leq \Pi_k$ for some $i \in [0, L-1]$.

5. EXPERIMENTAL SECTION

We test the algorithm on a regression problem with simulated data and a quadratic risk function, and on a classification problem with simulated data and logistic risk. Starting with the regression problem, we consider $K = 300$ agents, for which we generate $N_k = 100$ data points $\{\boldsymbol{u}_{k,n}, \boldsymbol{d}_k(n)\}$ according to the following model:

$$\boldsymbol{d}_k(n) = \boldsymbol{u}_{k,n} \boldsymbol{w}^\star + \boldsymbol{v}_k(n),$$

(30)
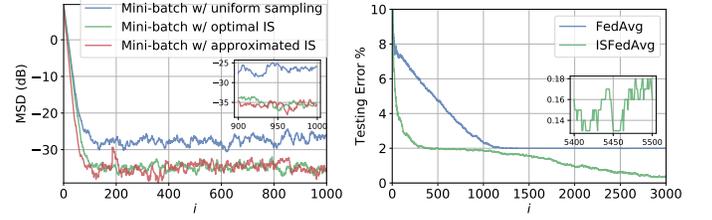
Fig. 1: Left: MSD plots of the regression problem. Right: percent testing error of the classification problem.

where $\boldsymbol{w}^\star$ is some generating model and $\boldsymbol{v}_k(n)$ Guassian noise independent of $\boldsymbol{u}_{k,n}$. The local risk is given by:

$$P_k(w) = \frac{1}{N_k} \sum_{n=1}^{N_k} \| \boldsymbol{d}_k(n) - \boldsymbol{u}_{k,n} w \|^2 + 0.001 \|w\|^2.$$

(31)

We set $L = 6$ and choose epoch sizes $E_k \in [1, 5]$ and batch sizes $B_k \in [1, 10]$ randomly. We test the performance by calculating the mean-square-deviation (MSD) at each iteration:

$$\text{MSD} = \| \boldsymbol{w}_i - w^o \|^2,$$

(32)

where w^o is explicitly calculated for the risk (31):

$$w^o = \left(\widehat{R}_u + \rho I \right)^{-1} \widehat{R}_u w^\star + \left(\widehat{R}_u + \rho I \right)^{-1} \widehat{r}_{uv},$$

(33)

$$\widehat{R}_u \triangleq \frac{1}{K} \sum_{k=1}^{K} \frac{1}{N_k} \sum_{n=1}^{N_k} \boldsymbol{u}_{k,n}^\top \boldsymbol{u}_{k,n},$$

(34)

$$\widehat{r}_{uv} \triangleq \frac{1}{K} \sum_{k=1}^{K} \frac{1}{N_k} \sum_{n=1}^{N_k} \boldsymbol{v}_k(n) \boldsymbol{u}_{k,n}.$$

(35)

We compare 3 algorithms: FedAvg, ISFedAvg with true probabilities (27)–(26), and ISFedAvg with approximate probabilities (28)–(29). We fix the step size $\mu = 0.01$, and run each algorithm 100 times and average the results to get the curves in Figure 1 on the left. We observe that both importance sampling schemes outperform the standard algorithm. Furthermore, the approximate probabilities do not degrade the performance of ISFedAvg algorithm.

We then solve a logistic regression problem on non-IID generated data $\{\boldsymbol{h}_{k,n}, \gamma_k(n)\}$ for $K = 100$ agents with varying $N_k \in [20, 100]$ and dimension $M = 2$. Each agent's features $\boldsymbol{h}_{k,n}$ are sampled from a normal distribution $\mathcal{N}(\mu_k, \sigma_k)$, where the variance and mean are choosen randomly. Then, the labels are calculated as follows:

$$\gamma_k(n) = \text{sign}(\boldsymbol{h}_{k,n}^\top \boldsymbol{w}_k^\star),$$

(36)

where $\boldsymbol{w}_k^\star$ are some random generating models that do not drift too far apart accross agents. We also generate 100 test samples. We compare the testing error for the FedAvg and ISFedAvg algorithm, and we observe that importance sampling improves the performance from 2% to almost 0% (Figure 1 on the right).

6. CONCLUSION

This work introduces a two-level importance sampling scheme to federated learning: one at the level of agents and another at the level of data. The theoretical results establish convergence of the new algorithm. Optimal inclusion probabilities are derived and then approximated to overcome the practical issues. Experiments illustrate the superior performance of the proposed sampling techniques.

7. REFERENCES

[1] K. Yuan, B. Ying, S. Vlaski, and A. H. Sayed, "Stochastic gradient descent with finite samples sizes," in *Proc. International Workshop on Machine Learning for Signal Processing (MLSP)*, Salerno, Italy, 2016, pp. 1–6.

[2] G. Alain, A. Lamb, C. Sankar, A. Courville, and Y. Bengio, "Variance reduction in SGD by distributed importance sampling," *arXiv:1511.06481*, 2015.

[3] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm," in *Proc. Advances in Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 1017–1025.

[4] P. Zhao and T. Zhang, "Stochastic optimization with importance sampling for regularized loss minimization," in *Proc. International Con-ference on Machine Learning (ICML)*, Lille, France, 2015, pp. 1355–1363.

[5] S. U. Stich, A. Raj, and M. Jaggi, "Safe adaptive importance sampling," in *Proc. Advances in Neural Information Processing Systems*, Long Beach, California, USA, 2017, pp. 4381–4391.

[6] H. B. McMahan, E. Moore, D. Ramage, and S. Hampson, "Communication-efficient learning of deep networks from decentralized data," *Proc. International Conference on Artificial Intelligence and Statistics*, vol. 54, pp. 1273–1282, 20–22 April 2017.

[7] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE International Conference on Communications (ICC)*, Shanghai, China, 2019, pp. 1–7.

[8] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, "Hybrid-fl for wireless networks: Cooperative learning mechanism using non-iid data," in *Proc. IEEE International Conference on Communications (ICC)*, Dublin, Ireland, 2020, pp. 1–7.

[9] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang, and H. V. Poor, "Fast-convergent federated learning," *arXiv:2007.13137*, 2020.

[10] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. International Con-ference on Machine Learning (ICML)*, vol. 97, Long Beach, California, USA, 09–15 Jun 2019, pp. 4615–4625.

[11] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *Proc. International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020, pp. 1–27.

[12] D. G. Horvitz and D. J. Thompson, "A generalization of sampling without replacement from a finite universe," *Journal of the American statistical Association*, vol. 47, no. 260, pp. 663–685, 1952.

[13] E. Rizk, S. Vlaski, and A. H. Sayed, "Dynamic federated learning," in *Proc. IEEE SPAWC*, Atlanta, Georgia, USA, 26–29 May 2020, pp. 1–5.

[14] H. O. Hartley and J. N. K. Rao, "Sampling with unequal probabilities and without replacement," *Ann. Math. Statist.*, vol. 33, no. 2, pp. 350–374, 06 1962.