VERTICAL AND HORIZONTAL FL

The tough case

Diego Klabjan

Northwestern | McCORMICK SCHOOL OF ENGINEERING

Framework

- Different features implies different model architectures at clients
- Ideas from fedAvg
  - Get common architecture and average
- Common architecture
  - Embed locally to the same dimensional space
  - Is sharing individual embeddings privacy preserving?
- Unclear if independent embeddings are going to work
  - Need to impose structure
- No notion of a global model
  - Inference applicable only locally

Algorithmic Concepts

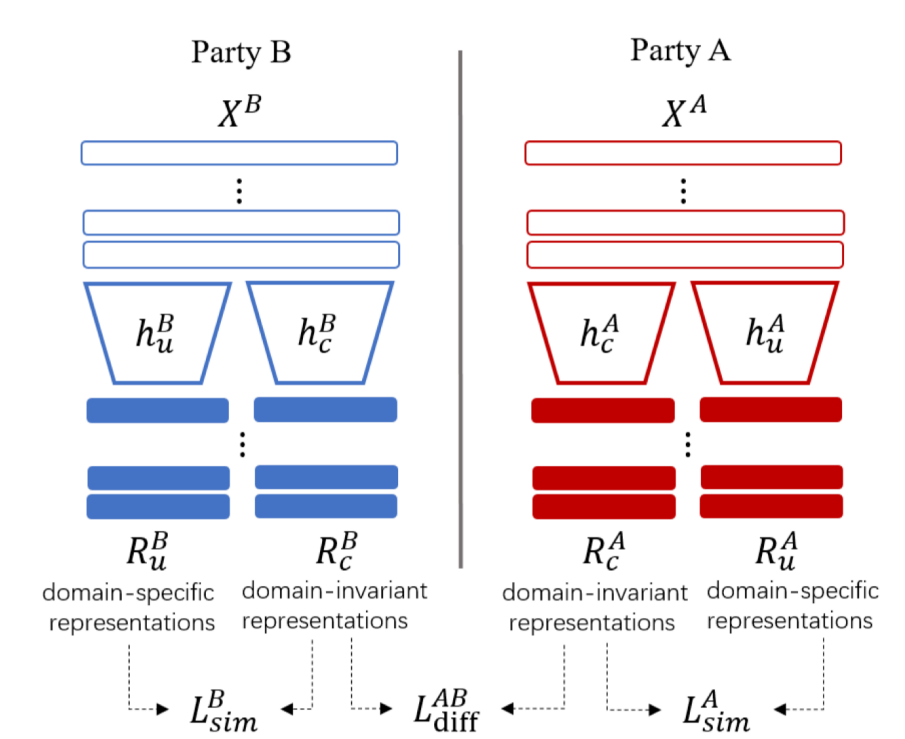
- FedAvg and siblings
  - Communicate models between iterations
- Server coordinates individual sample training
  - Server combines gradients
  - Sample IDs must be matched
  - Is this privacy preserving?
    - From many gradients, weights and architecture samples can be constructed
    - System of linear equations
- Embedding synchronization possible only for latter

VERTICAL FL

fedCVT [Kang at al, 2022]

- CVT = cross-view training
- Horizontal setting = features are partitioned, samples are shared
- Assumes two clients
  - Unclear how to use it with multiple clients
    - Scalable way
- Follow the framework
- Synchronize embeddings between two clients
  - Sample based algorithmic framework

Architecture



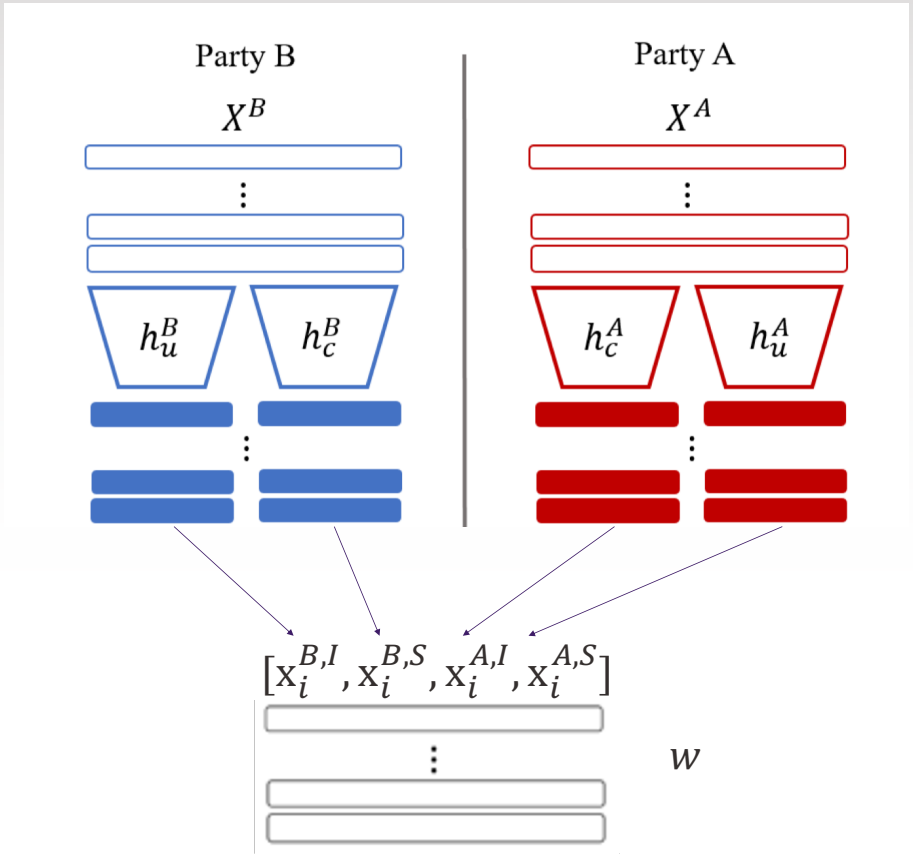
- Each client local embeddings
- Two heads
  - Two embedding types
- Images and text
  - Domain invariant = pertaining to patient
  - Domain specific
    - Feature for images at client A
    - Feature for text at client B

Synchronization

- Domain invariant
  - Should be close to each other
    $$\frac{1}{N}\sum_{i}\left\|\mathrm{x}_i^{A,I}-x_i^{B,I}\right\|_F^2$$
- Domain specific
  - Should be far apart from domain invariant
    $$\frac{1}{N}\sum_{i}\left\|\mathrm{x}_i^{A,S}\cdot(x_i^{B,S})^{\mathrm{T}}\right\|_F^2$$
  - Measured by orthogonality

- Overall loss function
  - The two terms + standard loss
  - Server knows the ground truth of every sample
- Train by using standard single sample SGD
  - Server computes gradients of loss
  - Clients backpropagate on local networks
    - Local weights are updated

- General architecture

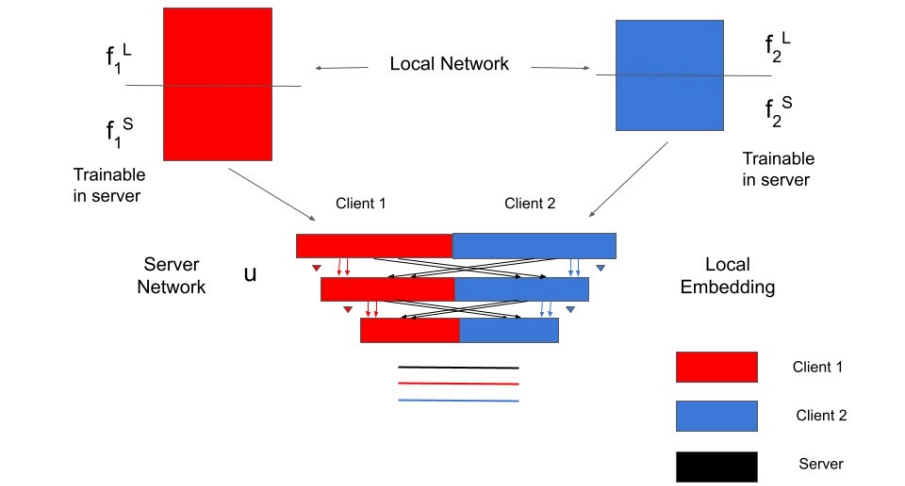


Training

- Loop
  - Server selects sample index i and sends to clients
  - Each client k computes and sends to the server $\mathrm{x}_i^{k,I}, \mathrm{x}_i^{k,S}$
  - Server computes forward step based on sample i
  - Server computes gradients and with respect to w at the bottom of the shared network
    - Uses backpropagation starting with the global loss function
    - Server sends these to each client
  - Server computes $w = w - \mu \nabla_w l$
  - Each client applies backpropagation and updates w^k

HYBRID FEDERATED LEARNING

FedEmb [Yang, Klabjan, et al 2023]



- Improved version
  - Server architecture based on graphs (GCN)
- Server trains based on embeddings received from clients and logits
  - Logits = ground truth for server
- Top layer weights exchanged with the server

FedEmb

Algorithm 1 FebEmd Training

for $j = 1, 2, \ldots, J$ **do**
  for $m = 1, 2, ..., M$ **do local training** in parallel:
    Train local network g_m using local dataset A_m.
    $l_m^L = \{f_m^L(\mathbf{x}_{F_m}; \tau_i^L) | \mathbf{x}_{F_m} \in A_m\}$ and $\theta_m^L = \{g_m(\mathbf{x}_{F_m})) | \mathbf{x}_{F_m} \in A_m\}$ with privacy protections.
    Pass l_m^L ,θ_m^S and f_m^S to the server.
  end for
  for $m = 1, 2, \ldots, M$ **do server tuning**:
    Formulate θ^S, l^L as cohesive set of vectors.
    Train network u using set of vectors θ^S, l^L.
    Let the updated weights be notated by NEW: $\alpha_j^S = \alpha^{S,NEW}$.
    Distribute back $h(\mathbf{x}_{F_m}^S)^{NEW}$ to corresponding client m.
  end for
end for

Results

Scheme & Privacy \ Datasets		MNIST		FASHION		KDD	
		Acc.	Clu.	Acc.	Clu.	Acc.	Clu.
Centralized	N/A	0.993	60k	0.895	60k	0.925	300k
FATE	Homo.	0.941	60k	0.795	60k	0.923	300k
FedEmb	Ran.	0.961	15k	0.828	15k	0.923	15k
	KM.	0.950	15k	0.812	15k	0.923	15k
	Ran.	0.954	1.2k	0.825	1.2k	0.924	1.2k
	KM.	0.951	1.2k	0.812	1.2k	0.924	1.2k
	Ran.	0.947	0.6k	0.820	0.6k	0.924	0.6k
	KM.	0.950	0.6k	0.813	0.6k	0.923	0.6k

FATE = commercial and open source implementations of fedCVT

COPING WITH PRIVACY FOR GRADIENTS

Homomorphic Encryption (HE)

- Encryption that preserves certain operations on the encrypted items
  - Server can still perform aggregation operations on variables without having access to unencrypted variables
- An approach that allows processing and computing directly on encrypted data
  - No need to decrypt data to perform basic operations
- Types
  - Additive homomorphic encryption: Enc(A+B) = Enc(A) + Enc(B)
  - Multiplicatives: Enc(A*B) = Enc(A)*Enc(B)
- Subject of active research and algorithms are becoming faster
  - HE is still relatively slow and is the primary time bottleneck in FL when used

Types

- Partially homomorphic encryption
  - Partially = single function in encrypted data possible
  - Summation or multiplication unlimited number of times
- Somehow homomorphic encryption
  - Summation and multiplications handled
  - Limited number of operations on encrypted data
- Fully homomorphic encryption
  - Summation and multiplications handled
  - Unlimited number of operations on encrypted data

Cryptography Ideas

- Problem is NP-hard if there is a hard instance
  - TSP, knapsack, maximum clique, etc
- Some problems are hard in expectation
  - An average instance is hard to solve
    - Problems on lattices
- Hacking
  - NP-hard problem often easy to solve
    - Hacker can decipher
  - Hard in expectation
    - No much a hacker can do

- Asymmetric cryptography
  - Entity generates public and private key
  - Anyone can have public key pu
  - Private key pr secretive
  - Algorithm
    - Public key owner sends message m as u = Enc(m, pu)
    - Receiver m = Dec(u, pr)
  - Attacker gets u and pu, without knowing knowing pr
    - Very hard to compute m

Lattice

- $L = \{v : v = \sum_k s_k a_k , s_k \in Z\}$
  - a are basis vectors
  - Complex since w integer
- Shortest lattice problem
  - $\min_{v \in L \setminus \{0\}} ||v||$
- This problem is NP-hard
- For an average basis is NP-hard

- $L = \{v : v = As, s_k \in Z\}$
- Consider $mod\ p$ for p being prime
  - Entries in $0 \leq \mathrm{A} \leq\ p - 1$
- Remains hard even under quantum computing

Attacking

- An attacker
  - Must find s from intercept message
  - Very hard problem on average
  - Similar to SLV
- Very hard ‘puzzle’ to solve
  - Think blockchain and mining

Error

- Recover message + error
- Standard scaling trick
  - Multiply by a factor to get integer
    - Removes the error
- Additive homomorphism
  - $(A_1, A_1 s + e_1 + m_1) + (A_2, A_2 s + e_2 + m_2) = (A_1 + A_2, (A_1 + A_2)s + (e_1 + e_2) + (m_1 + m_2)$
  - Error grows
  - Must know the maximum number of summations

FedCVT

- Use homomorphic encryption on backpropagated activations
- Server must perform only addition and multiplications
- Loss function must be approximated by polynomials
  - Taylor expansion and truncation
- Slows down training