A Quasi-Newton Method Based Vertical Federated Learning Framework for Logistic Regression

Kai Yang[♠♣], Tao Fan[♠], Tianjian Chen[♠], Yuanming Shi[♣], and Qiang Yang[♦]

[♠]WeBank
[♣]ShanghaiTech University
[♦]Hong Kong University of Science and Technology

Privacy-Preserving Collaborative Machine Learning

Main concerns: data privacy and security for building a machine learning model

Training machine learning model at cloud data center: risks of data breach and violation of data protection laws and regulations (e.g., GDPR by the European Union)

Federated learning: an emerging frontier field on privacy-preserving collaborative machine learning while leaving data instances at their providers locally [1]

- **horizontal federated learning structure:** each node has a subset of data instances with complete data attributes
- **vertical federated learning structure:** each node holds a disjoint subset of attributes for all data instances

Communication challenge: one of the main bottlenecks in federated learning due to the much worse network conditions than the cloud center

Vertical Federated Learning

Vertical federated learning framework: joint computation and communication design for different ML models (e.g., logistic regression, boosting-tree, etc.)

Design target:

- preserving data privacy
- low communication costs

State-of-the-art: SGD proposed in [2] based on Taylor expansion and additive homomorphic encryption

Challenges:

- high communication costs due to low convergence rate of SGD
- high computational costs of second-order methods

Proposal: computationally efficient **quasi-Newton method** to improve the convergence rate without introducing much additional communication costs at each iteration

Quasi-Newton methods:

- **L-BFGS:** high communication costs for transmitting inverse Hessian matrix
- **stochastic quasi-newton method in [Schraudolph, et al., 2007]:** unstable estimation for the inverse Hessian matrix with small batch sizes
- **stochastic L-BFGS in [Moritz, et al., 2016]:** requiring computing the full gradient for approximating inverse Hessian matrix, which doubles the average communication cost at each iteration
- **stochastic quasi-newton method in [3]:** updating the approximated inverse Hessian matrix every L iterations based on a gradient-like vector.

We develop a communication efficient vertical federated learning framework based on the stochastic quasi-Newton method proposed in [3].

Problem Statement: Vertical Logistic Regression

Problem setting of logistic regression:

- $\boldsymbol{X} \in \mathbb{R}^{n\times T}$: data set consisting of T data samples and each instance has n features
- $\boldsymbol{y} \in \{-1,+1\}^T$: labels of $\boldsymbol{X}$
- $\boldsymbol{w}$: model parameters
- $\boldsymbol{x}_i$: i-th data instance
- y_i: label of $\boldsymbol{x}_i$
- $l(\boldsymbol{w};\boldsymbol{x}_i,y_i)=\log(1+\exp(y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i))$: negative log-likelihood loss
- **Target:**

$$\underset{\boldsymbol{w}\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{T}\sum_{i}^{T} l(\boldsymbol{w};\boldsymbol{x}_i,y_i), \tag{1}$$

System setting of vertically federated learning for logistic regression: each party holds a disjoint subset of data features over a common sample IDs

- A and B: two honest-but-curious private parties
- A: the **host** data provider with only features ($\boldsymbol{X}^A \in \mathbb{R}^{n_A\times T}$)
- B: the **guest** data provider with features ($\boldsymbol{X}^B \in \mathbb{R}^{n_B\times T}$) and labels $\boldsymbol{y}\in\{-1,+1\}^T$

Vertically partitioned model parameters: party A and party B hold the model parameters corresponding to their features respectively, i.e., $\boldsymbol{w}=(\boldsymbol{w}^A\in\mathbb{R}^{n_A},\boldsymbol{w}^B\in\mathbb{R}^{n_B})$

Additively homomorphic encryption for exchanging encrypted intermediate values: e.g., Paillier.

- Encryption: $[\![u]\!]+[\![v]\!]=[\![u+v]\!], v\cdot[\![u]\!]=[\![vu]\!]$ where $[\![\cdot]\!]$ is the encryption operation
- Decryption: requiring a third party called the **coordinator**

Taylor loss: $l(\boldsymbol{w};\boldsymbol{x}_i,y_i)\approx\log 2-\frac{1}{2}y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i+\frac{1}{8}(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i)^2$ second-order Taylor approximation for loss function.

Our Work: A Quasi-Newton Method for Vertical Logistic Regression

Quasi-Newton Method Based Vertical Federated Learning

Target: reducing communication rounds without increasing much communication bandwidth at per round with quasi-Newton method

Key ideas:

- **curvature information** $\boldsymbol{H}$: estimated inverse Hessian matrix
- Update of quasi-Newton method: $\boldsymbol{w}\leftarrow\boldsymbol{w}-\eta\boldsymbol{H}\boldsymbol{g}$
- **Subsampled method** for curvature estimation [3]: updating $\boldsymbol{H}$ every L iterations to reduce the communication overhead as well as improve the stability of quasi-Newton algorithm

Gradient of Taylor loss: $\nabla l(\boldsymbol{w};\boldsymbol{x}_i,y_i)\approx\left(\frac{1}{4}\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i-\frac{1}{2}y_i\right)\boldsymbol{x}_i$ **Hessian of Taylor loss:** $\nabla^2 l(\boldsymbol{w};\boldsymbol{x}_i,y_i)\approx\frac{1}{4}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T}$

Proposed framework:

- **Computing Loss and Gradient at Party A&B:**

At each iteration, choose a mini-batch of data instances: $\mathcal{S}\subseteq\{1,\cdots,T\}$ is the index set.

- loss and gradient $\mathrm{loss},\boldsymbol{g}$: $\mathrm{loss}=F(\boldsymbol{w})=\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}l(\boldsymbol{w};\boldsymbol{x}_i,y_i)$, $\quad\boldsymbol{g}=\nabla F(\boldsymbol{w})=\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\nabla l(\boldsymbol{w};\boldsymbol{x}_i,y_i)$
- intermediate values $\boldsymbol{u}_A,\boldsymbol{u}_A^2,\boldsymbol{u}_B,\boldsymbol{u}_B^2,\boldsymbol{d}$: $\boldsymbol{u}_A=\{\boldsymbol{u}_A[i]=\boldsymbol{w}^{A^\mathsf{T}}\boldsymbol{x}_i^A:i\in\mathcal{S}\},\boldsymbol{u}_A^2=\{\boldsymbol{u}_A^2[i]=(\boldsymbol{w}^{A^\mathsf{T}}\boldsymbol{x}_i^A)^2:i\in\mathcal{S}\}$ $\boldsymbol{u}_B=\{\boldsymbol{u}_B[i]=\boldsymbol{w}^{B^\mathsf{T}}\boldsymbol{x}_i^B:i\in\mathcal{S}\},\boldsymbol{u}_B^2=\{\boldsymbol{u}_B^2[i]=(\boldsymbol{w}^{B^\mathsf{T}}\boldsymbol{x}_i^B)^2:i\in\mathcal{S}\}$
$\boldsymbol{d}=\{d_i=\frac{1}{4}(\boldsymbol{u}_A[i]+\boldsymbol{u}_B[i]-\frac{1}{2}y_i):i\in\mathcal{S}\}$
- **encrypted loss and gradient** $[\![\mathrm{loss}]\!],[\![\boldsymbol{g}]\!]$:
$[\![\mathrm{loss}]\!]\approx\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}[\![\log 2]\!]-\frac{1}{2}y_i([\![\boldsymbol{u}_A[i]]\!]+[\![\boldsymbol{u}_B[i]]\!])+\frac{1}{8}([\![\boldsymbol{u}_A^2[i]]\!]+2\boldsymbol{u}_B[i][\![\boldsymbol{u}_A[i]]\!]+[\![\boldsymbol{u}_B^2[i]]\!])$, $[\![\boldsymbol{g}]\!]\approx\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}[\![d_i]\!]\boldsymbol{x}_i=([\![\mathbf{g}^\mathbf{A}]\!],[\![\mathbf{g}^\mathbf{B}]\!])=(\sum_{i\in\mathcal{S}}[\![d_i]\!]\boldsymbol{x}_i^A,\sum_{i\in\mathcal{S}}[\![d_i]\!]\boldsymbol{x}_i^B)$, $[\![d_i]\!]=\frac{1}{4}([\![\boldsymbol{u}_A[i]]\!]+[\![\boldsymbol{u}_B[i]]\!]+[\![-\frac{1}{2}y_i]\!])$.

At each iteration, by transmitting $[\![\boldsymbol{u}_A]\!]$ from party A to party B, and transmitting $[\![\boldsymbol{d}]\!]$ from B to A, $[\![\boldsymbol{g}^A]\!]$ can be computed at party A, $[\![\mathrm{loss}]\!]$ and $[\![\boldsymbol{g}^B]\!]$ can be computed at party B privately.

- **Computing Updates for Estimating Curvature Information at Party A&B:**

Every L iterations, choose a subset of data instances for estimating curvature information: $\mathcal{S}_H$. The coordinator collects encrypted $\boldsymbol{v}=(\boldsymbol{v}^A,\boldsymbol{v}^B)\in\mathbb{R}^n$ from party A and B for updating the curvature information $\boldsymbol{H}$.

- difference of average model parameters $\boldsymbol{s}_t$: $\boldsymbol{s}_t=\bar{\boldsymbol{w}}_t-\bar{\boldsymbol{w}}_{t-1}=(\mathbf{s}_\mathbf{t}^\mathbf{A},\mathbf{s}_\mathbf{t}^\mathbf{B})$, $\bar{\boldsymbol{w}}_t=\sum_{i=k-L+1}^{k}\boldsymbol{w}_i/L$, $\bar{\boldsymbol{w}}_{t-1}=\sum_{i=k-2L+1}^{k-L}\boldsymbol{w}_i/L$
- $\boldsymbol{v},\boldsymbol{h}$: $\boldsymbol{v}_t=\nabla^2\hat{F}(\bar{\boldsymbol{w}}_t)\boldsymbol{s}_t$, where $\nabla^2\hat{F}(\bar{\boldsymbol{w}}_t)=\frac{1}{|\mathcal{S}_H|}\sum_{i\in\mathcal{S}_H}\nabla^2 l(\bar{\boldsymbol{w}}_t;\boldsymbol{x}_i,y_i)=\frac{1}{|\mathcal{S}_H|}\sum_{i\in\mathcal{S}_H}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T}$. $\boldsymbol{h}=\{h_i=\mathbf{\Delta\bar{u}_i^A}+\mathbf{\Delta\bar{u}_i^B}=\boldsymbol{s}_t^{A^\mathsf{T}}\boldsymbol{x}_i^A+\boldsymbol{s}_t^{B^\mathsf{T}}\boldsymbol{x}_i^B,i\in\mathcal{S}_H\}$.
- Computing $[\![\boldsymbol{v}]\!]$: $[\![\boldsymbol{v}_t]\!]=\frac{1}{|\mathcal{S}_H|}\sum_{i\in\mathcal{S}_H}[\![h_i]\!]\boldsymbol{x}_i=([\![\boldsymbol{v}_t^A]\!],[\![\boldsymbol{v}_t^B]\!])=\left(\frac{1}{|\mathcal{S}_H|}\sum_{i\in\mathcal{S}_H}[\![h_i]\!]\boldsymbol{x}_i^A,\frac{1}{|\mathcal{S}_H|}\sum_{i\in\mathcal{S}_H}[\![h_i]\!]\boldsymbol{x}_i^B\right)$,

Every L iterations, by transmitting $[\![\Delta\bar{\boldsymbol{u}}_A]\!]=\{[\![\Delta\bar{u}_i^A]\!]:i\in\mathcal{S}_H\}$ from party A to party B, and transmitting $[\![\boldsymbol{h}]\!]=\{[\![h_i]\!]:i\in\mathcal{S}_H\}$ from B to A, $[\![\boldsymbol{v}_t^A]\!]$ can be computed at party A and $[\![\boldsymbol{v}_t^B]\!]$ can be computed at party B privately.

- **Computing Descent Direction at the Coordinator:** By decryption the coordinator obtains $\mathrm{loss},\boldsymbol{g},\boldsymbol{v}$ from party A and B.

At each iteration, the coordinator should determine a descent direction $\tilde{\boldsymbol{g}}$ for updating $\boldsymbol{w}^A$ and $\boldsymbol{w}^B$: $\boldsymbol{w}\leftarrow\boldsymbol{w}-\tilde{\boldsymbol{g}}=\boldsymbol{w}-\eta\boldsymbol{H}\boldsymbol{g}=(\boldsymbol{w}^A-\tilde{\boldsymbol{g}}^A,\boldsymbol{w}^B-\tilde{\boldsymbol{g}}^B)$.

Every L iterations, the coordinator should also update $\boldsymbol{H}$ based on the collected encrypted loss $[\![\mathrm{loss}]\!]$, gradient $[\![\boldsymbol{g}]\!]$, and $[\![\boldsymbol{v}]\!]$ from party A&B.

- Initial point: $\boldsymbol{H}=(\boldsymbol{v}_t^\mathsf{T}\boldsymbol{s}_t/\boldsymbol{v}_t^\mathsf{T}\boldsymbol{v}_t)\boldsymbol{I}$. For $\forall j=t-M+1,\cdots,t$, iteratively compute $\boldsymbol{H}\leftarrow(\boldsymbol{I}-\rho_j\boldsymbol{s}_j\boldsymbol{v}_j^\mathsf{T})\boldsymbol{H}(\boldsymbol{I}-\rho_j\boldsymbol{v}_j\boldsymbol{s}_j^\mathsf{T})+\rho_j\boldsymbol{s}_j\boldsymbol{s}_j^\mathsf{T}$, $\rho_j=1/(\boldsymbol{v}_j^\mathsf{T}\boldsymbol{s}_j)$,

The source code will be released in an upcoming version of the FATE framework [4].

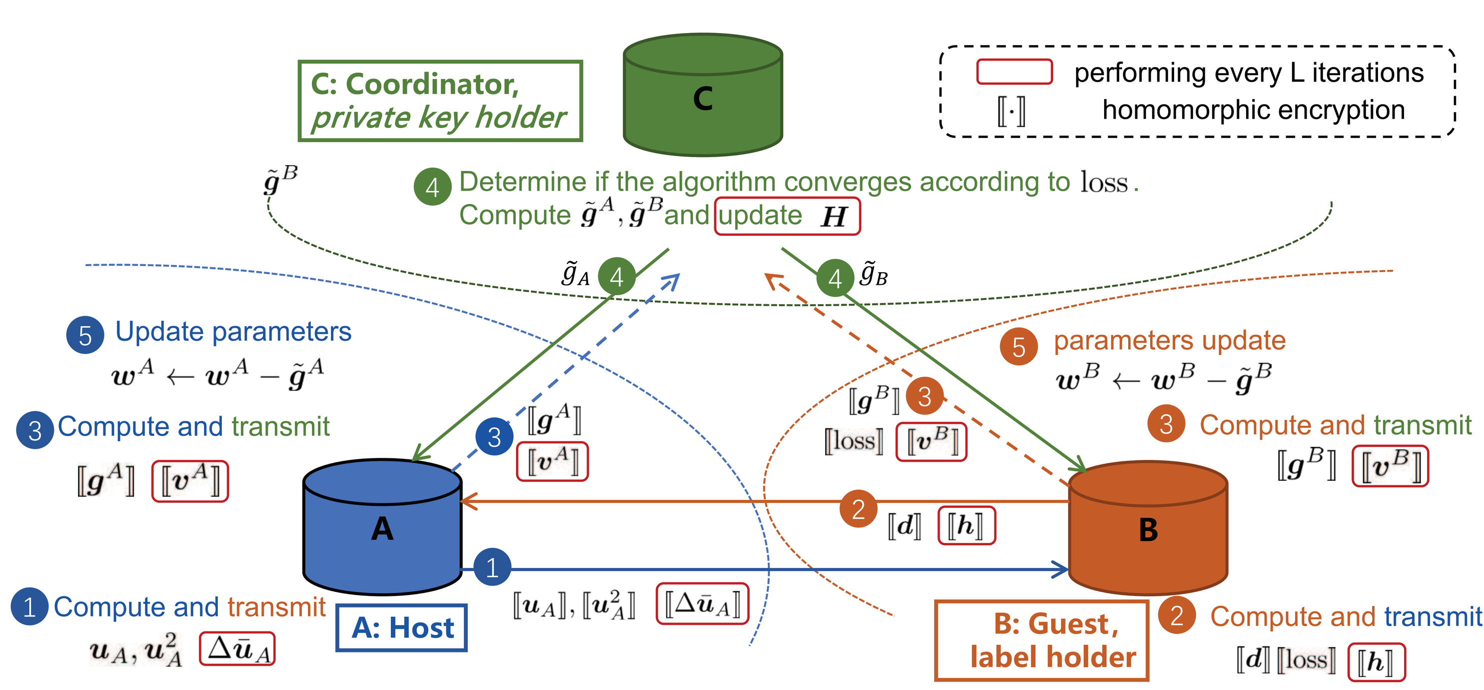

Figure 1: Proposed Framework for Vertical Federated Learning

Algorithm 1: Proposed Framework for Vertical Federated Learning

Input : $\boldsymbol{w}_0^A,\boldsymbol{w}_0^B,M,L$
Output: $\boldsymbol{w}^A,\boldsymbol{w}^B$

Set $t=0,\boldsymbol{H}=\boldsymbol{I}$
for *each round* $k=1,\cdots,$ **do**
  Choose a minibatch $\mathcal{S}$
  if *mod*$(k,L)\neq 0$ **then**
    Party A&B: compute $[\![\mathrm{loss}]\!],[\![\boldsymbol{g}]\!]$ as equation (3) (4)
    Coordinator: $\boldsymbol{w}_{k+1}=\boldsymbol{w}_k-\tilde{\boldsymbol{g}}_k$ where $\tilde{\boldsymbol{g}}_k=\eta\boldsymbol{H}\boldsymbol{g}$
  else
    $t\leftarrow t+1$
    Party A&B: Choose a minibatch $\mathcal{S}_H$
    compute $[\![\mathrm{loss}]\!],[\![\boldsymbol{g}]\!],[\![\boldsymbol{v}_t]\!]$ as equation (3) (4) (6)
    Coordinator: $\boldsymbol{w}_{k+1}=\boldsymbol{w}_k-\tilde{\boldsymbol{g}}_k$ where $\tilde{\boldsymbol{g}}_k=\eta\boldsymbol{H}\boldsymbol{g}$
    $\boldsymbol{s}_t=\sum_{i=k-L+1}^{k}\tilde{\boldsymbol{g}}_i/L-\sum_{i=k-2L+1}^{k-L}\tilde{\boldsymbol{g}}_i/L$
    if $t>1$ **then**
      $\boldsymbol{H}\leftarrow(\boldsymbol{s}_t^T\boldsymbol{v}_t)/(\boldsymbol{v}_t^T\boldsymbol{v}_t)\boldsymbol{I},\tilde{m}=\min\{M,t\}$
      for $j=t-\tilde{m}+1,\cdots,t$ **do**
        $\rho_j=1/(\boldsymbol{v}_j^T s_j)$
        $\boldsymbol{H}\leftarrow(\boldsymbol{I}-\rho_j\boldsymbol{s}_j\boldsymbol{v}_j^T)\boldsymbol{H}(\boldsymbol{I}-\rho_j\boldsymbol{v}_j\boldsymbol{s}_j^T)+\rho_j\boldsymbol{s}_j\boldsymbol{s}_j^T$
      end
    end
    $\tilde{w}_t=0$
  end
end

Communication Costs of Each Iteration

Communication costs of SGD [2]: $3|\mathcal{S}|$ encrypted numbers between party A and party B, and $2n$ encrypted numbers between party A&B and the coordinator.

Communication costs of proposed framework: $3|\mathcal{S}|+2|\mathcal{S}_H|/L$ encrypted numbers between party A and party B, and $(2+1/L)n$ encrypted numbers between party A&B and the coordinator. By choosing $|\mathcal{S}_H|\le|\mathcal{S}|$, the presented quasi-Newton method introduces no more than $1/L$ additional communication costs at per communication round compared with [2].

Experiments

Numerical experiments on two credit scoring data sets by setting $\mathcal{S}_H=\mathcal{S}$ and $L=4$:

- **Credit** 1: 30000 data instances and $n=25$ attributes.
- **Credit** 2: 150000 data instances and 10 attributes.

Table 1: Numerical Results on Two Public Data Sets

Batch Size	Method	Credit 1			Credit 2		
		Epochs	Loss	AUC	Epochs	Loss	AUC
1000	SGD	12	0.496218	0.7224	12	0.314555	0.7033
	Proposed	3	0.496600	0.7222	4	0.314643	0.7061
3000	SGD	18	0.496194	0.7219	14	0.314648	0.6982
	Proposed	12	0.496317	0.7225	6	0.314490	0.7077

Conclusions

- Addressing the communication challenge in vertical federated learning for logistic regression.
- A quasi-Newton framework to reduce the number of communication rounds without introducing much additional communication costs at each round.
- Computing an encrypted gradient and an additional vector every L iterations for updating the curvature information with additively homomorphic encryption.
- Advantages demonstrated via numerical experiments.

References

[1] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):12, 2019.

[2] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *arXiv preprint arXiv:1711.10677*, 2017.

[3] Richard H Byrd, Samantha L Hansen, Jorge Nocedal, and Yoram Singer, "A stochastic quasi-newton method for large-scale optimization," *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

[4] WeBank. FATE: An industrial grade federated learning framework. `https://fate.fedai.org`, 2018.

Contact Information

- Kai Yang: yangkai@shanghaitech.edu.cn
- Tao Fan: dylanfan@webank.com