Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information

Divyansh Jhunjhunwala 1 Shiqiang Wang 2 Gauri Joshi 1

Abstract

Standard federated learning (FL) algorithms typically require multiple rounds of communication between the server and the clients, which has several drawbacks including requiring constant network connectivity, repeated investment of computation resources and susceptibility to privacy attacks. One-Shot FL is a new paradigm that aims to address this challenge by enabling the server to train a global model in a single round of communication. In this work, we present FedFisher, a novel algorithm for one-shot FL that makes use of the Fisher information matrices computed at the local models of clients, motivated by a Bayesian perspective of FL. First, we theoretically analyze FedFisher for two-layer overparameterized ReLU neural networks and show that the error of our one-shot FedFisher global model becomes vanishingly small as the width of the neural networks and amount of local training at clients increases. Next we propose practical variants of FedFisher using the diagonal Fisher and K-FAC approximation for the full Fisher and highlight their communication and compute efficiency for FL. Finally, we conduct extensive experiments on various datasets, which show that these variants of FedFisher consistently improve over several competing baselines.

1. Introduction

Data collection and storage is becoming increasingly decentralized, both due to the proliferation of smart devices as well as privacy concerns stemming from transferring and storing data at a centralized location. Federated Learning (FL) is a framework that is designed to learn the parameters \( w \in \mathbb{R}^d \) of a model \( f(w, \cdot) \) on decentralized data that is distributed across a network of clients under the supervision of a central server (1; 2; 3). Our focus is particularly on the case where \( f(w, \cdot) \) is a neural network as is usually the case in practice. Formally, we formulate FL as the following distributed optimization problem:

\[
\min_{w \in \mathbb{R}^d} \left\{ L(w) := \frac{1}{M} \sum_{i=1}^{M} L_i(w) \right\}
\]

where,

\[
L_i(w) = \frac{1}{n} \sum_{(x_{ij}, y_{ij}) \in D_i} \ell(f(w, x_{ij}), y_{ij}).
\]

Here, \( M \) is the number of clients, \( D_i \) is the \( i \)-th client’s local dataset consisting of input-label pairs \( \{(x_{ij}, y_{ij})\}_{j=1}^{n_i} \) where \( x \in \mathbb{R}^p \) is the input and \( y \in \mathbb{R}^C \) is the label, and \( n = |D_i| \) is the dataset size. For simplicity, we consider the case where clients have equal amounts of data; our algorithm and analysis can easily be extended to the case where client objectives are unequally weighted. The loss function \( \ell(\cdot, \cdot) \) penalizes the difference between the prediction of the model \( f(w, x) \) and true label \( y \). We use \( D = \{D_i\}_{i=1}^{M} \) to denote the collection of data across all clients and \( N = Mn \) to denote the total data samples across clients.

Most standard FL algorithms such as FedAvg (4) and FedProx (5) require multiple rounds of communication between clients and server in order to train a global model in federated settings. However, a multi-round approach has several drawbacks for clients, including the need to frequently connect with the server, repeated investment of computational resources to update the global model in every round and increased susceptibility to privacy attacks. To overcome these drawbacks, a recent line of work has focused on the paradigm of one-shot FL which aims to learn the parameters of the global model in a single round of communication between clients and the server. Existing works for one-shot FL can be broadly split into two categories: (i) Knowledge Distillation (KD) methods and (ii) Neuron Matching (NM) methods. KD methods treat the collection of client models as an ensemble and propose to distill the knowledge from this ensemble into a single global model. However, to perform the distillation step, these works either assume that the server has access to an auxiliary public dataset (6; 7; 8), or propose to train generative models such as GANs (9; 10) or variational autoen-
Motivated by the limitations of multi-round FL and current approaches for one-shot FL, we ask the question: Can we devise a one-shot FL method that is simultaneously communication and computation efficient, privacy-preserving and has good practical performance? Furthermore, can we provide theoretical guarantees for such a method?

Our Contributions. In this work, we take a step towards providing an affirmative answer to both of the questions formulated above. To do so, we use the idea that Equation (2) can alternatively be reformulated as a posterior inference problem, specifically finding the mode of a global posterior (19; 20). Some highlights of our contribution are as follows.

- We propose FedFisher and show how the problem of finding a mode of the global posterior can be solved approximately in a one-shot manner, using the local models at each client and the Fisher information matrices computed at these local models (Section 2).
- For the case of overparameterized two-layer neural networks, we show that when we utilize the full Fisher information in FedFisher, the error of our one-shot global model becomes vanishingly small as the width of the neural networks and amount of local training at clients increases (Section 3).
- We propose practical variants of FedFisher using the diagonal and K-FAC approximation (21) for the full Fisher which we term as FedFisher( diag) and FedFisher(K-FAC). We evaluate FedFisher( diag) and FedFisher(K-FAC) on a range of one-shot FL tasks using deep neural networks and show that they give a consistent 5 – 10% accuracy improvement compared to competing KD and NM one-shot baselines (Section 4).


We begin by stating the following standard assumption on the loss function $\ell(\cdot, \cdot)$, which is true for most common loss functions such as the squared loss and cross entropy loss.

Assumption 1. Given $z = f(w, x)$, we assume that $\ell(z, y)$ is the negative log likelihood of $y$ under some exponential family probabilistic model, i.e., $\ell(z, y) = -\log P(y|z)$ where $P(y|z) = h(y) \exp(z^\top T(y) - A(z))$ and $h(y), A(z)$ are some real-valued functions.

Let us define the likelihood for a data point $(x, y)$ for a given $w$ as $P((x, y)|w) = q(x) \exp(-\ell(f(w, \mathbf{x}_i), y)$ where $q()$ is some prior on $x$, independent of $w$. We can now adopt a Bayesian viewpoint and find the maximum a posteriori probability (MAP) estimate, i.e., find $w$ where the posterior likelihood $P(w|D) \propto \prod_{i=1}^n P(w|D_i)$. Furthermore, modes of the global posterior coincide with the optimla of the FL objective in Equation (2), i.e., $\arg\max_{w \in \mathbb{R}^d} P(w|D) = \arg\min_{w \in \mathbb{R}^d} L(w)$.

Proposition 2 tells us that as long as clients compute and send their local posteriors $P(w|D_i)$ to the server, no further server-client communication is needed to find the global MAP estimate or equivalently a minimizer to our FL objective, giving us a one-shot algorithm. However, doing so is challenging since $P(w|D_i)$ typically does not have an analytical expression. To get a tractable solution, we propose to use some approximate inference techniques.

Mode of Local Posterior. To apply the approximate inference techniques detailed below, clients need to first compute $\tilde{w}_i \approx \arg\max_w P(w|D_i)$, an estimate for the mode of their local posterior under the flat prior. Note that this corresponds to a minimizer of $L_i(w)$ and therefore $\tilde{w}_i$ can be obtained using standard GD optimization.

Laplace Approximation for Local Posterior. Using a second order Taylor expansion around $\tilde{w}_i$, we get the following approximation for the log-posterior at the $i$-th client:

$$\log P(w|D_i) \approx \log P(\tilde{w}_i|D_i) - \frac{\lambda}{2} (w - \tilde{w}_i)^\top H_i(w - \tilde{w}_i)$$

where we additionally use $\nabla \log P(w|D_i)|_{w=\tilde{w}_i} \approx 0$. Here $H_i = -\frac{1}{n} \nabla^2 \log P(w|D_i)|_{w=\tilde{w}_i}$ is the Hessian of the negative log-posterior at client $i$ computed at $\tilde{w}_i$.

Approximating Hessian with Fisher. The Fisher information matrix $F_i$ (herein referred to as ‘the Fisher’) of the local model $\tilde{w}_i$ at client $i$ is defined as follows:

$$F_i = \frac{1}{n} \sum_{j=1}^n \nabla_{x_i} \left[ \nabla \log P(y|x_i, w) \nabla \log P(y|x_i, w)^\top \right]|_{w=\tilde{w}_i}$$

Now if $\tilde{w}_i$ fits the data at client $i$ perfectly, i.e., $f(\tilde{w}_i, \mathbf{x}_{ij}) = y_{ij}$ $\forall j \in [n]$, then it can be shown that $H_i = F_i$ (22; 23). Unlike the Hessian, the Fisher is guaranteed to be positive semi-definite, a condition required for tractable inference at the global server. However, communi-


Algorithm 1 FedFisher

1: **Input:** initial \( w_0 \), no. of iterations \( K, T \), client and server step sizes \( \eta \) and \( \eta_S \) respectively
2: **Global server does:**
3: Communicate \( w_0 \) to all clients;
4: **Clients** \( i \in [N] \) in parallel do:
5: Set \( w_i^{(0)} \leftarrow w_0 \);
6: For \( k = 0, \ldots, K - 1 \) iterations:
7: \( w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta \nabla L_i(w_i^{(t)}) \);
8: Set \( \hat{w}_i \leftarrow w_i^{(K)} \);
9: Compute \( \hat{F}_i \); // Approximation to true Fisher
10: Communicate \( \hat{w}_i \) and \( \hat{F}_i \) to the server;
11: **Global server does:**
12: Set \( w^{(0)} = \sum_{i=1}^{M} \hat{w}_i / M \);
13: For \( t = 0, \ldots, T - 1 \) iterations:
14: \( w^{(t+1)} \leftarrow w^{(t)} - \eta_S \left( \sum_{i=1}^{M} \hat{F}_i w^{(t)} - \sum_{i=1}^{M} \hat{F}_i \hat{w}_i \right) \).

3. Theoretical Analysis for Two Layer Overparameterized ReLU Neural Network

In this section we present a bound on the global training loss i.e., \( L(w^*) \), when \( f(w, \cdot) \) is a two layer overparameterized ReLU neural network. We begin by modeling a two-layer ReLU NN as follows,

\[
f(w, x) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r \sigma(x^\top w_r).
\]

Here \( m \) is the number of hidden nodes in the first layer, \( \{w_r\}_{r=1}^{m} \) are the weights of the first layer, \( \{a_r\}_{r=1}^{m} \) are the weights of the second layer and \( \sigma(x) = \max(x, 0) \) is the ReLU function. Similar to (24), we consider the \( a_r \)'s to be fixed beforehand (initialized to be +1 or 0 uniformly at random) and only consider the case where \( w_r \)'s are trained.

**Definition 1.** (Minimum eigenvalue of Gram Matrix (24))

For \( (x, y) \in \mathcal{D} \), define matrix \( H^w \in \mathbb{R}^{N \times N} \) as \( H^w_{i,j} = \mathbb{E}_{x \sim \mathcal{D}(x)} \mathbb{E}_{y \sim \mathcal{Y}(y)} [y_i y_j \| x_i x_j \| \leq C \| x_i \| \leq C \] \) and \( \lambda_0 = \lambda_{\min}(H^w) \).

**Assumption 2.** (Data normalization and uniqueness) For any \( (x, y) \in \mathcal{D} \), we have \( \| x \|_2 = 1 \) and \( |y| \leq C \), where \( C \) is some positive constant. For any \( (x, y), (x', y') \in \mathcal{D} \) we have \( \| x - x' \|_2 \geq \delta > 0 \)

**Assumption 3.** (Full Fisher) Clients compute and send their full Fisher, i.e., \( \hat{F}_i = F_i \).

Definition 1 and Assumption 2 are standard in NN optimization literature (24; 25; 26; 27). We note that most commonly used approximations such as the diagonal and K-FAC are primarily empirically motivated and have little theoretical understanding. Therefore, we use Assumption 3 to simplify our analysis and focus on bounding the error introduced by Equation (3) and suboptimality of \( \hat{w}_i \), i.e., \( \nabla L_i(\hat{w}_i) \neq 0 \).

**Theorem 1.** Under Assumptions 2, 3, for \( m \geq \text{poly}(N, \lambda_0^{-1}, \delta^{-1}, \kappa^{-1}) \), \( \ell(z, y) = \frac{1}{2}(z - y)^2 \) and i.i.d Gaussian initialization weights of \( w_0 \) as \( w_{0,r} \sim \mathcal{N}(0, \kappa I) \), and initializing \( a_r = \{-1, 1\} \) with probability \( 1/2 \) for all \( r \in [m] \), for step sizes \( \eta \in \mathcal{O}(\lambda_0/N^2) \), \( \eta_S \in \mathcal{O}(\lambda_0/N^2) \) and for a given failure probability \( \delta \in (0, 1) \), the following is true with probability \( 1 - \delta \)

\[
L(w^*) \leq \mathcal{O} \left( (1 - \eta \lambda_0/2) K N \delta \right) + \mathcal{O} \left( \frac{2(1 - \eta \lambda_0/2) \text{poly}(N, \kappa^{-1}, \lambda_0^{-1}, \delta^{-1})}{m} \right).
\]

**Takeaways.** Theorem 2 shows that for sufficiently wide networks, \( L(w^*) \) can be decomposed into the local optimization error, which measures how well \( \hat{w}_i \) fits the data at
the $i$-th client, and the error introduced by the Laplace approximation in Equation (3). While the overall error always decreases as the width $m$ increases, there is a trade-off in the number of local optimization steps $K$. In particular, a larger $K$ reduces the local optimization error but increases the Laplace error as each local model $\theta_i$ drifts further away from $\mathbf{w}^*$. Note by setting $m$ and $K$ appropriately, the error can be made arbitrarily small. In contrast, a similar bound for FedAvg has a fixed $O(1)$ bound on the error (28; 29).

4. Practical Variants of FedFisher and Experiments

Two of the most popular approximations for the Fisher are the diagonal Fisher and the Kronecker Factored Approximate Curve (K-FAC). Using these approximations as a substitute for the true Fisher in Equation (5), we get two practical variants of FedFisher which we term as FedFisher(Diag) and FedFisher(K-FAC) respectively. We highlight the computation and communication efficiency of these variations along with their privacy properties in Appendix C. In particular, we ensure that the communication cost of these variants matches that of FedAvg. Here we evaluate the performance of FedFisher(Diag) and FedFisher(K-FAC) in comparison to state-of-the-art (SOTA) one-shot FL baselines across a range of image recognition tasks in a FL setting. The datasets that we use are (i) MNIST (30), (ii) FashionMNIST (31) (iii) SVHN (32) (iv) CIFAR-10 (33) and (v) CINIC-10 (34).

Baselines. FedAvg is the de-facto baseline in all our experiments. The other baselines that we compare with are (i) PFNM (ii) OTFusion (12) and (iii) DENSE (9), PFNM and OTFusion are SOTA neuron matching methods that first permute the weights of the local models and then average them, instead of direct averaging. DENSE, is a SOTA method for one-shot FL using data-free knowledge distillation that uses GANs to artificially generate data for distillation at the server. We avoid comparing with baselines that need auxiliary data or maintain an ensemble of local models (35; 36) at the server to ensure fairness of comparison.

Models and Experimental Setup. For MNIST and FashionMNIST we use the MLPNet architecture proposed in (12); for SVHN and CIFAR10 we use a CNN model proposed in (18); for CINIC-10 we use ResNet-18 (37). We note that PFNM does not currently support ResNet18 architecture. To simulate data heterogeneity among the client datasets, we split our original image dataset into $N$ partitions using a Dirichlet sampling procedure with parameter $\alpha$, as is common in FL literature (38; 39), with a lower value of $\alpha$ implying a more heterogeneous split. Further details can be found in Appendix D.

FedFisher outperforms baselines across varying heterogeneity and number of clients. We evaluate FedFisher and other baselines in the range of $\alpha = \{0.05, 0.075, 0.1\}$ which can be considered as moderate to high data heterogeneity. Table 1 summarizes the results obtained the algorithms across the various datasets. We see that FedFisher variants comprehensively outperform baselines in all regimes. In particular for MNIST, FashionMNIST and SVHN, FedFisher(K-FAC) gives almost 10% improvement compared to the nearest baseline. Table 2 summarizes the results obtained when we keep the heterogeneity parameter $\alpha = 0.3$ fixed and vary the number of clients $M$ in the range $\{10, 20, 30\}$. Note that since the total dataset size is fixed, as we increase $M$ each client gets assigned fewer data samples. Thus as $M$ increases, local models have a larger tendency to overfit the data they are trained on, making it harder to aggregate such models. Nonetheless, we see that FedFisher variants, especially FedFisher(K-FAC) continue to outperform other baselines with up to 20% improvement in some cases like the SVHN and relatively lower drop in accuracy with larger $M$. This highlights the effectiveness of FedFisher as a one-shot algorithm which can tackle data heterogeneity in FL settings and scale to relatively large number of clients while being computation and communication efficient.

5. Conclusion

In this work, we propose FedFisher, a novel algorithm for one-shot FL motivated by a Bayesian perspective of FL. We theoretically analyze FedFisher for two-layer over-parameterized neural networks and propose practical versions FedFisher(Diag) and FedFisher(K-FAC) that outperform current state-of-the-art one-shot methods while being computation and communication efficient.
ture work involves extending the analysis of FedFisher for deeper neural networks and investigating the use of differential-privacy to improve privacy guarantees.

References


Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information


A. Additional Related Work

One-Shot FL. We review here some additional work on one-shot FL apart from the knowledge distillation and neuron matching baselines discussed in our work. Initial works such as (35) propose to just use the ensemble of client models at the server. The work (36) discusses how we can improve the prediction of this ensemble when the label distribution across client data is highly skewed. Another line of work proposes that clients send some distilled form of their data to the server (40; 41). However, the privacy guarantees of such methods is unclear. The work (42) proposes techniques to optimize the weights of given to the local models when aggregating them at the server to improve one-shot performance. However, their analysis is limited to simple linear models and does not consider combing neural networks. We also note the existence of works that propose to perform clustering in a one-shot manner in FL setting (43; 44); these approaches our orthogonal to our problem of finding a global minimizer in a one-shot manner.

Convergence of overparameterized NNs in FL. The works (28; 45; 46) study the convergence of FedAvg for overparameterized neural networks. We note that these works are primarily concerned with convergence and do not propose any new algorithms as such compared to our work. We also note the existence of related works (47; 48), that proposes to use NTK style Jacobian features to speed up FL training; however these works usually require multiple training rounds.

B. Proofs


Proposition 2. (Global Posterior Decomposition (19)) Under the flat prior $P(w) \propto 1$, the global posterior decomposes into a product of local posteriors, i.e., $P(w|D) \propto \prod_{i=1}^{M} P(w|D_i)$. Furthermore, modes of the global posterior coincide with the optima of the FL objective in Equation (2), i.e, $\text{arg max}_{w \in \mathbb{R}^d} P(w|D) = \text{arg min}_{w \in \mathbb{R}^d} L(w)$.

Proof. We have,

$$P(w|D) \propto P(D|w) \propto 1$$  \hspace{1cm} (10)

$$= \prod_{i=1}^{M} P(D_i|w) \hspace{1cm} (\mathcal{D}_i \text{ are i.i.d generated})$$ \hspace{1cm} (11)

$$\propto \prod_{i=1}^{M} P(w|D_i).$$  \hspace{1cm} (12)

Also,

$$\text{arg max}_{w \in \mathbb{R}^d} P(w|D) = \text{arg max}_{w \in \mathbb{R}^d} \log P(D|w)$$ \hspace{1cm} (13)

$$= \text{arg max}_{w \in \mathbb{R}^d} \log P(D|w)$$ \hspace{1cm} (14)

$$= \text{arg max}_{w \in \mathbb{R}^d} \sum_{i=1}^{M} \sum_{j=1}^{n} (\log q(x_{ij}) - \ell(f(w, x_{ij}), y_{ij})) \hspace{1cm} (\text{Assumption 1})$$ \hspace{1cm} (15)

$$= \text{arg max}_{w \in \mathbb{R}^d} - \sum_{i=1}^{M} \sum_{j=1}^{n} \ell(f(w, x_{ij}), y_{ij}) \hspace{1cm} (\text{Assumption 1})$$ \hspace{1cm} (16)

$$= \text{arg min}_{w \in \mathbb{R}^d} L(w).$$ \hspace{1cm} (17)

$\square$

B.2. Lemma 1 and Proof.

Lemma 1. Let $w^{(1)}, w^{(2)}, \ldots$ be the iterates generated by running the following gradient descent (GD) procedure: $w^{(t+1)} = w^{(t)} - \eta_S \left( \sum_{i=1}^{M} \tilde{F}_i w^{(t)} - \tilde{F}_i \tilde{w}_i \right)$ with $w^{(0)} = \sum_{i=1}^{M} \tilde{w}_i / M$ and $\eta_S \leq 1/\lambda_{\text{max}}$ where $\lambda_{\text{max}}$ is the maximum eigen value of $\sum_{i=1}^{M} \tilde{F}_i$. Then, $\lim_{T \to \infty} w^{(T)} = w^*$. 
Claim 1. The set of $w$ such that $Fw = b$ is given by,

$$S = \{V_2 V_2^T w + V_1 \Sigma_1^{-1} V_1^T b | w \in \mathbb{R}^d\}. \quad (21)$$

Proof. Let $w = V_2 V_2^T x + V_1 \Sigma_1^{-1} V_1^T b$ for some $x \in \mathbb{R}^d$. Firstly we see that,

$$Fw = V_1 \Sigma_1 V_1^T (V_2 V_2^T x + V_1 \Sigma_1^{-1} V_1^T b) = b \quad (22)$$

Now let $w^o$ be such that $Fw^o = V_1 \Sigma_1 V_1^T w^o = b$. This implies $V_1^T w^o = \Sigma_1^{-1} V_1^T b$. We have,

$$w^o = VV^T w^o \quad (24)$$

$$= V_2 V_2^T w^o + V_1 V_1^T w^o \quad (25)$$

$$= V_2 V_2^T w^o + V_1 \Sigma_1^{-1} V_1^T b \quad (26)$$

Combining Equation (26), Equation (26) we have,

$$w \text{ such that } Fw = b \iff w \in S. \quad (27)$$

Claim 2.

$$w^* = \arg \min_{w \in S} \|w - \bar{w}\|_2^2 = V_2 V_2^T \bar{w} + V_1 \Sigma_1^{-1} V_1^T b \quad (28)$$

Proof.

Case 1: $\bar{w} \in S$. In this case we have, $w^* = V_2 V_2^T \bar{w} + V_1 \Sigma_1^{-1} V_1^T b = \bar{w}$, therefore $\arg \min_{w \in S} \|w - \bar{w}\|_2^2 = 0$.

Case 2: $\bar{w} \notin S$. Suppose $w^* = \arg \min_{w \in S} \|w - \bar{w}\|_2^2 = V_2 V_2^T \bar{w} + V_1 \Sigma_1^{-1} V_1^T b$ for some $\bar{w} \neq \bar{w}$. Let $w^o = V_2 V_2^T \bar{w} + V_1 \Sigma_1^{-1} V_1^T b$. We have,

$$\|\bar{w} - w^*\|_2^2 = \|\bar{w} - V_2 V_2^T \bar{w} - V_1 \Sigma_1^{-1} V_1^T b\|_2^2 \quad (29)$$

$$= \|V_2 V_2^T (\bar{w} - \bar{w}) + V_1 V_1^T \bar{w} - V_1 \Sigma_1^{-1} U^T b\|_2^2 \quad (30)$$

$$= \|V_2 V_2^T (\bar{w} - \bar{w})\|_2^2 + \|V_1 V_1^T \bar{w} - V_1 \Sigma_1^{-1} U^T b\|_2^2 \quad (31)$$

$$= \|V_2 V_2^T (\bar{w} - \bar{w})\|_2^2 + \|V_1 V_1^T \bar{w} + V_2 V_2^T \bar{w} - V_2 V_2^T \bar{w} - V_1 \Sigma_1^{-1} U^T b\|_2^2 \quad (32)$$

$$= \|V_2 V_2^T (\bar{w} - \bar{w})\|_2^2 + \|\bar{w} - w^o\|_2^2 \quad (33)$$

$$> \|\bar{w} - w^o\|_2^2 \quad (34)$$
leading to a contradiction. Thus, \( \mathbf{w}^* = V_2 V_2^T \tilde{\mathbf{w}} + V_1 \Sigma_1^{-1} V_1^T \mathbf{b} \).

Now the GD step in Lemma 1 can be written as,

\[
\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta_S (\mathbf{F} \mathbf{w}^{(t)} - \mathbf{b}) = (I - \eta_S \mathbf{F}) \mathbf{w}^{(t)} + \eta_S \mathbf{b}.
\]

Therefore,

\[
\mathbf{w}^{(T)} = (I - \eta_S \mathbf{F})^T \mathbf{w}^{(0)} + \eta_S \sum_{t=0}^{T-1} (I - \eta_S \mathbf{F})^t \mathbf{b}
\]

\[
= V(I - \eta_S \Sigma)^T V^T \mathbf{w}^{(0)} + \eta_S \sum_{t=0}^{T-1} V(I - \eta_S \Sigma)^t V^T \mathbf{b}
\]

\[
= (V_1(I - \eta_S \Sigma_1)^TV_1 + V_2 V_2^T)\mathbf{w}^{(0)} + \eta_S \left(V_1 \sum_{t=0}^{T-1} (I - \eta_S \Sigma_1)^t V_1 + V_2 V_2^T\right)\mathbf{b}.
\]

\[
= (V_1(I - \eta_S \Sigma_1)^TV_1 + V_2 V_2^T)\mathbf{w}^{(0)} + \eta_S \left(V_1 \sum_{t=0}^{T-1} (I - \eta_S \Sigma_1)^t V_1\right)\mathbf{b}.
\]

where the last line uses the fact \( \mathbf{b} \) lies in the column space of \( \mathbf{F} \).

In the limit \( T \to \infty \) and with \( \eta_S \leq 1/\lambda_{\text{max}}(\Sigma_1) \), we have,

\[
\lim_{T \to \infty} (I - \eta_S \Sigma_1)^T = 0 \quad \text{and} \quad \lim_{T \to \infty} \sum_{t=0}^{T-1} (I - \eta_S \Sigma_1)^t = \frac{1}{\eta_S} \Sigma_1^{-1}.
\]

Thus,

\[
\lim_{T \to \infty} \mathbf{w}^{(T)} = V_2 V_2^T \mathbf{w}^{(0)} + \frac{1}{\eta_S} V_1 \Sigma_1^{-1} V_1^T \mathbf{b}
\]

\[
= V_2 V_2^T \tilde{\mathbf{w}} + \frac{1}{\eta_S} V_1 \Sigma_1^{-1} V_1^T \mathbf{b}
\]

\[
= \mathbf{w}^*
\]

which the last line follows from Equation (21) and Equation (28). This completes our proof.

\[\square\]

\subsection*{B.3. Proofs for 2 layer Overparameterized ReLU NN}

Recall our two-layer ReLU NN is modeled as follows,

\[
f(\mathbf{w}, \mathbf{x}) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r \mathbf{x}^T \mathbf{w}_r \mathbb{1}\{x^T \mathbf{w}_r \geq 0\}
\]

We can write the output of the neural network alternatively as,

\[
f(\mathbf{w}, \mathbf{x}) = \phi(\mathbf{w}, \mathbf{x})^T \mathbf{w}
\]

where \( \phi(\mathbf{w}, \mathbf{x}) = \frac{1}{\sqrt{m}} [a_1 \mathbb{1}\{x^T \mathbf{w}_1 \geq 0\}; a_2 \mathbb{1}\{x^T \mathbf{w}_2 \geq 0\}; \ldots; a_m \mathbb{1}\{x^T \mathbf{w}_m \geq 0\}] \in \mathbb{R}^{mp} \).

We see that,

\[
\nabla_{\mathbf{w}} f(\mathbf{w}, \mathbf{x}) = \phi(\mathbf{w}, \mathbf{x})
\]

Also observe that,

\[
\|\phi(\mathbf{w}, \mathbf{x})\|_2^2 \leq \frac{1}{m} \sum_{i=1}^{m} \|x\|_2^2 = 1
\]

\textbf{Definition of } \mathbf{A}_w, \bar{A}, \mathbf{H}_w, \bar{H}. \quad \text{We define the following matrices } \mathbf{A}_w \text{ and } \bar{A} \text{ that will be used in the proof.}
Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information

\[ A_w = \begin{bmatrix} \phi^T(w, x_{11}) \\ \vdots \\ \phi^T(w, x_{Mn}) \end{bmatrix} \in \mathbb{R}^{(N \times mp)} \]  

(50)

and,

\[ \hat{A} = \begin{bmatrix} \phi^T(\hat{w}_1, x_{11}) \\ \vdots \\ \phi^T(\hat{w}_M, x_{Mn}) \end{bmatrix} \in \mathbb{R}^{(N \times mp)} \]  

(51)

We also define \( H_w = A_w A_w^\top \), \( \hat{H} = \hat{A} \hat{A}^\top \).

**Initialization.** Recall \( w_0 \) is the common initialization point for all the local models before they perform local optimization, i.e., \( w_i^{(0)} = w_0 \). Also recall \( w_i^{(0)} = \bar{w} = \sum_{i=1}^M \bar{w}_i/M \) is the initialization point for the server optimization.

**Simplification of Fisher.** Under the squared loss \( \ell(z, y) = \frac{1}{2} (y - z)^2 \), it can be shown that the Fisher at each client can be simplified as follows (22),

\[
F_i = \frac{1}{n} \sum_{j=1}^n (\nabla_w f(w, x_{ij}) \nabla_w f(w, x_{ij})^\top)_{w=\bar{w}_i} \\
= \frac{1}{n} \sum_{j=1}^n \phi(\bar{w}_i, x_{ij}) \phi(\bar{w}_i, x_{ij})^\top \quad \text{(using Equation (48))}
\]  

(52)

**True Model and Proxy Model Output:** Let \( f(w) = [f(w, x_{11}), f(w, x_{12}), \ldots, f(w, x_{Mn})] \) be the vector of true model responses. Let \( y = [y_{11}, y_{12}, \ldots, y_{Mn}] \) be the vector of true labels and let \( \hat{y} = [f(\bar{w}_1, x_{11}), f(\bar{w}_1, x_{12}), \ldots, f(\bar{w}_M, x_{Mn})] \) be the vector of predicted local model outputs.

For any \( x_{ij} \) and any weight vector \( w \), the proxy model output is defined as \( f^*(w, x_{ij}) = \phi(\bar{w}_i, x_{ij})^\top w \). Let \( \hat{f}(w) = [\hat{f}(w, x_{11}), \hat{f}(w, x_{12}), \ldots, \hat{f}(w, x_{Mn})] \) be the vector of proxy Fisher model responses.

We will now state some lemmas from previous work that we utilize in our proof.

**Lemma 2.** (Theorem 3.1 and Lemma C.1 in (49)) If we set \( m = \Omega\left(\frac{N^6}{\lambda_0^2 \log^5 \frac{N}{\delta}}\right) \) and \( \eta = \mathcal{O}\left(\frac{\lambda_0}{N^2}\right) \), then with probability at least \( 1 - \delta \) over the random initialization of \( w_0 \), we have

\[ \begin{align*}
&\sum_{j=1}^n (y_{ij} - f(w_0, x_{ij}))^2 = \mathcal{O}\left(\frac{n}{\eta}\right) \quad \forall i \in [M] \\
&\sum_{j=1}^n (y_{ij} - f(w, x_{ij}))^2 \leq (1 - \eta \lambda_0/2)^K \mathcal{O}\left(\frac{n}{\eta^2}\right) \quad \forall i \in [M] \\
&\|\bar{w}_{i,r} - w_{0,r}\|_2 \leq 4\sqrt{n(1-\eta \lambda_0/4)^K} \sqrt{\sum_{j=1}^n (y_{ij} - f(w_0, x_{ij}))^2} = R_0 \quad \forall i \in [M], r \in [m]
\end{align*} \]

(53)

Note that (49) provides the above guarantees for a single machine setting. In order for the above result to hold simultaneously for all \( i \in [M] \), we set the failure probability \( \delta' = \delta/M \), leading \( m \) to have a polynomial dependence on \( N \) instead of \( n \).

**Lemma 3.** For a given \( R \), define the following event:

\[ A_{ij,r}(R) = \{ \exists w : \|w - w_{0,r}\|_2 \leq R, \|x_{ij}^\top w_{0,r} \geq 0\} \neq \{ x_{ij}^\top w_0 \geq 0 \} \]

(54)

We have \( \Pr(A_{ij,r}(R)) \leq \frac{2R}{\sqrt{n \eta \lambda_0}} \).

**Lemma 4.** (Lemma 3.1 in (24)). If \( m = \Omega\left(\frac{N^2}{\lambda_0^2 \log \frac{N}{\delta}}\right) \), we have with probability \( 1 - \delta \), \( \|H_{w_0} - H^\infty\|_2 \leq \frac{\lambda_0}{4} \) and \( \lambda_{\min}(H_{w_0}) \geq \frac{3 \lambda_0}{4} \).

The following three lemmas are our contribution and form the basis of our proof.

**Lemma 5.** If \( m = \Omega\left(\frac{N^6}{\lambda_0^2 \log^5 \frac{N}{\delta}}\right) \), with probability \( 1 - \delta \), \( \lambda_{\min}(\hat{H}) \geq \lambda_0/2 \) and \( \lambda_{\max}(\hat{H}) \leq N \).
Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information

Proof.

The \((k,l)\)-th entry of \(\tilde{H}\) is given by,

\[
\tilde{H}_{kl} = \frac{1}{m} x_{k,k'}^T x_{l,l'} \sum_{r=1}^{m} \left( I \left\{ x_{k,k'}^T w_{k',r} \geq 0 \right\} I \left\{ x_{l,l'}^T w_{l',r} \geq 0 \right\} - I \left\{ x_{k,k'}^T w_{0,r} \geq 0 \right\} I \left\{ x_{l,l'}^T w_{0,r} \geq 0 \right\} \right)
\]

where \(k' = \lfloor k/n \rfloor, k' = k - k' + 1, l' = \lceil l/n \rceil, l'' = l - l' + 1\).

We have,

\[
\mathbb{E} \left[ I \left\{ \tilde{H}_{kl} - (H_{\omega^{(0)}})_{kl} \right\} \right]
\]

\[
= \mathbb{E} \left[ \frac{1}{m} x_{k,k'}^T x_{l,l'} \sum_{r=1}^{m} \left( I \left\{ x_{k,k'}^T w_{k',r} \geq 0 \right\} I \left\{ x_{l,l'}^T w_{l',r} \geq 0 \right\} - I \left\{ x_{k,k'}^T w_{0,r} \geq 0 \right\} I \left\{ x_{l,l'}^T w_{0,r} \geq 0 \right\} \right) \right]
\]

\[
\leq \mathbb{E} \left[ \frac{1}{m} \sum_{r=1}^{m} \left( I \left\{ x_{k,k'}^T w_{k',r} \geq 0 \right\} I \left\{ x_{l,l'}^T w_{l',r} \geq 0 \right\} - I \left\{ x_{k,k'}^T w_{0,r} \geq 0 \right\} I \left\{ x_{l,l'}^T w_{0,r} \geq 0 \right\} \right) \right]
\]

\[
\leq \mathbb{E} \left[ \frac{1}{m} \sum_{r=1}^{m} \left( I \left\{ x_{k,k'}^T w_{k',r} \geq 0 \right\} \right) + I \left\{ x_{l,l'}^T w_{l',r} \geq 0 \right\} \right] + I \left\{ x_{l,l'}^T w_{0,r} \geq 0 \right\}
\]

\[
= \frac{1}{m} \sum_{r=1}^{m} \Pr(A_{k,k',r}(R_0) \cup A_{l,l',r}(R_0))
\]

\[
\leq \frac{4R_0}{\sqrt{2\pi \kappa}}
\]

Equation (58) uses Cauchy-Schwartz and Assumption 2, Equation (59) follows from the definition of \(A_{ijr}\) in Lemma 3, Equation (62) uses the result in Lemma 3.

Thus we have,

\[
\mathbb{E} \left[ \| \tilde{H} - H_{\omega^{(0)}} \|_F \right] \leq \mathbb{E} \left[ \sum_{k,l} |\tilde{H}_{kl} - (H_{\omega^{(0)}})_{kl}| \right] \leq \frac{4N^2 R_0}{\sqrt{2\pi}}
\]

By Markov’s inequality, with probability \(1 - \delta\), we have

\[
\| \tilde{H} - H_{\omega^{(0)}} \|_F \leq \frac{4N^2 R_0}{\sqrt{2\pi \delta \kappa}}
\]

Thus,

\[
\| \tilde{H} - H_{\omega^{(0)}} \|_2 \leq \| \tilde{H} - H_{\omega^{(0)}} \|_F \leq \frac{4N^2 R_0}{\sqrt{2\pi \delta \kappa}}
\]

This implies,

\[
\lambda_{\min}(\tilde{H}) \geq \lambda_{\min}(H_{\omega^{(0)})} - \frac{4N^2 R_0}{\sqrt{2\pi \delta \kappa}} \geq \frac{\lambda_0}{2}
\]

where the last line follows from definition of \(R_0\) in Lemma 2 and \(m = \Omega \left( \frac{N^6}{\lambda_0^2 \delta^2 \kappa^2} \right)\).
Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information

We also have,

\[
\| H \|_2 = \left\| \tilde{A} \tilde{A}^\top \right\|_2 = \left\| \tilde{A}^\top \tilde{A} \right\|_2 \tag{67}
\]

\[
= \left\| \sum_{i=1}^{M} \sum_{j=1}^{n} \phi(\tilde{w}_i, x_{ij}) \phi(\tilde{w}_i, x_{ij})^\top \right\|_2 \tag{68}
\]

\[
\leq \sum_{i=1}^{M} \sum_{j=1}^{n} \| \phi(\tilde{w}_i, x_{ij}) \phi(\tilde{w}_i, x_{ij})^\top \|_2 \tag{69}
\]

\[
\leq N \max_{i,j} \| \phi(\tilde{w}_i, x_{ij}) \|_2 \tag{70}
\]

\[
\leq N \tag{71}
\]

Equation (70) follows from the triangle inequality, for Equation (72) we use Equation (49) and Assumption 2.

Therefore \( \lambda_{\text{max}}(H) = \| H \|_2 \leq N \).

\[\square\]

**Lemma 6.** (Theorem 3.1 and Lemma C.1 in (49)) If we set \( m = \Omega \left( \frac{N^6}{\lambda_0^2 \delta^2} \right) \) and \( \eta = O \left( \frac{\lambda_0}{N^2} \right) \), then with probability at least \( 1 - \delta \) over the random initialization of \( w_0 \), we have

- \( \| f(w^{(0)}) - \tilde{y} \|_2^2 = O \left( \frac{N^2}{\lambda_0^2} \right) \)
- \( \| f(w^{(t)}) - \tilde{y} \|_2^2 \leq (1 - \eta S \lambda_0 / 2n)^t \| f(w^{(0)}) - \tilde{y} \|_2^2 \)
- \( \| w_r^* - w_r^{(0)} \|_2 \leq \frac{4\sqrt{N}}{m \lambda_0} \| f(w^{(0)}) - \tilde{y} \|_2 = R_1 \)

**Proof.**

**First part.**

We have,

\[
(\bar{f}(w^{(0)}, x_{ij}) - \tilde{y}_{ij})^2 = (\phi(\tilde{w}_1, x_{ij})^\top w^{(0)} - \phi(\tilde{w}_i, x_{ij})^\top \tilde{w}_i)^2 \tag{73}
\]

\[
= \left( \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r I \{ x_{ij}^\top \tilde{w}_{i,r} \} x_{ij}^\top \left( \sum_{l=1}^{M} \tilde{w}_{l,r}/M - \tilde{w}_{i,r} \right) \right)^2 \tag{74}
\]

\[
\leq \sum_{r=1}^{m} \left( a_r I \{ x_{ij}^\top \tilde{w}_{i,r} \} x_{ij}^\top \left( \sum_{l=1}^{M} \tilde{w}_{l,r}/M - \tilde{w}_{i,r} \right) \right)^2 \tag{75}
\]

\[
\leq \sum_{r=1}^{m} \left( \sum_{l=1}^{M} \tilde{w}_{l,r}/M - \tilde{w}_{i,r} \right)^2 \tag{76}
\]

\[
\leq 4m R_0^2 \tag{77}
\]

\[
= O \left( \frac{n^2}{\lambda_0^2 \delta} \right) \tag{78}
\]

where Equation (75) follows from Jensen’s inequality, Equation (76) uses Cauchy-Schwartz and Assumption 2, Equation (78) follows from Lemma 2.

Thus, \( \| \bar{f}(w^{(0)}) - \tilde{y} \|_2^2 = O \left( \frac{N^2}{\lambda_0^2} \right) \leq O \left( \frac{N n^2}{\lambda_0^2} \right) \)
Second part.

The GD step that the central server performs for FedFisher can be written as,

$$w^{(t+1)} = w^{(t)} - \eta_S \sum_{i=1}^{M} F_i \left( w^{(t)} - \bar{w}_i \right)$$  \hspace{1cm} (79)

$$= w^{(t)} - \eta_S \sum_{i=1}^{M} \sum_{j=1}^{n} \phi(\bar{w}_i, x_{ij}) \phi(\bar{w}_i, x_{ij})^T \left( w^{(t)} - \bar{w}_i \right) \quad (\eta_S = \eta_S / n)$$  \hspace{1cm} (80)

$$= w^{(t)} - \eta_S \sum_{i=1}^{M} \sum_{j=1}^{n} \phi(\bar{w}_i, x_{ij}) \left( \hat{f}(w^{(t)}, x_{ij}) - \tilde{y}_{ij} \right)$$  \hspace{1cm} (81)

$$= w^{(t)} - \eta_S \tilde{A}^T (\hat{f}(w^{(t)}) - \tilde{y})$$  \hspace{1cm} (82)

Equation (81), Equation (82) follow from the definition of $\hat{f}(w, x_{ij}), \tilde{y}_{ij}, \tilde{A}$ in Appendix B.3.

We have,

$$\hat{f}(w^{(t+1)}) - \hat{f}(w^{(t)}) = \tilde{A}(w^{(t+1)} - w^{(t)})$$  \hspace{1cm} (83)

$$= -\eta_S \tilde{H}(\hat{f}(w^{(t)}) - \tilde{y})$$  \hspace{1cm} (84)

Therefore,

$$\left\| \hat{f}(w^{(t+1)}) - \tilde{y} \right\|_2^2 = \left\| \hat{f}(w^{(t+1)}) - \hat{f}(w^{(t)}) + \hat{f}(w^{(t)}) - \tilde{y} \right\|_2^2$$  \hspace{1cm} (85)

$$= \left\| \hat{f}(w^{(t+1)}) - \tilde{y} \right\|_2^2 - 2\eta_S (\hat{f}(w^{(t)}) - \tilde{y})\tilde{H}(\hat{f}(w^{(t)}) - \tilde{y})$$

$$+ \eta_S^2 \left\| \tilde{H}(\hat{f}(w^{(t)}) - \tilde{y}) \right\|_2^2$$  \hspace{1cm} (86)

$$\leq (1 - \eta_S \lambda_0 + \eta_S^2 N^2) \left\| \hat{f}(w^{(t)}) - \tilde{y} \right\|_2^2$$  \hspace{1cm} (87)

$$\leq \left( 1 - \frac{\eta_S \lambda_0}{2n} \right) \left\| \hat{f}(w^{(t)}) - \tilde{y} \right\|_2^2$$  \hspace{1cm} (88)

where Equation (87) follows from Lemma 5 and Equation (88) follows from using $\tilde{\eta}_S \leq \frac{\lambda_0}{2N^2}$

Third part.

We have Equation (81),

$$\left\| w_r^{(t+1)} - w_r^{(t)} \right\|_2 = \left\| \frac{1}{\sqrt{m}} a_r \eta_S \sum_{i=1}^{M} \sum_{j=1}^{n} x_{ij} \mathbb{I} \{ x_{ij}^T w_r^{(t)} \geq 0 \} (\tilde{y}_{ij} - \hat{f}(w^{(t)}, x_{ij})) \right\|_2$$

$$\leq \eta_S \sqrt{\frac{N}{m}} \left\| \tilde{y} - \hat{f}(w^{(t)}) \right\|_2$$  \hspace{1cm} (89)

where the last inequality follows from Cauchy-Schwartz and Assumption 2.

Therefore,
Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information

\[
\| \mathbf{w}_r^* - \mathbf{w}_r^{(0)} \|_2 \leq \sum_{t=0}^{\infty} \| \mathbf{w}_r^{(t+1)} - \mathbf{w}_r^{(t)} \|_2 \leq \frac{\hat{\eta}_S \sqrt{N}}{\sqrt{m}} \sum_{t=0}^{\infty} \| \mathbf{y} - \hat{f}(\mathbf{w}^{(t)}) \|_2 \leq \frac{\hat{\eta}_S \sqrt{N}}{\sqrt{m}} \sum_{t=0}^{\infty} \left(1 - \frac{\hat{\eta}_S \lambda_0}{4}\right)^t \| \mathbf{y} - \hat{f}(\mathbf{w}^{(0)}) \|_2 \leq 4\sqrt{N} \frac{\| \mathbf{y} - \hat{f}(\mathbf{w}^{(0)}) \|_2}{m \lambda_0}
\]

(91) (92) (93) (94)

where Equation (93) follows from Equation (88).

**Lemma 7.** Let \( S_{ij} = \{ r \in [m] : I \{ \mathbf{x}_{ij}^\top \mathbf{w}_r^* \geq 0 \} = I \{ \mathbf{x}_{ij}^\top \mathbf{w}_0 \geq 0 \} \} \) and \( S_{ij}^\perp = [m] - S_{ij} \). With probability \( 1 - \delta \) over the initialization, we have \( \sum_{i=1}^{M} \sum_{j=1}^{n} |S_{ij}^\perp|^2 = \mathcal{O} \left( \frac{m^2 N^2 (R_0^2 + R_1^2)}{\delta^2} \right) \)

**Proof.**

We have,

\[
\mathbb{E} [ |S_{ij}^\perp| ] = \sum_{i=1}^{m} \Pr( I \{ \mathbf{x}_{ij}^\top \mathbf{w}_r^* \geq 0 \} \neq I \{ \mathbf{x}_{ij}^\top \mathbf{w}_0 \geq 0 \}) \leq \Pr ( I \{ \mathbf{x}_{ij}^\top \mathbf{w}_r^* \geq 0 \} \neq I \{ \mathbf{x}_{ij}^\top \mathbf{w}_0 \geq 0 \} ) \cup \{ I \{ \mathbf{x}_{ij}^\top \mathbf{w}_0 \geq 0 \} \} \leq \Pr(A_{ijr}(R_0 + R_1)) + \Pr(A_{ijr}(R_0)) \leq \frac{4m(R_0 + R_1)}{\sqrt{2\pi}}
\]

(95) (96) (97) (98)

Equation (97) follows from union-bound and Lemma 3, Lemma 2, Lemma 6.

Thus, using Markov’s inequality, with probability at least \( 1 - \delta \) we have,

\[
|S_{ij}^\perp| = \mathcal{O} \left( \frac{m(R_0 + R_1)}{\delta} \right)
\]

(99)

Setting the failure probability of each \( |S_{ij}| \) being large as \( \delta/N \), we have with probability \( 1 - \delta \), for all \( i \in [M] \) and \( j \in [n] \) simultaneously,

\[
|S_{ij}^\perp| = \mathcal{O} \left( \frac{mN(R_0 + R_1)}{\delta} \right)
\]

(100)

Thus with probability \( 1 - \delta \) we have,

\[
\sum_{i=1}^{M} \sum_{j=1}^{n} |S_{ij}|^2 \leq \mathcal{O} \left( \frac{m^2 N^2 (R_0^2 + R_1^2)}{\delta^2} \right)
\]

(101)

**Proof of Theorem 1.**

We first state the full theorem statement with the exact dependence of \( M \) on \( (N, \lambda_0^{-1}, \delta^{-1}, \kappa^{-1}) \).

**Theorem 2.** Under Assumptions 2, 3, for \( m = \Omega \left( \frac{N^9}{\lambda_0^{5} \kappa \delta^2} \right) \), and i.i.d Gaussian initialization weights of \( \mathbf{w}_0 \) as \( \mathbf{w}_{0,r} \sim \mathcal{N}(0, \kappa) \), and initializing \( a_r = \{-1, 1\} \) with probability \( 1/2 \) for all \( r \in [m] \), for step sizes \( \eta = \mathcal{O}(\lambda_0/N^2) \), \( \eta_S = \mathcal{O}(\lambda_0/N^2) \) and for a given failure probability \( \delta \in (0, 1) \), the following is true with probability \( 1 - \delta \) over the random initialization:

\[
L(\mathbf{w}^*) \leq \mathcal{O} \left( \frac{(1 - \eta \lambda_0/2) K N}{\delta} \right) + \mathcal{O} \left( \frac{(2 - (1 - \eta \lambda_0/2) K) N^9}{\lambda_0^8 \delta^4 m} \right)
\]

(102)

**Proof.**

15
Note that the conditions in all the preceding lemmas are satisfied by setting \( m = \Omega \left( \frac{N^9}{\lambda_0 \delta^2 \lambda_0^2} \right) \), \( \eta = \mathcal{O}(\lambda_0/N^2) \), \( \eta_S = \mathcal{O}(\lambda_0/N^2) \) and hence we can now apply these lemma results for our proof.

From Lemma 6, we observe that

\[
\hat{y}_{ij} = \hat{f}(w^*, x_{ij}) = \phi(\hat{w}_i, x_{ij})^\top w^*
\]

We have,

\[
L(w^*) = \frac{1}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} (f(w^*, x_{ij}) - y_{ij})^2
\]

\[
= \frac{1}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( (\phi(w^*, x_{ij})^\top w^* - \hat{y}_{ij} + \hat{y}_{ij} - y_{ij})^2 \right)
\]

\[
\leq \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \phi(w^*, x_{ij})^\top w^* - \hat{y}_{ij} \right)^2 + \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} (\hat{y}_{ij} - y_{ij})^2
\]

\( T_2 \) can be bounded as \( \mathcal{O} \left( (1 - \eta \lambda_0/2)^K \frac{N}{\delta^2} \right) \) using the result from Lemma 2.

Our goal is to now bound \( T_1 \) as follows.

\[
T_1 = \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \phi(w^*, x_{ij})^\top w^* - \hat{y}_{ij} \right)^2
\]

\[
= \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \phi(w^*, x_{ij})^\top w^* - \phi(\hat{w}_i, x_{ij})^\top w^* \right)^2
\]

\[
= \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r x_{ij}^\top w_r^* (1 \{ x_{ij}^\top w_r^* \geq 0 \} - 1 \{ x_{ij}^\top \hat{w}_{i,r} \geq 0 \} ) \right)^2
\]

\[
= \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \frac{1}{\sqrt{m}} \sum_{r \in S^+_{ij}} a_r x_{ij}^\top w_r^* (1 \{ x_{ij}^\top w_r^* \geq 0 \} - 1 \{ x_{ij}^\top \hat{w}_{i,r} \geq 0 \} ) \right)^2
\]

\[
\leq \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \frac{|S^+_{ij}|}{m} \sum_{r \in S^+_{ij}} (a_r x_{ij}^\top w_r^*)^2 \right)
\]

\[
\leq \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \frac{|S^+_{ij}|}{m} \sum_{r \in S^+_{ij}} (x_{ij}^\top w_r^* - x_{ij}^\top \hat{w}_{i,r})^2 \right)
\]

\[
\leq \frac{2}{N} \sum_{i=1}^{M} \sum_{j=1}^{n} \left( \frac{|S^+_{ij}|^2}{m} \max_{r \in [m]} \| w_r^* - \hat{w}_{i,r} \|^2 \right)
\]

\[
\leq \mathcal{O} \left( \frac{m^2 N (R^1_0 + R^2_1)}{\delta^2 m} \right)
\]

\[
= \mathcal{O} \left( \frac{N^9}{\lambda_0 \delta^2 \lambda_0} (2 - (1 - \eta \lambda_0/2)^K) \right)
\]

Equation (110) uses Jensen’s inequality and definition of \( S_{ij} \) in Lemma 7, Equation (111) uses the observation that since \( r \in S^+_{ij} \) we have sign\( x_{ij}^\top w_r^* \) \( \neq \) sign\( x_{ij}^\top \hat{w}_{i,r} \) which implies \( |x_{ij}^\top w_r^* - x_{ij}^\top \hat{w}_{i,r}| \). Equation (112) uses Cauchy Schwartz. Equation (113) uses Lemma 7, Lemma 6, Lemma 2, Equation (114) substitutes \( R_0 = \mathcal{O} \left( \frac{n}{\sqrt{m^2 \lambda_0} (1 - (1 - \eta \lambda_0/4)^K) } \right) \) and \( R_1 = \mathcal{O} \left( \frac{N^2}{\sqrt{m^2 \lambda_0} } \right) \).
Thus we have,
\[
L(w^*) \leq O \left( \left(1 - \eta \lambda_0/2 \right)^K \frac{N}{\delta} \right) + O \left( \left(2 - (1 - \eta \lambda_0/2)^K \right) \frac{N^9}{\lambda_0^8 \delta^4 m} \right). 
\]
(115)
This completes the proof.

C. Computation and Communication Efficiency of FedFisher (Diag) and FedFisher (K–FAC)

C.1. Computation Efficiency

To compute their diagonal Fisher or K–FAC, clients need to perform an additional forward-backward pass over the data, i.e., an additional epoch of training, plus some small overhead cost. This is a reasonable cost for FL setups since the bulk of the computation cost goes into computing \(\tilde{w}_i\), which needs multiple local epochs. We also experimentally verify this in Table 3, where we show that FedFisher (Diag) and FedFisher (K–FAC) add less than 12.5% of the total computational cost of FedAvg at clients for our FL setups.

<table>
<thead>
<tr>
<th>Dataset/Model</th>
<th>FedAvg</th>
<th>FedFisher (Diag)</th>
<th>FedFisher (K–FAC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST/MLP</td>
<td>18.7</td>
<td>19.7 (+5.3%)</td>
<td>19.9 (+6.4%)</td>
</tr>
<tr>
<td>CIFAR10/CNN</td>
<td>36.1</td>
<td>40.5 (+12.1%)</td>
<td>38.5 (+6.6%)</td>
</tr>
<tr>
<td>CINIC10/ResNet18</td>
<td>314.5</td>
<td>353.8 (+12.4%)</td>
<td>344.3 (+9.4%)</td>
</tr>
</tbody>
</table>

C.2. Communication Efficiency

We assume that the number of bits used to represent a scalar is 32 by default. For FedAvg, clients just need to transfer \(\tilde{w}_i\), making the total communication cost 32\(d\) bits. Our goal in this section is to show that we can introduce compression techniques in FedFisher (Diag) and FedFisher (K–FAC) to match the communication cost of FedAvg while having similar accuracy as the uncompressed version of these algorithms. To do so, we use standard uniform quantization and SVD compression as described below.

Uniform Quantization. Let \(s_q \in \{1, 2, \ldots, 16\}\) be the factor by which we want to compress our information. We define number of quantization levels as \(l_q = 2^{[32/s_q]} - 1\). Now given a vector \(x \in \mathbb{R}^d\), we define each element of the quantized \(x\) as follows,
\[
Q(x, s_q)_i = \|x\|_\infty \text{sign}(x_i) \zeta_i(x, s_q) 
\]
(116)
where \(\|x\|_\infty = \max_{i \in [d]} |x_i|\) and \(\zeta_i(x, s_q) = \left[l_q \frac{|x_i|}{\|x\|_\infty}\right]/l_q\). Now to communicate \(\zeta_i(x, s_q)\) we only need \([32/s_q] - 1\) bits, to communicate \(\text{sign}(x_i)\) we need 1 bit and to communicate \(\|x\|_\infty\) we need 32 bits. Thus the communication cost of \(Q(x, s_q)\) becomes \(d([32/s_q] - 1) + d + 32 = d([32/s_q]) + 32\) bits.

Singular Value Decomposition Compression. Let \(A \in \mathbb{R}^{(m \times m)}\) matrix. The SVD decomposition of \(A\) can be written as,
\[
A = U \Sigma V^T 
\]
(117)
where \(U \in \mathbb{R}^{(m \times m)}\) is the matrix of left singular vectors, \(\Sigma \in \mathbb{R}^{(m \times m)}\) is a diagonal matrix with each element corresponding to a singular value and \(V \in \mathbb{R}^{(m \times m)}\) is the matrix of right of singular vectors. The singular values are assumed to be sorted by magnitude, i.e., \(|\Sigma_{1,1}| \geq |\Sigma_{2,2}| \ldots |\Sigma_{m,m}|\). We see that the total cost for communicating \(A\) will 32\(m^2\) bits. To reduce this cost, a natural idea is to send only a limited number of singular values and singular vectors obtained by the SVD decomposition of \(A\). Specifically let \(s_v\) be the factor by which we want to compress the information in \(A\). We define \(l_v = [m/2s_v]\). Then the SVD decomposition of \(A\) can be defined as,
\[
V(A, s_v) = U_{l_v} \Sigma_{l_v} V_{l_v}^T 
\]
(118)
where \(U_{l_v} \in \mathbb{R}^{(m \times l_v)}\) corresponds to first \(l_v\) columns of \(U\), \(\Sigma_{l_v} \in \mathbb{R}^{(l_v \times l_v)}\) is a diagonal matrix corresponding to the first \(l_v\) elements of \(\Sigma\) and \(V \in \mathbb{R}^{(m \times l_v)}\) corresponds to the first \(l_v\) columns of \(V\). The communication cost of \(V(A, s_v)\)
Towards a Theoretical and Practical Understanding of One-Shot Federated Learning with Fisher Information

becomes $32(ml_v + l_v + ml_v) \approx 64ml_v \leq 32n^2/s_v$ bits, thereby achieving close to $s_v$ compression.

Compression in FedFisher (Diag). For FedFisher (Diag), clients need to communicate $\hat{w}_i$ and $\hat{F}_i$ where the number of parameters in $\hat{F}_i$ is exactly $d$. To ensure comparable communication to FedAvg, we quantize the weights corresponding to each layer of a neural network in $\hat{w}_i$ and $\hat{F}_i$ by a factor of 2, i.e., $s_q = 2$. This ensures that the communication of FedFisher (Diag) is $32(d + 2L)$ bits where $L$ is the number of layers in the neural network. We note that there is a small overhead of $64L$ bits; however, is negligible since $d \gg 2L$ for our neural networks. Table 4 shows that the compression affects the accuracy of FedFisher (Diag) by less than $1\%$ for the CIFAR-10 and MLP datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>FedAvg</th>
<th>FedFisher (Diag)</th>
<th>FedFisher (Diag) (Compressed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FashionMNIST</td>
<td>74.30</td>
<td>78.53</td>
<td>77.59</td>
</tr>
<tr>
<td>CIFAR10</td>
<td>44.12</td>
<td>45.32</td>
<td>44.67</td>
</tr>
</tbody>
</table>

Compression in FedFisher (K-FAC). Let $\{m_0, m_1, m_2, \ldots, m_L\}$ be the dimensions of each layer of a $L$ layer neural network with $m_0$ corresponding to the dimension of the input. For FedFisher (K-FAC), $\hat{F}_i$ can be represented as $\{(A_1 \otimes B_1), (A_2 \otimes B_2), \ldots, (A_L \otimes B_L)\}$ where $A_i \in \mathbb{R}^{(m_{i-1} \times m_i)}$ and $B_i \in \mathbb{R}^{(m_i \times m_i)}$ represents the Kronecker factors of the $l$-th layer. Thus, the communication cost of $\hat{F}_i$ in this case is $\sum_{l=1}^{L} 32(m_{l-1}^2 + m_l^2)$ bits.

Our goal is to ensure that the communication cost of compressed $\hat{F}_i$ is less than $16d$ bits (we compress $\hat{w}_i$ to $16d$ bits using quantization, similar to FedFisher (Diag)). To do so, we use a mix of quantization and SVD compression. Specifically each $A_i$ and $B_i$ is first compressed to the SVD decomposition corresponding to maintaining the top $l_v$ vectors. This ensures that the communication cost of $(A_i, B_i)$ is $32(2m_{l-1}l_v + 2ml_v + 2l_v)$. This SVD decomposition is then further compressed using $s_q$ quantization compression to ensure that the communication cost is $(32/s_q)(2m_{l-1}l_v + 2ml_v + 2l_v)$. We set $s_q$ and $l_v$ such that $\sum_{l=1}^{L} (32/s_q)(2m_{l-1}l_v + 2ml_v + 2l_v) \leq 16d$. The corresponding $s_v$ is then defined as $\lceil m/2l_v \rceil$.

Table 5 and Table 6 summarizes the results obtained by different levels of $s_q$ and $s_v$ for the FashionMNIST and CIFAR datasets respectively. We see that keeping $s_q = 4$ and setting $s_v$, accordingly usually gives the best performance and hence we use this setting for all our experiments.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$s_q$</th>
<th>$s_v$</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>FedAvg</td>
<td>$\times$</td>
<td>$\times$</td>
<td>77.81</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>$\times$</td>
<td>$\times$</td>
<td>84.93</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>2</td>
<td>2.5</td>
<td>82.70</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>4</td>
<td>1.25</td>
<td>83.30</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>8</td>
<td>1</td>
<td>79.71</td>
</tr>
</tbody>
</table>

Table 6: Test accuracy performance of FedFisher (K-FAC) with different levels of compression on CIFAR10 with 5 clients and $\alpha = 0.5$

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$s_q$</th>
<th>$s_v$</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>FedAvg</td>
<td>$\times$</td>
<td>$\times$</td>
<td>48.93</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>$\times$</td>
<td>$\times$</td>
<td>63.97</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>2</td>
<td>8.1</td>
<td>60.51</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>4</td>
<td>4.1</td>
<td>62.67</td>
</tr>
<tr>
<td>FedFisher (K-FAC)</td>
<td>8</td>
<td>2.1</td>
<td>48.57</td>
</tr>
</tbody>
</table>
D. Additional Experimental Details

The local optimization procedure is the same across all algorithms. In particular, clients perform $T$ epochs of local training using the SGD optimizer with local learning rate $\eta = 0.01$, batch size 64 and momentum factor 0.9. We set $T = 100$ for the experiments on CINIC-10 and $T = 30$ for the others. To compute the Fisher diagonal and Fisher K-FAC we use the nngeometry package (50). For hyperparameter tuning we assume that the server has access to a dataset of 500 samples, sampled uniformly from the original training set. We describe the hyperparameters tuned for each of the algorithms below.

**FedFisher(Diag) and FedFisher(K-FAC).** While Algorithm 1 performs the server optimization with GD, this can be replaced with any other suitable GD optimizer. We choose to use the Adam optimizer here. We set $\eta_S = 0.01, \beta_1 = 0.9, \beta_2 = 0.99$ and $\epsilon = 0.01$ for the Adam optimizer and number of steps $T = 2000$ for all our experiments. We measure the validation performance after every 100 steps and use the model which achieves the best validation performance as the final FedFisher model.

**PFNM.** For PFMN we tune the $\sigma, \sigma_0$ and $\gamma$ parameters as done in the official implementation available at https://github.com/IBM/probabilistic-federated-neural-matching. The grid for $\sigma$ is $\{0.1, 0.5, 1.0\}$, grid for $\sigma_0$ is $\{1.0, 10.0\}$ and grid for $\gamma$ is $\{0.001, 1, 50\}$.

**DENSE.** For DENSE we use the default settings as described in Section 3.1.4 of (9). In particular we set $T_G = 30, \lambda_1 = 1, \lambda_2 = 0.5$ and train the server model with SGD optimizer with learning rate $\eta_S = 0.01$ and momentum factor 0.9. To ensure computational fairness among baselines we limit the number of distillation epochs $T = 20$. We use the validation data to determine the model which achieves the best validation performance during the server training and use this as the final model.

**OTFusion.** The official code for OTFusion available at https://github.com/sidak/otfusion contains more than 15 hyperparameters, making it hard to tune each of these parameters. Among these we found that the correction hyperparameter and type of ground metric normalization affect performance most and hence we focus on tuning these hyperparameters for our experiments, while using the default settings for the other hyperparameters.