Supplementary material for "Minimizing Layerwise Activation Norm Improves Generalization in Federated Learning"

M. Yashwanth[1], Gaurav Kumar Nayak[2], Harsh Rangwani[1], Arya Singh[3],
R. Venkatesh Babu[1], and Anirban Chakraborty[1]

[1]Indian Institute of Science, Bangalore, [2]University of Central Florida, [3]BITS Pilani

[1]{yashwanthm,harshr,venky,anirban}@iisc.ac.in, [2]gauravkumar.nayak@ucf.edu,
[3]f20180762g@alumni.bits-pilani.ac.in

Contents

1. Notations and Preliminaries

We describe the necessary preliminaries and notation below.

1.1. Notations

By default, we assume the notation $\|.\|$ for 2-norm. $\|.\|_F$ is the Frobenious norm. $\|.\|_2$ is the usual euclidean norm. $\lambda_{max}(\mathbf{A})$ is maximum eigenvalue of matrix $\mathbf{A}$. $\odot$ denotes the element-wise product of two matrices or vectors. $\otimes$ denotes the Kronecker product of two matrices or vectors. $\nabla_{\mathbf{b}}\mathbf{a}$ denotes the Jacobian of $\mathbf{a}$ w.r.t $\mathbf{b}$. eig($\mathbf{A}$) denotes the eigenvalues of matrix $\mathbf{A}$. $vec(\mathbf{A})$ denotes the vectorization operation. If $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$, then $vec(\mathbf{A}) \in \mathbb{R}^{n_1 n_2 \times 1}$($n_2$ columns of $\mathbf{A}$ are stacked one after other). $\mathbf{H}_{\mathbf{W}}(\mathcal{L})$ is Hessian of Loss $\mathcal{L}$ with respect to parameters $\mathbf{W}$. $\lambda(\mathbf{H})$ denotes eigenvalue of $\mathbf{H}$. $\mathcal{O}$ denotes the usual Big-**O** notation.

1.2. Preliminaries

If $\mathbf{x} \in \mathbb{R}^{d \times 1}$

$$eig(\mathbf{x}\mathbf{x}^\top) = \|\mathbf{x}\|_2^2 \tag{1}$$

$$(\mathbf{A} \otimes \mathbf{B})^\top = \mathbf{A}^\top \otimes \mathbf{B}^\top \tag{2}$$

If **A**,**B**,**C**,**D** matrices of compatible dimensions then the following holds

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD} \tag{3}$$

$$(\mathbf{A} \otimes \mathbf{B})\mathbf{C} = \mathbf{AC} \otimes \mathbf{B} \tag{4}$$

$$eig(\mathbf{A} \otimes \mathbf{B}) = eig(\mathbf{A}) \otimes eig(\mathbf{B}). \tag{5}$$

For any symmetric matrices $\mathbf{S}_1$ and $\mathbf{S}_2$, the following holds.

$$\lambda_{max}(\mathbf{S}_1 + \mathbf{S}_2) \leq \lambda_{max}(\mathbf{S}_1) + \lambda_{max}(\mathbf{S}_2) \tag{6}$$

We now provide proof of the theorems in the next section.

2. Proof for the Theorems in the main paper

Theorem 1. *If $\mathbf{H}_{ll} \in R^{d_l}$ denotes the layer l Hessian and $\mathbf{H} \in R^d$ denotes the over all Hessian and $\sum_{l=1}^{L} d_l = d$, where L is the total number of layers. If the Hessian entries are bounded above we then have the following result.* $\lambda(\mathbf{H}) \in \cup_{l=1}^{L}[\lambda_{min}(H_{ll}) - \mathcal{O}(max(d_l, d-d_l)), \lambda_{max}(H_{ll}) + \mathcal{O}(max(d_l, d-d_l))]$

Proof. Let $\mathbf{x} \in R^{d\times 1}$ be the eigen vector of $\mathbf{H}$ be the Hessian matrix and it is symmetric partitioned with $\mathbf{H}_{ij} \in R^{d_i \times d_j}$. The block diagonal matrices are the ones with $i = j$ and there are L such matrices along the diagonal. We also assume that $\mathbf{x}$ is partitioned into L vectors as $\mathbf{x} = [\mathbf{x}_1{}^\intercal, \mathbf{x}_2{}^\intercal, ..., \mathbf{x}_L{}^\intercal]^\intercal$ where $\mathbf{x}_j \in R^{d_j \times 1}$.

Since $\mathbf{x}$ is assumed to be the eigenvector and λ be associated eigenvalue we have the following

$$\lambda \mathbf{x}_l = \mathbf{H}_{ll}\mathbf{x}_l + \sum_{j=1,j\neq l}^{L} \mathbf{H}_{lj}\mathbf{x}_j \tag{7}$$

Here $\mathbf{x}_l$ is chosen such that $\|\mathbf{x}_l\| \geq \|\mathbf{x}_i\|$ for all i. Throughout the proof we mean $\|.\|$ as 2-norm.

Taking the norm on Eq. 7 we get the following

$$\begin{aligned} \|(\lambda \mathbf{I} - \mathbf{H}_{ll})\mathbf{x}_l\| &= \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\mathbf{x}_j\| \\ &\leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\|\|\mathbf{x}_j\| \end{aligned} \tag{8}$$

The first inequality is by triangle inequality and the second is by the definition of norm $\|\mathbf{A}\mathbf{x}\| \leq \|\mathbf{A}\|\|\mathbf{x}\|$.

Dividing the equation 8 by $\|\mathbf{x}_l\|$ we get the following

$$\begin{aligned} \frac{\|(\lambda \mathbf{I} - \mathbf{H}_{ll})\mathbf{x}_l\|}{\|\mathbf{x}_l\|} &\leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| \frac{\|\mathbf{x}_j\|}{\|\mathbf{x}_l\|} \\ &\leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| \end{aligned} \tag{9}$$

The second inequality follows as $\|\mathbf{x}_j\| \leq \|\mathbf{x}_l\|$.

It is easy to see that

$$\frac{\|(\lambda \mathbf{I} - \mathbf{H}_{ll})\mathbf{x}_l\|}{\|\mathbf{x}_l\|} \geq \min_i |\lambda - \lambda_i(\mathbf{H}_{ll})| \tag{10}$$

as $(\lambda \mathbf{I} - \mathbf{H}_{ll})$ is a symmetric matrix. Here $\lambda_1(\mathbf{H}_{ll}) \geq \lambda_2(\mathbf{H}_{ll}) ... \geq \lambda_{d_l}(\mathbf{H}_{ll})$ using Eq 9 and Eq 10 we get the following.

$$\min_i |\lambda - \lambda_i(\mathbf{H}_{ll})| \leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| \tag{11}$$

This implies that any of the following is true

$|\lambda - \lambda_1(\mathbf{H}_{ll})| \leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\|$ or $|\lambda - \lambda_2(\mathbf{H}_{ll})| \leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\|$ or the $|\lambda - \lambda_{d_l}(\mathbf{H}_{ll})| \leq \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\|$.

Hence, we take the worst-case possibility that contains all the regions i.e.,

$$\lambda \leq \lambda_1(\mathbf{H}_{ll}) + \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| \tag{12}$$

$$\lambda \geq \lambda_{d_l}(\mathbf{H}_{ll}) - \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| \tag{13}$$

Note that $\lambda_1(\mathbf{H}_{ll}) \triangleq \lambda_{max}(\mathbf{H}_{ll})$ and $\lambda_{d_l}(\mathbf{H}_{ll}) \triangleq \lambda_{min}(\mathbf{H}_{ll})$.

It remains to show that $\sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| \leq \mathcal{O}(max(d_l, d - d_l))$.

$$\|\mathbf{H}_{lj}\|^2 \leq \|\mathbf{H}_{lj}\|_F^2 \leq B d_l d_j \tag{14}$$

where $\|\mathbf{H}_{lj}\|_F$ is the Frobenious norm of the matrix. The first inequality in Eq. 14 is because the 2-norm is, at most, the Frobenious norm, and the second inequality is from the assumption of the theorem that each entry is bounded above by B.

$$\begin{aligned} \sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\| &\leq \sqrt{(L-1)\sum_{j=1,j\neq l}^{L} \|\mathbf{H}_{lj}\|^2} \\ &\leq \sqrt{B(L-1)d_l(d-d_l)} \\ &\leq \mathcal{O}(max(d_l, d-d_l)). \end{aligned} \tag{15}$$

The first inequality uses Cauchy Schwartz inequality, and the second inequality uses Eq. 14 and also the fact that $\sum_{i=1}^{i=1} d_i = d$ and the third inequality follows by definition of $\mathcal{O}$. From Eq. 12, Eq. 13 and Eq. 15 we have shown that $\lambda(\mathbf{H}) \in [\lambda_{min}(H_{ll}) - \mathcal{O}(max(d_l, d-d_l)), \lambda_{max}(H_{ll}) + \mathcal{O}(max(d_l, d-d_l))]$. But the l can be anywhere from $l = 1$ to $l = L$ hence we take the union of all the possible regions and hence we get the desired result $\lambda(\mathbf{H}) \in \cup_{l=1}^{L}[\lambda_{min}(H_{ll}) - \mathcal{O}(max(d_l, d-d_l)), \lambda_{max}(H_{ll}) + \mathcal{O}(max(d_l, d-d_l))]$

□

The general versions of the Gershgorin theorem to block matrices and is studied in [2, 6, 8]. We presented the proof for the Hessian matrices, which are symmetric, by simply extending the usual Gershgorin circle theorem to block matrices. We bound the eigenvalues of the Hessian in terms of the min and max eigenvalues of the layerwise Hessian.

We consider the C class classification problem. The training set $S = \{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^{N}$ is considered, where each $\mathbf{x}^i \in \mathbb{R}^D$ or $\mathbf{x}^i \in \mathbb{R}^{C\times H\times W}$ and $\mathbf{y}^i \in \{0,1\}^C$ is drawn iid from the distribution $\mathcal{D}$. We then consider an L-layer neural network with ReLU non-linearity, where the network outputs the logits $\mathbf{z}^i$. The logits are obtained by a series of fully connected (FC)/convolutional (CONV) layers, followed by a non-linearity, represented concisely by Eq. 16 for an FC layer with parameters $(\{\mathbf{W}_l, \mathbf{b}_l\})$ and Eq. 17 for a CONV layer with parameters $(\{\mathbf{W}_l, \mathbf{b}_l\})$. We denote the collection of all the model parameters as $\theta := \{\mathbf{W}_1, \mathbf{b}_1 ... \mathbf{W}_L, \mathbf{b}_L\}$ and $\theta \in \mathbb{R}^d$ where all the model parameters rolled into a

single vector of dimension d. Let $f_\theta(\mathbf{x^i})$ denote the final layer output of the model

$$\mathbf{z}_l^i = \text{FC}(\mathbf{a}_{l-1}^i; \{\mathbf{W}_l, \mathbf{b}_l\}) \tag{16}$$

$$\mathbf{z}_l^i = \text{CONV}(\mathbf{a}_{l-1}^i; \{\mathbf{W}_l, \mathbf{b}_l\}) \tag{17}$$

$$\mathbf{a}_l^i = \sigma(\mathbf{z}_l^i) \tag{18}$$

where $\sigma(.)$ denotes non-linearity $\mathbf{a}_0 = \mathbf{x}^i$, $\mathbf{z}^i = \mathbf{a}_L^i = f_\theta(\mathbf{x^i})$. Finally, we use the cross-entropy loss

$$\mathcal{L}(\mathbf{y}^i, \mathbf{z}^i) = \sum_{c=1}^{C} -\mathbf{y}^i[c] log(\hat{\mathbf{y}}^i[c]). \tag{19}$$

where $\hat{\mathbf{y}}$ is the softmax on logits $\mathbf{z}^i$ as

$$\hat{\mathbf{y}} = exp(\mathbf{z}^i) / \sum_{m=1}^{C} exp(\mathbf{z}^i[m])) \tag{20}$$

where $\mathbf{x}^i$,$\mathbf{y}^i$ denotes the i^{th} input sample and label respectively. We use the notation $\mathcal{L}^i$ for $\mathcal{L}(\mathbf{y}^i, \mathbf{z}^i)$. The overall Loss computed on Batch size of B is denoted by

$$\mathcal{L} = \frac{1}{B} \sum_{i=1}^{B} \mathcal{L}^i \tag{21}$$

We have the following Lemma due to [9]. We denote $\theta \in \mathbf{R}^d$ as the collection of all the parameters, where d denotes the total number of parameters.

Lemma 1. *For the Network described in Eq. 16 to Eq. 21. The Hessian of loss $\mathcal{L}^i$ with respect to weights of FC layer $\mathbf{W}_l$ denoted by $H_{\mathbf{W}_l}(\mathcal{L}^i)$ is given by $H_{\mathbf{W}_l}(\mathcal{L}^i) = \mathbf{M}_l(x^i, \theta) \otimes \mathbf{a}_{l-1}^i \mathbf{a}^{i\top}_{l-1}$, where $\mathbf{M}_l(x^i)$ is a symmetric matrix.*

Proof. For the detailed derivation, please refer to Appendix A1 of [9]. We present the proof for the sake of completeness. We fix a layer l for which we want to compute the Hessian, the inputs to layer l is given by $\mathbf{a}_{l-1}^i$. The layer l is parameterized by $\mathbf{W}_l$ and $\mathbf{b}_l$.

$$\mathbf{z^i}_l = \mathbf{W}_l \mathbf{a^i}_{l-1} + \mathbf{b}_l \tag{22}$$

By using the chain rule for Hessian as [7, 9] we get the following.

$$H_{\mathbf{w}_l}(\mathcal{L}^i) = \frac{\partial \mathbf{z}_l^i}{\partial \mathbf{w}_l}^\top H_{\mathbf{z}^i}(\mathcal{L}^i) \frac{\partial \mathbf{z}_l^i}{\partial \mathbf{w}_l} + \sum_{n=1}^{d_l} \frac{\partial l(\mathbf{z}^i, \mathbf{y}^i)}{\partial \mathbf{z}^i[n]} \nabla^2_{\mathbf{w}_l} \mathbf{z}^i[n] \tag{23}$$

Here $\mathbf{w}_l := vec(\mathbf{W}_l)$, $\mathbf{z}^i[n]$ is the n^{th} element of the vector $\mathbf{z}^i$. $\nabla^2_{\mathbf{w}_l} \mathbf{z}^i[n]$ is Hessian of $\mathbf{z}^i[n]$ w.r.t $\mathbf{w}_l$. Also note that by convention $H_{\mathbf{w}_l}(\mathcal{L}^i) := H_{\mathbf{W}_l}(\mathcal{L}^i)$ as we are only concerned with the Hessian of the loss w.r.t to the parameters of the layer l not the structure in which these parameters are present.

From 22 we get the following

$$\frac{\partial \mathbf{z}_l^i}{\partial \mathbf{w}_l} = \mathbf{I}_{d_l} \otimes \mathbf{a}_{l-1}^{i\top} \tag{24}$$

It is easy to see that $\nabla^2_{\mathbf{w}_l} \mathbf{z}^i[n] = 0$ and from 24 we have the following

$$H_{\mathbf{w}_l}(\mathcal{L}^i) = (\mathbf{I}_{d_l} \otimes \mathbf{a}_{l-1}^i) H_{\mathbf{z}^i}(\mathcal{L}^i) (\mathbf{I}_{d_l} \otimes \mathbf{a}_{l-1}^{i\top}) \tag{25}$$

The above equation can be simplified as

$$H_{\mathbf{w}_l}(\mathcal{L}^i) = \mathbf{M}_l(x^i, \theta) \otimes \mathbf{a}_{l-1}^i \mathbf{a}_{l-1}^{i\top} \tag{26}$$

where $\mathbf{M}_l(x^i, \theta) := H_{\mathbf{z}^i}(\mathcal{L}^i)$. It can be seen that $\mathbf{M}_l(x^i, \theta)$ is a symmetric matrix by definition. This concludes the proof □

Consider the CONV layer with input feature map of dimension $\mathbf{a}_{l-1}^i \in \mathbb{R}^{C_{l-1} \times H_{l-1} \times W_{l-1}}$, the output feature map $\mathbf{z}_l^i \in \mathbb{R}^{m \times H_l \times W_l}$ and convolutional kernel $\mathbf{W}_l \in \mathbb{R}^{m \times C_{l-1} \times K_1 \times K_2}$, we then have the following Lemma due to [9].

We now state the two of our results that relate the layer-wise top eigenvalues to the activation norm of each layer. Consider a FC layer as in Eq. 16 with $\mathbf{a}_{l-1}^i \in \mathbb{R}^{d_{l-1}}$ and weights $\mathbf{W}_l \in \mathbb{R}^{d_l \times d_{l-1}}$. We then have the following result.

Theorem 2. *If $\|\theta\|_2 \leq \tilde{B}$ then the top eigenvalue of layer-wise Hessian for the loss $\mathcal{L}$ w.r.t to $\mathbf{W}_l$ denoted by $\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}))$ for l = 2 to L, computed over the batch of samples for a L layered fully connected neural network for multi-class classification is given by $\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \alpha_l \sum_{i \in B} \|\mathbf{a}_{l-1}^i\|_2^2$ where $\alpha_l > 0$.*

Proof. We use the results from the previous Lemma's and the fact that Hessian for the batch is the average of Hessian of all the individual samples.

$$\mathcal{L} = \frac{1}{B} \sum_{i \in B} \mathcal{L}^i \tag{27}$$

$$\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}) = \frac{1}{B} \sum_{i \in B} \mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i) \tag{28}$$

By repeated application of Eq. 6 to Eq. 28, we have the following.

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \frac{1}{B} \sum_{i \in B} \lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i)) \tag{29}$$

From the Lemma 1 we have the following

$$H_{\mathbf{W}_l}(\mathcal{L}^i) = \mathbf{M}_l(x^i, \theta) \otimes \mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1} \tag{30}$$

By using Eq. 5 in the above Eq. 30 we get the following.

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i)) = \lambda_{max}(\mathbf{M}_l(x^i, \theta)) \lambda_{max}(\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}) \tag{31}$$

We now show that the $\lambda_{max}(\mathbf{M}_l(\mathbf{x}^i, \theta))$ exists and its finite in the following arguments.

From [4], we know that the eigenvalues are the continuous functions of the coefficients of characteristic polynomials, and so is the top eigenvalue.

Since every entry in the matrix $\mathbf{M}_l(\mathbf{x}^i, \theta)$ is a continuous function of θ. The coefficients of the characteristic polynomials are also continuous functions of θ. Since the continuity is preserved under the composition of continuous functions, i.e., if f is continuous and g is continuous, then the composition fog is continuous.

If top eigenvalue λ_{max} is a continuous function of the coefficients of characteristic polynomial and the coefficients are again a continuous function of the variable θ. We conclude that λ_{max} is a continuous function θ.
The set $\{\theta : \|\theta\|_2 \leq \tilde{B}\}$ where $\theta \in \mathbb{R}^d$ is compact. Continuous function map compact sets to compact sets. Thus the function λ_{max} attains its supremum and its finite.

$$\lambda_{max}(\mathbf{M}_l(\mathbf{x}^i, \theta)) \leq \sup_{\theta}(\lambda_{max}(\mathbf{M}_l(\mathbf{x}^i, \theta))) = \alpha^i_l \tag{32}$$

We note that $\alpha^i_l > 0$, suppose if its negative, by 31 we see that top eigenvalue of layerwise Hessian is negative; this implies the loss function is concave, which contradicts the fact that neural-networks are non-convex and non-concave functions.

By using the bound in Eq. 32 in Eq. 31, we can bound the Eq. 29 as below.

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \frac{1}{B} \sum_{i \in B} \alpha^i_l \|\mathbf{a}^i_{l-1}\|^2_F \tag{33}$$

we have the following, where α_l is the maximum over all the training samples α^i_l.

$$\alpha^i_l \leq \max_i(\alpha^i_l) = \alpha_l \tag{34}$$

Using Eq. 34 in the Eq. 33 we get the following

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \frac{1}{B} \alpha_l \sum_{i \in B} \|\mathbf{a}^i_{l-1}\|^2_2 \tag{35}$$

This completes the proof. □

Lemma 2. *For the Network described in Eq. 16 to Eq. 21. The Hessian of loss $\mathcal{L}^i$ with respect to weights of CONV layer $\mathbf{W}_l$ denoted by $H_{\mathbf{W}_l}(\mathcal{L}^i)$ is approximated by $H_{\mathbf{W}_l}(\mathcal{L}^i) \approx \tilde{\mathbf{M}}_l(x^i) \otimes \mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}$.*

Proof. For a detailed discussion, please refer to Appendix A.2 of [9]. □

Theorem 3. *If $\|\theta\|_2 \leq \tilde{B}$, the top eigenvalue of layerwise Hessians for the loss (w.r.t to $\boldsymbol{W}_l$ for l = 2 to L) computed over the Batch of samples for a L layered convolutional neural network for multi-class classification is given by $\lambda_{max}(\mathbf{H}_{\boldsymbol{W}_l}(\mathcal{L})) \leq \alpha_l \sum_{i \in B} \|\boldsymbol{a}^i_{l-1}\|^2_F$ where where $\alpha_l > 0$.*

Proof. In the proof technique, we follow the exact similar steps as the above theorem with some minor changes. The major change here is we now use the convolutional layers.

$$\mathcal{L} = \frac{1}{B} \sum_{i \in B} \mathcal{L}^i \tag{36}$$

$$\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}) = \frac{1}{B} \sum_{i \in B} \mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i) \tag{37}$$

By repeated application of Eq. 6 to the Eq. 37 we have the following.

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \frac{1}{B} \sum_{i \in B} \lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i)) \tag{38}$$

From the Lemma 2 we have the following

$$H_{\mathbf{W}_l}(\mathcal{L}^i) = \tilde{\mathbf{M}}_l(x^i) \otimes \mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1} \tag{39}$$

By using Eq. 5 in the above Eq. 39 we get the following.

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i)) = \lambda_{max}(\tilde{\mathbf{M}}_l(x^i)) \lambda_{max}(\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}) \tag{40}$$

We cannot use Eq. 1 directly to find the eigenvalue of $\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}$ as $\mathbf{a}^i_{l-1}$ is matrix not a vector, because we are dealing with convolutional layers. Since $\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}$ is a positive semi-deifinite matrix we have the following inequality

$$\lambda_{max}(\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}) \leq \text{Trace}(\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}) \tag{41}$$

By using the identity $\text{Trace}(\mathbf{a}^i_{l-1} \mathbf{a}^{i\top}_{l-1}) = \|\mathbf{a}^i_{l-1}\|^2_F$ in the Eq. 40 we get the following

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L}^i)) \leq \lambda_{max}(\tilde{\mathbf{M}}_l(x^i)) \|\mathbf{a}^i_{l-1}\|^2_F \tag{42}$$

We can use similar reasoning as in Theorem 2 to bound the value of the $\lambda_{max}(\tilde{\mathbf{M}}_l(x^i))$.

By substituting the above inequality 42 in 38 we get the following.

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \frac{1}{B} \sum_{i \in B} \alpha^i_l \|\mathbf{a}^i_{l-1}\|^2_F \tag{43}$$

where we have used the fact $\lambda_{max}(\tilde{\mathbf{M}}_l(x^i)) \leq \alpha_l^i$.

If we denote α_l as the maximum overall α_l^i over the batch. We then get the following

$$\lambda_{max}(\mathbf{H}_{\mathbf{W}_l}(\mathcal{L})) \leq \frac{1}{B}\alpha_l \sum_{i \in B} \|\mathbf{a}_{l-1}^i\|_F^2 \tag{44}$$

This completes the proof. □

3. Model Architectures

In Table 1, the model architecture is shown. We use PyTorch style representation. For example convolutional (CONV) layer(3,64,5) means 3 input channels, 64 output channels and the kernel size is 5. Maxpool(2,2) represents the kernel size of 2 and a stride of 2. Fully Connected (FC)(384,200) represents an input dimension of 384 and an output dimension of 200. The architecture for CIFAR-100 is exactly the same as used in [1].

Table 1. Models used for Tiny-ImageNet and CIFAR-100 datasets.

CIFAR-100 Model	**Tiny-ImageNet Model**
	ConvLayer(3,64,3)
	GroupNorm(4,64)
	Relu
	MaxPool(2,2)
	ConvLayer(64,64,3)
	GroupNorm(4,64)
ConvLayer(3,64,5)	Relu
Relu	MaxPool(2,2)
MaxPool(2,2)	ConvLayer(64,64,3)
ConvLayer(64,64,5)	GroupNorm(4,64)
Relu	Relu
MaxPool(2,2)	MaxPool(2,2)
Flatten	Flatten
FullyConnected(1600,384)	FullyConnected(4096,512)
Relu	Relu
FullyConnected(384,192)	FullyConnected(512,384)
Relu	Relu
FullyConnected(192,100)	FullyConnected(384,200)

4. Sensitivity to hyper-parameter ζ

In figure 1, we perform sensitivity analysis on the hyper-parameter ζ i.e. how model accuracy varies over different values of ζ. We consider the algorithm FedAvg+ MAN on CIFAR-100 dataset with Dirichlet-based non-iid data partition ($\delta = 0.3$). We observe that accuracy is stable over $\zeta \in \{0.1, 1.5\}$.

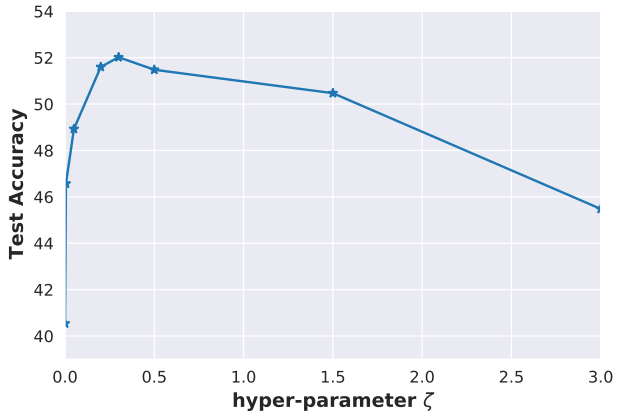


Figure 1. Sensitivity of Accuracy to the hyper-parameter ζ. It can be seen that accuracy is stable over $\zeta \in \{0.1, 3.0\}$

5. Additional Results

5.1. Results for FedSAM/ASAM and FedSpeed with and without MAN

We have developed all our experiments based on the open source code provided by [1]. In the figure 2, we provide the plots for communication rounds vs Accuracy on CIFAR-100 dataset for FedSAM, FedASAM and FedSpeed with and without using our MAN regularizer. It can be seen that MAN regularizer consistently improves the performance of all the algorithms. Similar results for Tiny-ImageNet are provided in the figure 3. We can observe consistent improvement in the performance of the algorithms when MAN regularizer is added.

5.2. CIFAR-10 Results

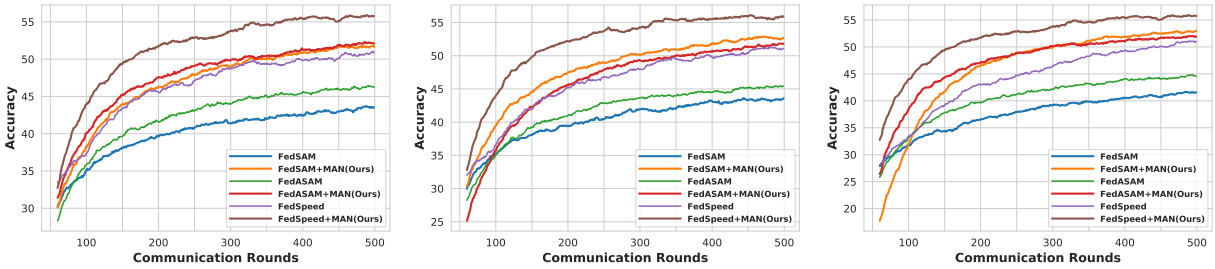
In this section we provide the results for CIFAR-10 dataset. In the table 2 we report the performance of all the algorithms (FedAvg,FedDyn,FedDC, FedSAM,FedASAM and FedSpeed) with and without using the MAN. It can be clearly seen that the performance of all the algorithms can be improved when our MAN regularizer is used with the algorithms.

5.3. Empirical Hessian Analysis

In Table 3 we present the top eigenvalue and the trace of the Hessian of the loss of global model for FedASAM, FedSpeed and their improved versions using MAN i.e, FedASAM+MAN and FedSpeed+MAN.

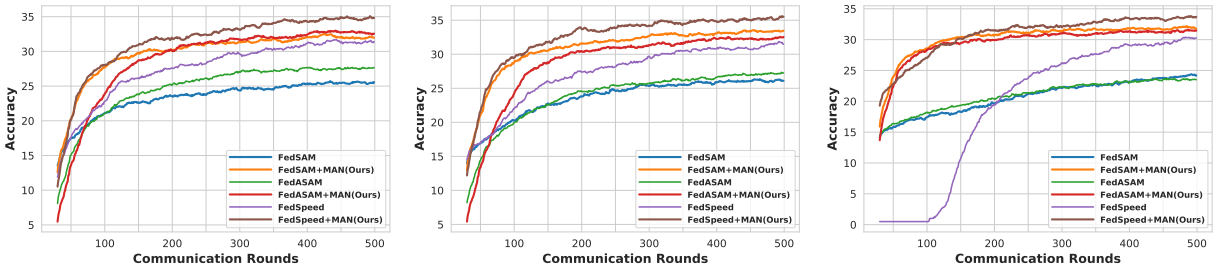
6. Non-iid Data generation

We now briefly describe how the data is generated using the Dirichlet distribution. This distribution is parameterized by parameter δ. For every client, we sample a vector for the



(a) $\delta = 0.3$ (b) $\delta = 0.6$ (c) iid

Figure 2. Convergence Comparison for CIFAR-100: We compare performance of the algorithms FedAvg, FedDyn, FedDC and the proposed FedAvg+MAN, FedDyn+MAN and FedDC+MAN for 500 communication rounds. It can be clearly seen that proposed approach significantly improves the existing algorithms across the communication rounds.



(a) $\delta = 0.3$ (b) $\delta = 0.6$ (c) iid

Figure 3. Convergence Comparison for Tiny-ImageNet: We compare the performance of the algorithms FedAvg, FedDyn, FedDC and the proposed FedAvg+MAN, FedDyn+MAN, and FedDC+MAN for 500 communication rounds. It can be clearly seen that the proposed approach significantly improves the existing algorithms.

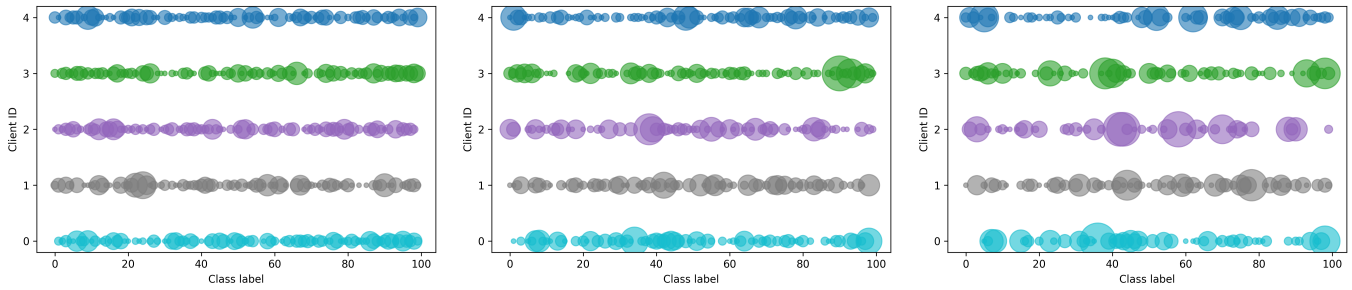
Table 2. Comparison of various methods with and without MAN regularizer with different degrees of heterogeneity on CIFAR-10 dataset. MAN clearly improves the performance consistently across all the methods. All the experiments are repeated for three different initializations and their mean and standard deviations are reported.

Method	CIFAR10		
	$\delta = 0.6$	$\delta = 0.3$	iid
FedAvg	79.34 ± 0.19	80.19 ± 0.46	81.44 ± 0.43
FedAvg+MAN	**82.53** ± 0.25	**83.35** ± 0.09	**84.18** ± 0.15
FedSAM	80.42 ± 0.47	81.40 ± 0.17	82.54 ± 0.14
FedSAM+MAN	**81.56** ± 0.1	**82.54** ± 0.16	**84.19** ± 0.26
FedASAM	79.90 ± 0.43	80.83 ± 0.08	82.27 ± 0.45
FedASAM+MAN	**80.81** ± 0.06	**81.85** ± 0.08	**84.22** ± 0.1
FedDyn	82.00 ± 0.22	82.53 ± 0.06	84.16 ± 0.41
FedDyn+MAN	**83.84** ± 0.38	**84.63** ± 0.17	**84.80** ± 0.25
FedDC	83.10 ± 0.37	83.64 ± 0.13	84.8 ± 0.21
FedDC+MAN	**83.27** ± 0.18	**83.59** ± 0.15	**85.08** ± 0.1
FedSpeed	84.06 ± 0.11	84.24 ± 0.14	85.14 ± 0.3
FedSpeed+MAN	**84.37** ± 0.25	**84.82** ± 0.16	**85.95** ± 0.24

Table 3. Comparison of top eigenvalues and trace of the algorithms with and without MAN regularizer, lower values are better. We can observe that by augmenting MAN regularization, i.e., FedASAM+MAN, FedSpeed+MAN, we obtain lower trace and lower top eigenvalues, which is indicative of flat minimum, and hence, it attains better accuracy.

Method	CIFAR-100			
	$\delta = 0.3$		$\delta = 0.6$	
	Top eigenvalue	Trace	Top eigenvalue	Trace
FedASAM	51.49	8744	53.39	9056
FedASAM+MAN	**43.80**	**4397**	**42.00**	**4747**
FedSpeed	47.31	6463	49.75	6519
FedSpeed+MAN	**40.41**	**3806**	**37.64**	**3674**

Dirichlet distribution. This vector is of the length of a total number of classes and represents the label distribution of the clients. The lower value of δ implies high heterogeneity, i.e.; label distribution is non-uniform; only a few labels dominate the samples on the client's data. We demonstrate this behavior in Figure 4, where the label distribution of 5 clients is drawn from the Dirichlet distribution by varying



(a) $\delta = 1.0$ (b) $\delta = 0.6$ (c) $\delta = 0.3$

Figure 4. Label distribution of 5 clients based on Dirichlet distribution for CIFAR-100 dataset is shown for $\delta = 1.0$, $\delta = 0.6$ and $\delta = 0.3$. The degree of heterogeneity increases as the value of δ decreases. Each client gets 500 samples, and for $\delta = 1.0$ the client receives all the labels more uniformly compared to the case of $\delta = 0.3$ where only a few labels are dominant.

the parameter δ. We can observe as the value of δ decreases, the label distribution across the clients become more non-uniform. The dataset we have used is CIFAR-100, so the label distribution has support over 100 classes.

7. Hyper-parameter settings

All the algorithms use a learning rate of 0.1, batch size of 50, client participation of 10%, and a gradient clipping threshold of 10. We use 5 local epochs for client training learning rate decay of 0.998 for every round was used.

For FedAvg+MAN, we use $\zeta = 0.6$ by default. For FedDyn, we use $\alpha = 0.01$. For FedDC also uses $\alpha = 0.01$. For FedSpeed, we use $\rho = 0.1$ for non-iid settings and $\rho = 0.01$ for iid setting, $\beta = 1.0$,$\gamma = 1.0$ and no gradient cutoff threshold to 0.05. For FedSAM we use $\rho = 0.05$. Only for Tiny-ImageNet with iid partition, we set $\rho = 0.03$. When we use MAN, i.e, FedSAM+MAN, we set $\rho = 0.01$ for Tiny-ImageNet with iid partition, and weight decay of $1e{-}3$ is used. For FedASAM we use $\rho = 0.5$ and $\eta = 0.2$. Only for iid partition we set the values to $\rho = 0.1$ and $\eta = 0.2$.

8. Algorithm details of FedAvg+MAN, FedDyn+MAN and FedDC+MAN

We present the algorithm details for implementing FedAvg+MAN, FedDyn+MAN, and FedDC+MAN in the Algorithms 1, 2 and 3 respectively. Each client minimizes the activation norm as a regularizer in the algorithm and the cross-entropy loss, as shown in the below Eq. 45.

$$f_k(\mathbf{w}) \triangleq L_k(\mathbf{w}) + \zeta L_k^{act}(\mathbf{w}) \tag{45}$$

$L_k(\mathbf{w})$ denotes the task-specific loss in our case, it is (cross-entropy loss) and $L_k^{act}(\mathbf{w})$ is the activation norm loss that is used to attain flatness and is described in detail in the Sec.3.2.3 of the main paper. The hyper-parameter ζ trades off between the flatness and the cross-entropy loss. In this way, it is straightforward to integrate the proposed regularizer 'MAN' into the existing FL algorithms. The complete details of individual algorithms can be found in FedAvg [5], FedDyn [1] and FedDC [3]. We can similarly extend the MAN to FedSAM/ASAM and FedSpeed as well. We simply add activation norms to the client loss function as in Eq. 45.

Algorithm 1 FedAvg+MAN

Server Executes
Initialize $\mathbf{w}^t$
for every communication round t in T **do**
  sample random subset S of clients, $S \subset [m]$
  for every client k in S in parallel **do**
    $\mathbf{w}_k^t$ = **ClientUpdate**(k,$\mathbf{w}^{t-1}$)
  end for
  $\mathbf{w}^t$ = **ServerAggregation**($\mathbf{w}_k^t$)
end for
procedure CLIENTUPDATE(k,$\mathbf{w}^{t-1}$)
  set $\mathbf{w}_k^t = \mathbf{w}^{t-1}$
  for every epoch e in E **do**
    for every batch b in B **do**
      Compute $f_k(\mathbf{w})$
      $\mathbf{w}_k^t = \mathbf{w}_k^t - \nabla f_k(\mathbf{w}_k^t)$
    end for
  end for
  return $\mathbf{w}_k^t$
end procedure
procedure SERVERAGGREGATION($\mathbf{w}_k^t$)
  $\mathbf{w}^t = \sum_k \frac{n_k}{n} \mathbf{w}_k^t$
end procedure

References

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *Interna-*

Algorithm 2 FedDyn+MAN

Input T, $\mathbf{w}^0$, α, $\nabla G_k(\mathbf{w}_k^0) = \mathbf{0}$
Initialize $\mathbf{w}^t$
for every communication round t in T **do**
  sample random subset S of clients, $S \subset [m]$
  for every client k in S in parallel **do**
    w_t^k = **ClientUpdate**(k,w_{t-1})
  end for
  w_t = **ServerAggregation**(w_t^k)
end for
procedure CLIENTUPDATE(k,w_{t-1})
  set $\mathbf{w}_k^t = \underset{w}{arg\,min} G_k(\mathbf{w}) - \langle \nabla G_k(\mathbf{w}_k^{t-1}), \mathbf{w} \rangle + \frac{\alpha}{2} \|\mathbf{w} - \mathbf{w^{t-1}}\|^2$
  set $\nabla G_k(\mathbf{w}_k^t) = \nabla G_k(\mathbf{w}_k^{t-1}) - \alpha(\mathbf{w}_k^t - \mathbf{w}^{t-1})$
  return w_t^k
end procedure
procedure SERVERUPDATE(k,w_{t-1})
  set $\mathbf{h}^t = \mathbf{h}^{t-1} - \frac{\alpha}{m}(\sum_{k \in S} \mathbf{w}_k^t - \mathbf{w}^{t-1})$
  set $\mathbf{w}^t = \frac{1}{|S|}(\sum_{k \in S} \mathbf{w}_k^t - \frac{1}{\alpha}\mathbf{h}^t)$
  return $\mathbf{w}_t$
end procedure

Algorithm 3 FedDC+MAN

Input T, $\mathbf{w}^0$, α, $\nabla G_k(\mathbf{w}_k^0) = \mathbf{0}$
Initialize $\mathbf{w}^t$
for every communication round t in T **do**
  sample random subset S of clients, $S \subset [m]$
  for every client k in S in parallel **do**
    w_t^k = **ClientUpdate**(k,w_{t-1})
  end for
  w_t = **ServerAggregation**(w_t^k)
end for
procedure CLIENTUPDATE(k,w_{t-1})
  set $\mathbf{w}_k = \mathbf{w}^{t-1}$
  for every epoch e in E **do**
    $\mathbf{w}_k = \mathbf{w}_k - \eta \nabla f_k(\mathbf{w}_k^t, h_k, D_k, \mathbf{w}_{t-1})$
  end for
  Set the local client drift $\Delta \mathbf{w}_k = \mathbf{w}_k - \mathbf{w}^{t-1}$
  Update the local drift $\mathbf{h}_k = \mathbf{h}_k + \Delta \mathbf{w}_k$
  Return $\mathbf{w}_k^t$, $\mathbf{h}_k$
end procedure
procedure SERVERUPDATE(k,w_{t-1})
  Update global model $\mathbf{w}^t = \frac{1}{|S|} \sum_{k \in S} (\mathbf{w}_k^t + \mathbf{h_k})$
  Update global drift $\Delta \mathbf{w} = \frac{1}{|S|}(\sum_{k \in S} \Delta \mathbf{w}_k$
  return $\mathbf{w}^t$, $\Delta \mathbf{w}$
end procedure

tional Conference on Learning Representations. 5, 7

[2] David G Feingold and Richard S Varga. Block diagonally dominant matrices and generalizations of the gerschgorin circle theorem. 1962. 2

[3] Liang Gao, Huazhu Fu, Li Li, Yingwen Chen, Ming Xu, and Cheng-Zhong Xu. Feddc: Federated learning with non-iid data via local drift decoupling and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10112–10121, June 2022. 7

[4] Gary Harris and Clyde Martin. The roots of a polynomial vary continuously as a function of the coefficients. *Proceedings of the American Mathematical Society*, 100:390–392, 06 1987. 4

[5] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 7

[6] Hector N Salas. Gershgorin's theorem for matrices of operators. *Linear algebra and its applications*, 291(1-3):15–36, 1999. 2

[7] Maciej Skorski. Chain rules for hessian and higher derivatives made easy by tensor calculus. *arXiv preprint arXiv:1911.13292*, 2019. 3

[8] Christiane Tretter. *Spectral theory of block operator matrices and applications*. World Scientific, 2008. 2

[9] Yikai Wu, Xingyu Zhu, Chenwei Wu, Annie Wang, and Rong Ge. Dissecting hessian: Understanding common structure of hessian in neural networks. *arXiv preprint arXiv:2010.04261*, 2020. 3, 4