42nd IEEE Symposium on Security and Privacy 2021 Poster Session

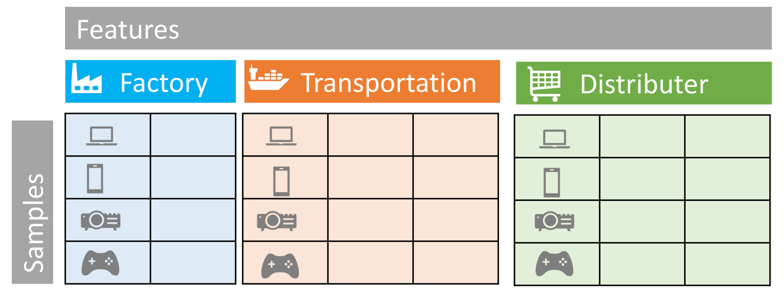
FedV: Privacy-Preserving Federated Learning over Vertically Partitioned Data

Runhua Xu, Nathalie Baracaldo, Yi Zhou, Ali Anwar and Heiko Ludwig, *IBM Research, San Jose, CA, USA*

INTRODUCTION

Federated Learning: Collaborative train a machine learning model without sharing/revealing training data introduced in [2]

An example of vertically distributed data



Vertical FL

Parties have different features
Only one party has label → Together they form the complete feature set
Data is private → Privacy or regulatory constraints

Overview of FedV [1]

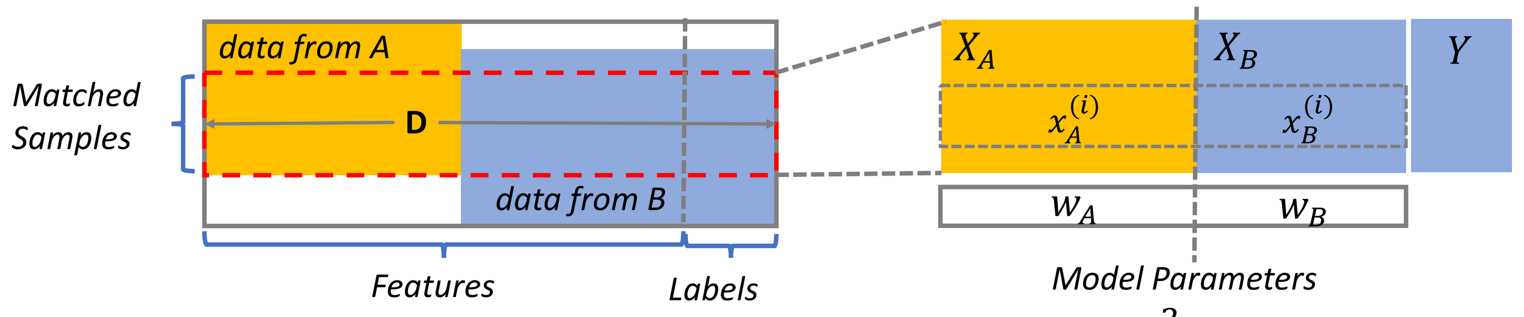
Existing HE-based Solution [5]	FedV Solution
Require peer-to-peer communication ✗	✓ No peer-to-peer communication
Scalability issue: design only for two parties ✗	✓ Scalable for more than two parties
Require Taylor approximation ✗	✓ No Taylor approximation

Experimental Evaluation: FedV reduces in average
- 10-70% in training and
- 80-90% in data transfer

BACKGROUND

Gradient Descent in Vertical FL

Suppose participants A, B (B owns labels) in VFL - target loss function is mean squared loss



Challenges :
1. Compute gradient descent without sharing $x_p^{(i)} w_p$
2. Compute $\left(y^{(i)} - f(x^{(i)}; w)\right)[x_p^{(i)}]$ without sharing $x_p^{(i)}$

Mean Squared Loss → $E_D(\boldsymbol{w}) = 1/n \sum_{i=1}^{n} \left(y^{(i)} - f(\boldsymbol{x}^{(i)}; \boldsymbol{w})\right)^2$

Gradient Descent → $E_D(\boldsymbol{w}) = 1/n \sum_{i=1}^{n} L(y^{(i)}, f(\boldsymbol{x}^{(i)}; \boldsymbol{w})) + \lambda R(\boldsymbol{w}); \boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla E_D(\boldsymbol{w})$

$\rightarrow \nabla E_D(\boldsymbol{w}) = -2/n \sum_{i=1}^{n} \left(y^{(i)} - f(\boldsymbol{x}^{(i)}; \boldsymbol{w})\right) \left[\boldsymbol{x}_A^{(i)}; \boldsymbol{x}_B^{(i)}\right]$

$\rightarrow \nabla E_D(\boldsymbol{w}) = -2/n \sum_{i=1}^{n} \left(y^{(i)} - x_A^{(i)} w_A - x_B^{(i)} w_B\right) x_A^{(i)}; \left(y^{(i)} - x_A^{(i)} w_A - x_B^{(i)} w_B\right) x_B^{(i)}$

Inner-Product Functional Encryption Schemes

Allows a decryptor to compute $<\boldsymbol{x}, \boldsymbol{y}> = \sum x_i y_i$ over ciphertext $C = E_{sk}(\boldsymbol{x})$ of $\boldsymbol{x}$ without learning $\boldsymbol{x}$

NOTE: $\boldsymbol{x} = (x_1, \ldots, x_n)$ is a vector, how is $\boldsymbol{x}$ composed?

Single-Input FE[3]

All elements in $\boldsymbol{x}$ are from one source P, i.e., all x_i are from source P
- P has a public key pk_P
- P encrypts the entire vector $\boldsymbol{x}$

$C = E_{pk_P}(\boldsymbol{x})$

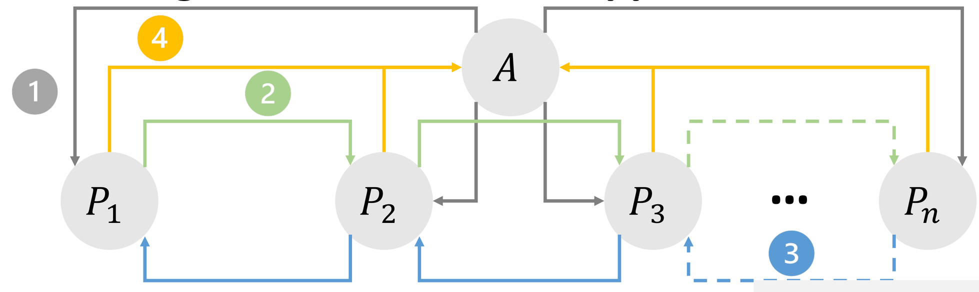
Multi-Input FE[4]

Elements in $\boldsymbol{x}$ are from multiple sources $P_1, \ldots, P_n$, i.e., x_i is just from source P_i

$C = \{E_{sk_{P_1}}(x_1), \ldots, \{E_{sk_{P_n}}(x_n)\}$
- Each source has its own secret key, i.e., P_i has a secret key sk_{P_i}
- Each source encrypts its data, i.e., P_i encrypts the element x_i, $\{E_{sk_{P_i}}(x_i)\}$

Comparison of Communication Topology – FedV and SOTA

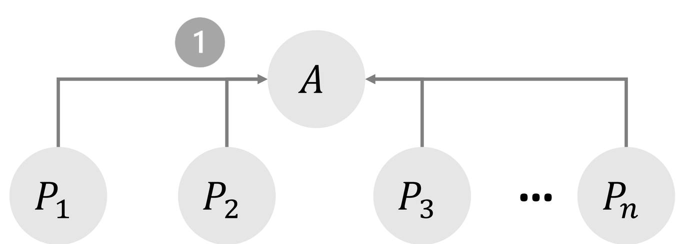
Existing Work: Additive HE Approach for SGD[5]



Suppose P_1 is active party (with label)

FOUR STEPS
1. Encrypted mask $\langle m \rangle$
2. $\langle u_1 \rangle \leftarrow \langle m \rangle \circ \langle w_1 x_1 - y \rangle \ldots \langle u_i \rangle \leftarrow \langle u_{i-1} \rangle + \langle m \rangle \circ \langle w_i x_i \rangle$
3. $\langle f(x; w) - y \rangle \leftarrow \langle u_n \rangle$
4. $\langle \nabla E(w)_1 \rangle \leftarrow \langle f(x; w) - y \rangle x_1 \ldots \langle \nabla E(w)_i \rangle \leftarrow \langle f(x; w) - y \rangle x_i$

FedV-SecGrad



ONLY ONE STEP
1. Feature-dim encrypted update
   - Active party: $ct_{fd,1} \leftarrow \langle w_1 x_1 - y \rangle$
   - Passive party: $ct_{fd,i} \leftarrow \langle w_i x_i \rangle$

   Sample-dim encrypted update $ct_{sd,i} \leftarrow \langle (f(x; w) - y) x_i \rangle$

Highlight
- No peer-to-peer communication
- One-way communication (per training round)
- Efficient computation compared to multiplication computation in additive HE
  - example: $u\langle m \rangle \rightarrow \langle m \rangle_1 + \cdots \langle m \rangle_u$

Demonstration – FedV-SecGrad

Active Party | ***Passive Parties*** | ***Aggregator***

1 Private Entity Resolution

Active party: $\begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n_{p_1})} \end{bmatrix} \begin{bmatrix} x_1^{(1)} & \cdots & x_m^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n_{p_1})} & \cdots & x_m^{(n_{p_1})} \end{bmatrix} \Rightarrow \begin{bmatrix} CLK^{(1)} \\ \vdots \\ CLK^{(n_{p_1})} \end{bmatrix}$

Passive parties: $\begin{bmatrix} x_{m+1}^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_{m+1}^{(n_{p_2})} & \cdots & x_d^{(n_{p_2})} \end{bmatrix} \Rightarrow \begin{bmatrix} CLK^{(1)} \\ \vdots \\ CLK^{(n_{p_2})} \end{bmatrix}$

Aggregator: *Matched Set* $\begin{bmatrix} CLK^{(1)} \\ \vdots \\ CLK^{(n)} \end{bmatrix}$ $\begin{bmatrix} \boldsymbol{\pi}_{p_1} \\ \vdots \\ \boldsymbol{\pi}_{p_2} \end{bmatrix}$ *Permutation*

$\boldsymbol{\pi}_{p_1}$: $\begin{bmatrix} y^{(1')} \\ \vdots \\ y^{(n'_{p_1})} \end{bmatrix} \begin{bmatrix} x_1^{(1')} & \cdots & x_m^{(1')} \\ \vdots & \ddots & \vdots \\ x_1^{(n'_{p_1})} & \cdots & x_m^{(n'_{p_1})} \end{bmatrix}$

$\boldsymbol{\pi}_{p_2}$: $\begin{bmatrix} x_{m+1}^{(1')} & \cdots & x_d^{(1')} \\ \vdots & \ddots & \vdots \\ x_{m+1}^{(n'_{p_2})} & \cdots & x_d^{(n'_{p_2})} \end{bmatrix}$

2 Private Batch Selection

one-time-chain (r)

$\begin{bmatrix} y^{(i+1)} \\ \vdots \\ y^{(i+s)} \end{bmatrix} \begin{bmatrix} x_1^{(i+1)} & \cdots & x_m^{(i+1)} \\ \vdots & \ddots & \vdots \\ x_1^{(i+s)} & \cdots & x_m^{(i+s)} \end{bmatrix} D_{p_1}^{s \times m}$ *e.g., i-th batch samples*

$\begin{bmatrix} x_{m+1}^{(i+1)} & \cdots & x_d^{(i+1)} \\ \vdots & \ddots & \vdots \\ x_{m+1}^{(i+s)} & \cdots & x_d^{(i+s)} \end{bmatrix} D_{p_2}^{s \times (d-m)}$

Demonstration - Logistic Regression (non-linear model)

$$\nabla E(\boldsymbol{w}) = 1/s \sum_{i}^{s} \left(\frac{1}{1 + e^{-\boldsymbol{w}^T x^{(i)}}} - y^{(i)} \right) \boldsymbol{x}^{(i)}$$

- PHASE 1: Feature-Dim SA: $\boldsymbol{w}^T \boldsymbol{x} \rightarrow \boldsymbol{u}$
- Normal Computation: $\frac{1}{1+e^{-u}} - y^{(i)} \rightarrow \boldsymbol{u}'$
- PHASE 2: Sample-Dim SA: $\boldsymbol{u}'\boldsymbol{x} \rightarrow \boldsymbol{\nabla E}(\boldsymbol{w})$

3 FedV-SecGrad (2Phased-SA)

model parameters of last round $[w_1 \ \cdots \ w_m, w_{m+1} \ \cdots \ w_d]$

$[w_1 \ \cdots \ w_m]$ | $[w_{m+1} \ \cdots \ w_d]$

PHASE 1

$\begin{bmatrix} E_{sk_1}^{MIFE}\left(w_1 x_1^{(i+1)} + \cdots + w_m x_m^{(i+1)} - y^{(i+1)}\right) \\ \vdots \\ E_{sk_1}^{MIFE}\left(w_1 x_1^{(i+s)} + \cdots + w_m x_m^{(i+s)} - y^{(i+s)}\right) \end{bmatrix}$ $\begin{bmatrix} E_{sk_2}^{MIFE}\left(w_{m+1} x_{m+1}^{(i+1)} + \cdots + w_b x_b^{(i+1)}\right) \\ \vdots \\ E_{sk_2}^{MIFE}\left(w_{m+1} x_{m+1}^{(i+s)} + \cdots + w_b x_b^{(i+s)}\right) \end{bmatrix}$

dk_{v_p} - decryption key based on v_p (fusion weight vector)

$\begin{bmatrix} D_{\mathrm{dk}_{v_p}}^{MIFE}\left(\{c_{\mathrm{fd},p_1}^{(i+1)}, c_{\mathrm{fd},p_2}^{(i+1)}\}\right) \\ \vdots \\ D_{\mathrm{dk}_{v_p}}^{MIFE}\left(\{c_{\mathrm{fd},p_1}^{(i+s)}, c_{\mathrm{fd},p_2}^{(i+s)}\}\right) \end{bmatrix} \Rightarrow \begin{bmatrix} u^{(i+1)} \\ \vdots \\ u^{(i+b)} \end{bmatrix}$

PHASE 2

$\begin{bmatrix} E_{pk}^{SIFE}\left(x_1^{(i+1)}\right) & \cdots & E_{pk}^{SIFE}\left(x_m^{(i+1)}\right) \\ \vdots & \ddots & \vdots \\ E_{pk}^{SIFE}\left(x_1^{(i+s)}\right) & \cdots & E_{pk}^{SIFE}\left(x_m^{(i+s)}\right) \end{bmatrix}$ $\begin{bmatrix} E_{pk}^{SIFE}\left(x_{m+1}^{(i+1)}\right) & \cdots & E_{pk}^{SIFE}\left(x_d^{(i+1)}\right) \\ \vdots & \ddots & \vdots \\ E_{pk}^{SIFE}\left(x_{m+1}^{(i+s)}\right) & \cdots & E_{pk}^{SIFE}\left(x_d^{(i+s)}\right) \end{bmatrix}$

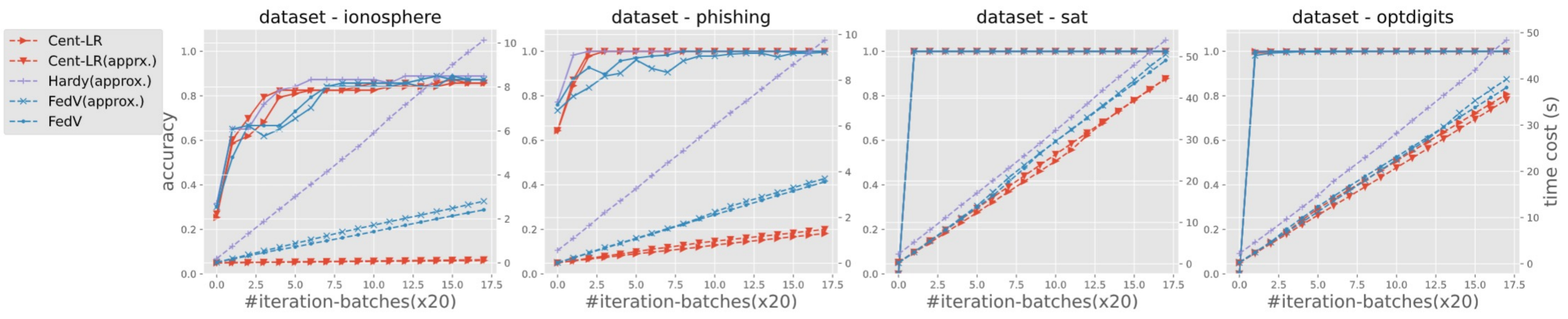
dk_u - decryption key based on $\boldsymbol{u}$

$\begin{bmatrix} D_{\mathrm{dk}_u}^{SIFE}\left(\{c_{\mathrm{sd},1}^{(i+1)}, \ldots, c_{\mathrm{sd},1}^{(i+s)}\}\right) \\ \vdots \\ D_{\mathrm{dk}_u}^{SIFE}\left(\{c_{\mathrm{sd},d}^{(i+1)}, \ldots, c_{\mathrm{sd},d}^{(i+s)}\}\right) \end{bmatrix} \Rightarrow \nabla E(\boldsymbol{w})$

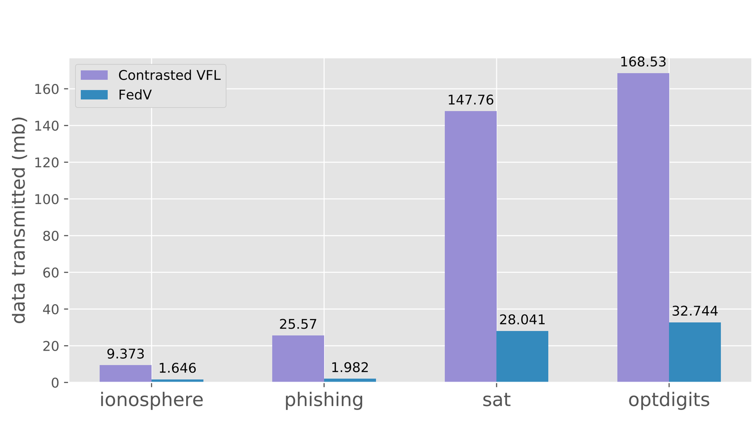
4 Model Update $\nabla E(\boldsymbol{w})$ can be used for any gradient-based step to update the ML model

EXPERIMENTAL EVALUATION

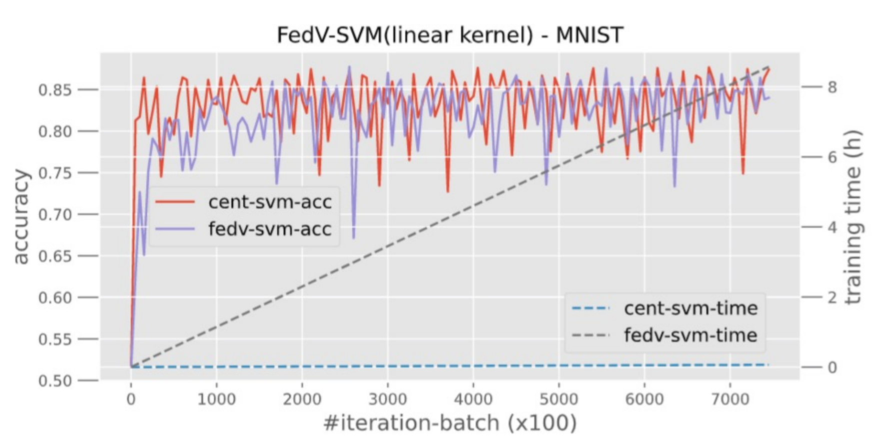
Comparable model accuracy, training time reduced by 10% to 70%



Data transfer reduced by 80% to 90%



Work well on large size of dataset



[1] Xu, Runhua, et al. "FedV: Privacy-Preserving Federated Learning over Vertically Partitioned Data." *arXiv preprint arXiv:2103.03918* (2021).
[2] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." Artificial Intelligence and Statistics. PMLR, 2017.
[3] M. Abdalla, et al., "Simple functional encryption schemes for inner products," in PKC 15.
[4] M. Abdalla, et al., "Multi-input functional encryption for inner products: function-hiding realizations and constructions without pairings," in CRYPTO 18.
[5] Hardy, Stephen, et al. "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption." arXiv preprint arXiv:1711.10677 (2017).