Improving Vertical Federated Learning by Efficient Communication with ADMM

Chulin Xie[1] Pin-Yu Chen[2] Ce Zhang[3] Bo Li[1]

[1]UIUC, [2]IBM Research, [3]ETH Zurich

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

IBM Research

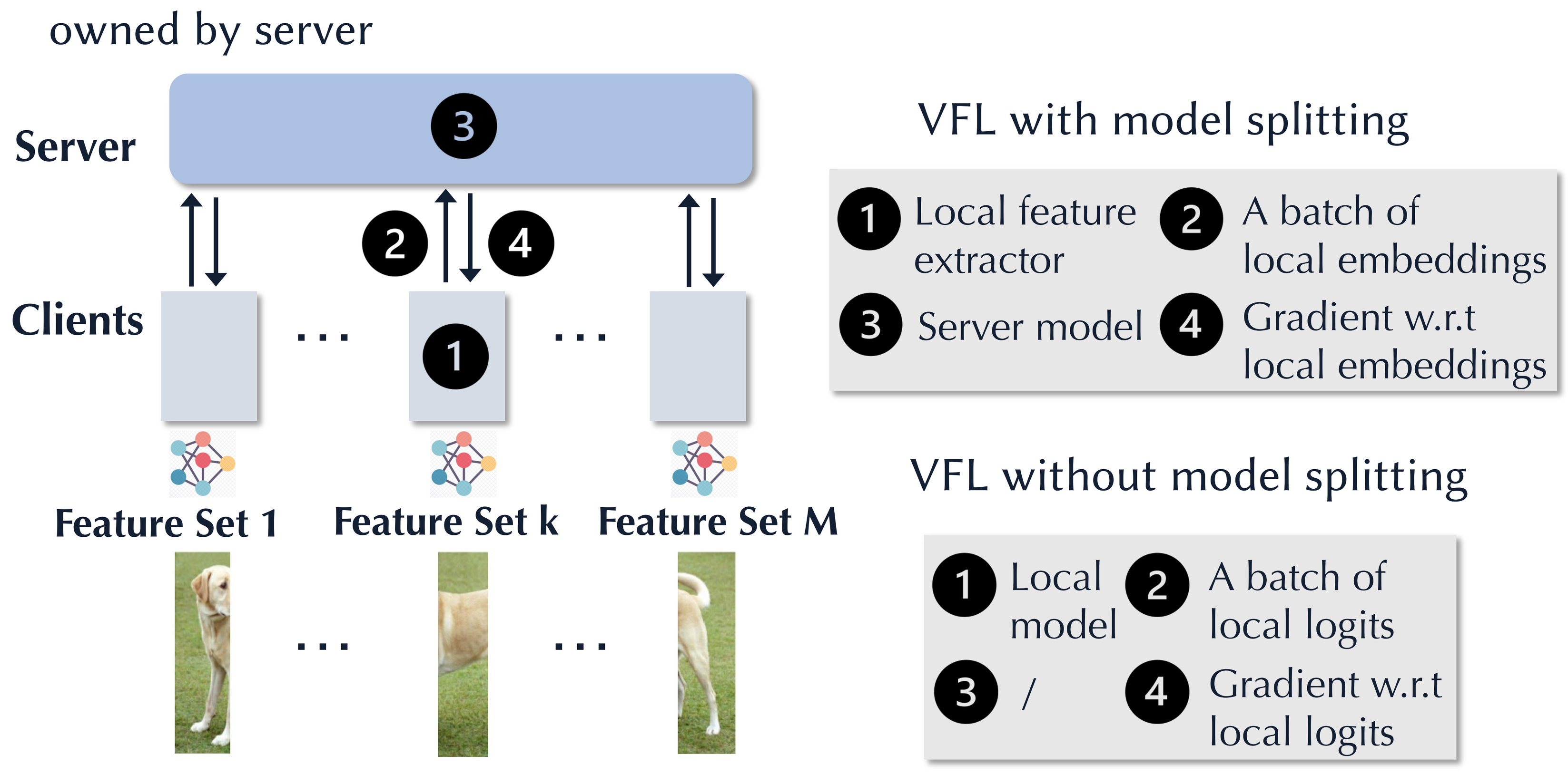
ETH zürich

NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

Vertical Federated Learning (VFL):

- Features of the samples are partitioned across clients and the labels are owned by server



Challenges:

- **Aggregation**: *averaging* local embeddings fails to capture the unique properties of each client.
- **Communication**: communicating gradients for *each* training step incurs high costs.

Our contributions:

✓**Framework**: an effective framework with multiple heads (**VIM**).
✓**Algorithm**: an ADMM-based method (**VIMADMM**) reducing communication costs by allowing multiple local updates at each step.
✓**Empirical study**: 1) VIMADMM converges faster and achieves higher accuracy; 2) client-level explanation under VIM based on the linear heads.

VFL with Multiple Heads (VIM)

- **VFL setting (with model splitting setting)**
- The features of one sample $\{x_j^1, x_j^2, \ldots, x_j^M\}$ is distributed to M clients
- Each client has a local feature set $X_k = \{x_j^k\}_{j=1}^N$ and uploads local embeddings $f(x_j^k; \theta_k)$
- Server aggregates local embeddings and computes gradient with labels $\{y_j\}_{j=1}^N$
- **Challenges**: using averaged local embeddings as server model input [1] might lose the unique aspects of each local feature set.
- **Our idea**: server learns a model with multiple linear heads $W_1, W_2, \ldots, W_M$ corresponding to local clients, taking their separate contribution into account

$$\min_{\{W_k\}_{k=1}^M, \{\theta_k\}_{k=1}^M} \sum_{j=1}^N \ell\Big(\sum_{k=1}^M \underbrace{f(x_j^k; \theta_k)}_{\text{Local embedding}} \underbrace{W_k}_{\text{Server heads}}, y_j\Big) + \underbrace{\sum_{k=1}^M \beta_k \mathcal{R}_k(\theta_k) + \sum_{k=1}^M \beta_k \mathcal{R}_k(W_k)}_{\text{Regularization}}$$

Solving VIM with ADMM-based method

- **Key idea:** Multiple heads in VIM enable the alternating direction method of multipliers (ADMM) via a special decomposition into simpler sub-problems that can be solved in a distributed manner.
- **Formulation:** an equivalent constrained optimization problem

$$\sum_{j=1}^N \ell(\underbrace{z_j}_{\text{auxiliary variables}}, y_j) + \sum_{k=1}^M \beta_k \mathcal{R}_k(\theta_k) + \sum_{k=1}^M \beta_k \mathcal{R}_k(W_k)$$

$$\text{s.t. } \underbrace{\sum_{k=1}^M f(x_j^k; \theta_k) W_k - z_j = 0}_{\text{a consensus between the server's output and auxiliary variable}}, \forall j \in [N]$$

- **Augmented Lagrangian:**

$$\sum_{j=1}^N \ell(z_j, y_j) + \sum_{k=1}^M \beta_k \left(\mathcal{R}_k(\theta_k) + \mathcal{R}_k(W_k)\right) + \sum_{j=1}^N \underbrace{\lambda_j^\top}_{\text{dual variables}} \left(\sum_{k=1}^M f(x_j^k; \theta_k) W_k - z_j\right) + \underbrace{\frac{\rho}{2}}_{\text{constant penalty factor}} \sum_{j=1}^N \left\| \sum_{k=1}^M f(x_j^k; \theta_k) W_k - z_j \right\|_F^2$$

- **Solution:**

❖ Decomposes the problem into four sets of sub-problems over $\{W_k\}$, $\{\theta_k\}$, $\{z_j\}$, $\{\lambda_j\}$
❖ Alternatively updating in server and clients
❖ Each sub-problem set can be solved in parallel across M clients or N samples.

$$\left.\begin{aligned} z_j^{(t+1)} &= \operatorname*{argmin}_{z_j} \mathcal{L}(\{\theta_k^{(t+1)}\}, \{W_k^{(t+1)}\}, z_j, \{\lambda_{j'}^{(t)}\}), \forall j \in [N], \\ \lambda_j^{(t+1)} &= \operatorname*{argmin}_{\lambda_j} \mathcal{L}(\{\theta_k^{(t+1)}\}, \{W_k^{(t+1)}\}, \{z_{j'}^{(t+1)}\}, \lambda_j), \forall j \in [N], \\ W_k^{(t+1)} &= \operatorname*{argmin}_{W_k} \mathcal{L}(\{\theta_{k'}^{(t)}\}, W_k, \{z_j^{(t)}\}, \{\lambda_j^{(t)}\}), \forall k \in [M], \end{aligned}\right\} \text{server}$$

$$\left.\theta_k^{(t+1)} = \operatorname*{argmin}_{\theta_k} \mathcal{L}(\theta_k, \{W_{k'}^{(t+1)}\}, \{z_j^{(t)}\}, \{\lambda_j^{(t)}\}), \forall k \in [M],\right\} \text{client}$$

VIMADMM Workflow

- **Key steps of VIMADMM** at each round t:

(1) Communication from client to server: a batch of local embeddings $\{h_j^{k(t)}\}_{j \in B(t)}$

(2) Sever updates auxiliary variables:

$$z_j^{(t)} = \operatorname*{argmin}_{z_j} \ \ell(z_j, y_j) - \lambda_j^{(t-1)\top} z_j + \frac{\rho}{2} \left\| \sum_{k=1}^M h_j^{k(t)} W_k^{(t)} - z_j \right\|_F^2, \forall j \in B(t)$$

(3) Sever updates dual variables:

$$\lambda_j^{(t)} = \lambda_j^{(t-1)} + \rho \left(\sum_{k=1}^M h_j^{k(t)} W_k^{(t)} - z_j^{(t)} \right), \forall j \in B(t)$$

(4) Sever updates linear heads:

$$W_k^{(t+1)} = \operatorname*{argmin}_{W_k} \ \beta \mathcal{R}(W_k) + \textstyle\sum_{j \in B(t)} \lambda_j^{(t)\top} h_j^{k(t)} W_k + \sum_{j \in B(t)} \frac{\rho}{2} \left\| \sum_{i \in [M], i \neq k} h_j^{i(t)} W_i^{(t)} + h_j^{k(t)} W_k - z_j^{(t)} \right\|_F^2, \forall k \in [M]$$

(5) Communication from server to each client: residual variable, dual variables, and one *corresponding* linear head

$$s_j^{k(t+1)} \triangleq z_j^{(t)} - \sum_{i \in [M], i \neq k} h_j^{i(t)} W_i^{(t+1)}, \forall j \in B(t), \forall k \in [M]$$

(6) Client updates local model parameters:

$$\theta_k^{(t+1)} = \operatorname*{argmin}_{\theta_k} \ \beta \mathcal{R}(\theta_k) + \textstyle\sum_{j \in B(t)} \lambda_j^{(t+1)\top} f(x_j^k; \theta_k) W_k^{(t+1)} + \frac{\rho}{2} \sum_{j \in B(t)} \left\| s_j^{k(t+1)} - f(x_j^k; \theta_k) W_k^{(t+1)} \right\|_F^2$$

Solved by **multiple** local SGD steps.

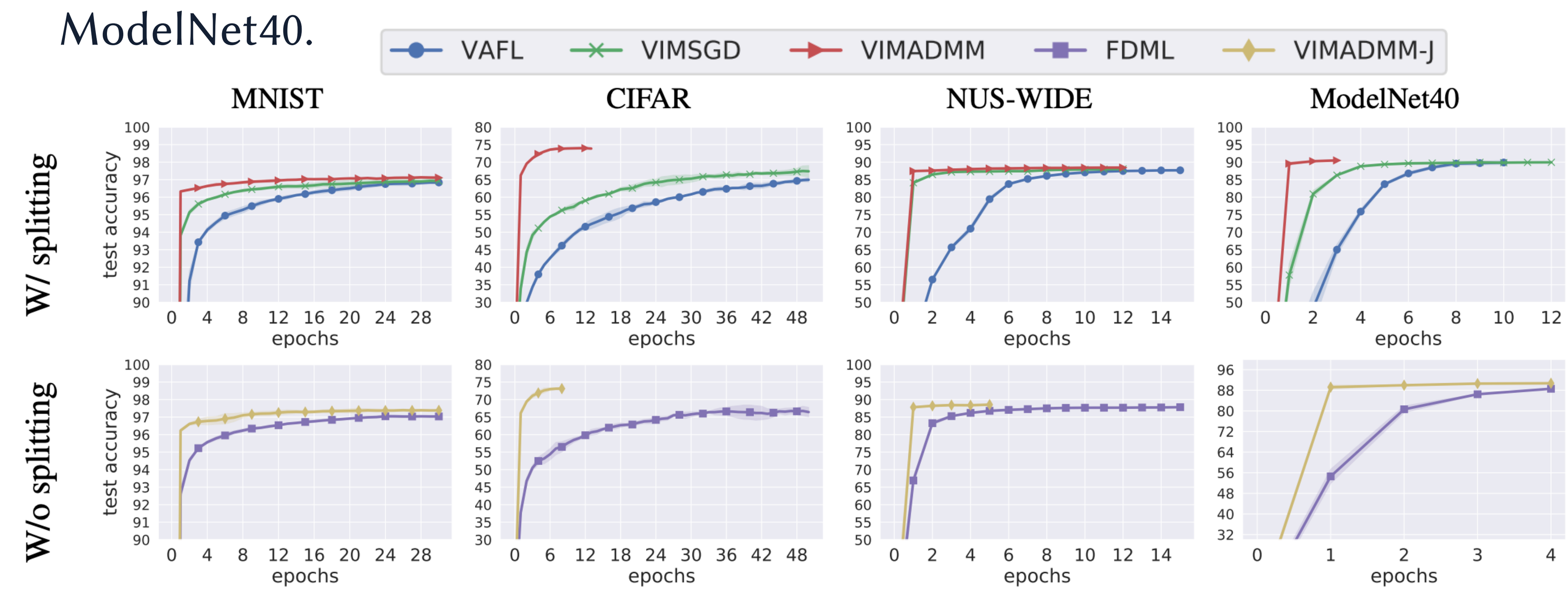
- **Comparing to** SGD-based VFL methods [1,2] where server sends gradients to clients at every training step of the local models, VIMADMM has lower communication costs.

❖ Reduce **frequency** by allowing **multiple local updating steps** at each round;
❖ Reduce **dimensionality** of information by **exchanging ADMM-related variables.**

Experiments

VIMADMM vs SOTA [1,2]

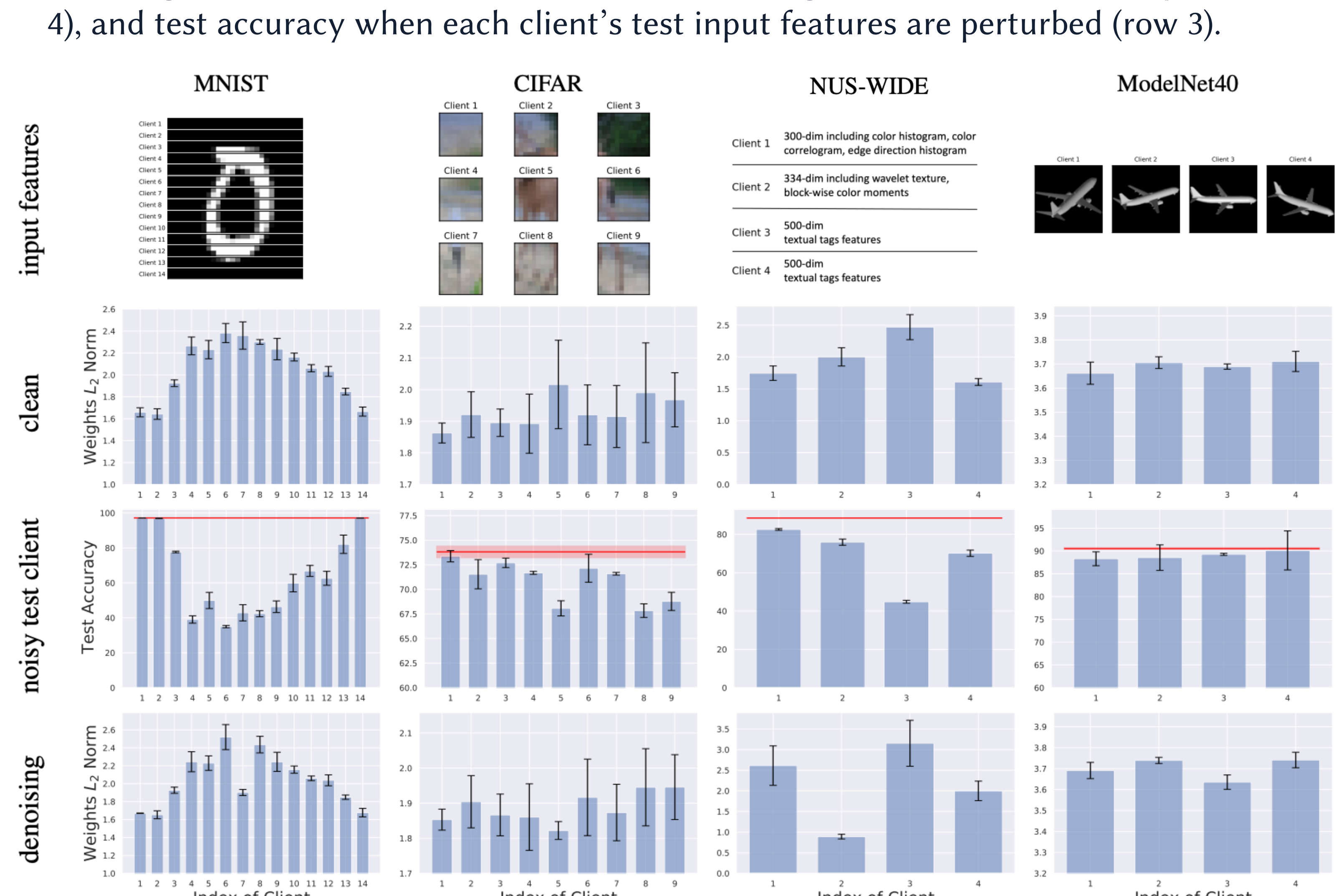
- VFL classification on four datasets: MNIST, CIFAR, NUS-WIDE and ModelNet40.



❖ Faster convergence; improved communication efficiency; higher accuracy.

Client-level Explainability of VIM

- The weights norm of linear heads under clean setting (row 2), under one noisy client (row 4), and test accuracy when each client's test input features are perturbed (row 3).



❖ The weights of linear heads reflect the importance of local clients;
❖ Perturbing the client with high weights has higher impact on test accuracy;
❖ VIM enables client denoising by lowering their weights.

More details and results are in our paper:

- Details for algorithm VIMADMM-J for VFL without model splitting setting.
- Results on communication costs comparison, effect of penalty factor and local steps, client summarization and the visualization of the local embedding.

References:

[1] Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. Vafl: a method of vertical asynchronous federated learning. arXiv, 2020.

[2] Yaochen Hu, Di Niu, Jianming Yang, and Shengping Zhou. Fdml: A collaborative machine learning framework for distributed features. KDD, 2019.