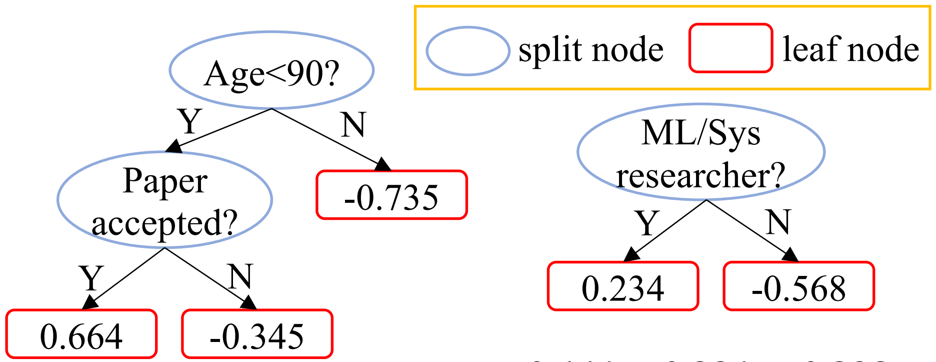
Effective and Efficient Federated Tree Learning on Hybrid Data

Qinbin Li[1], Chulin Xie[2], Xiaojun Xu[2], Xiaoyuan Liu[1], Ce Zhang[3,4], Bo Li[2,3], Bingsheng He[5], Dawn Song[1]
[1]UC Berkeley, [2]UIUC, [3]University of Chicago, [4]Together AI, [5]National University of Singapore

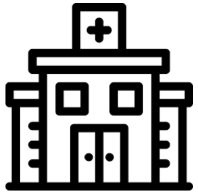
Tree Models are Powerful and Efficient



$$y_p = 0.664 + 0.234 = 0.898$$

GBDT [3]

Credit risk assessment, pricing…

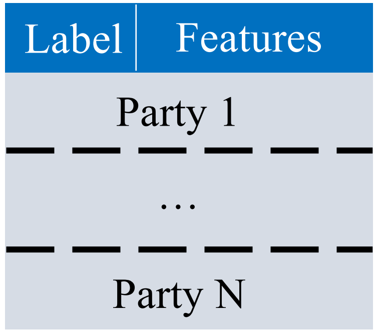


sepsis, cardiovascular…

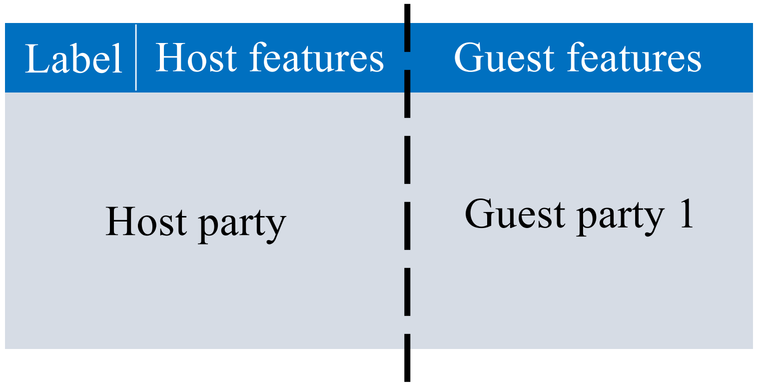
kaggle champions

[3] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016.

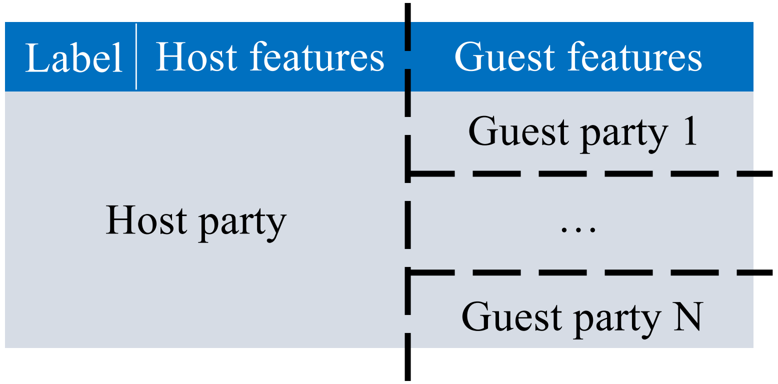
Federated GBDT on Hybrid Data



Horizontal



Vertical



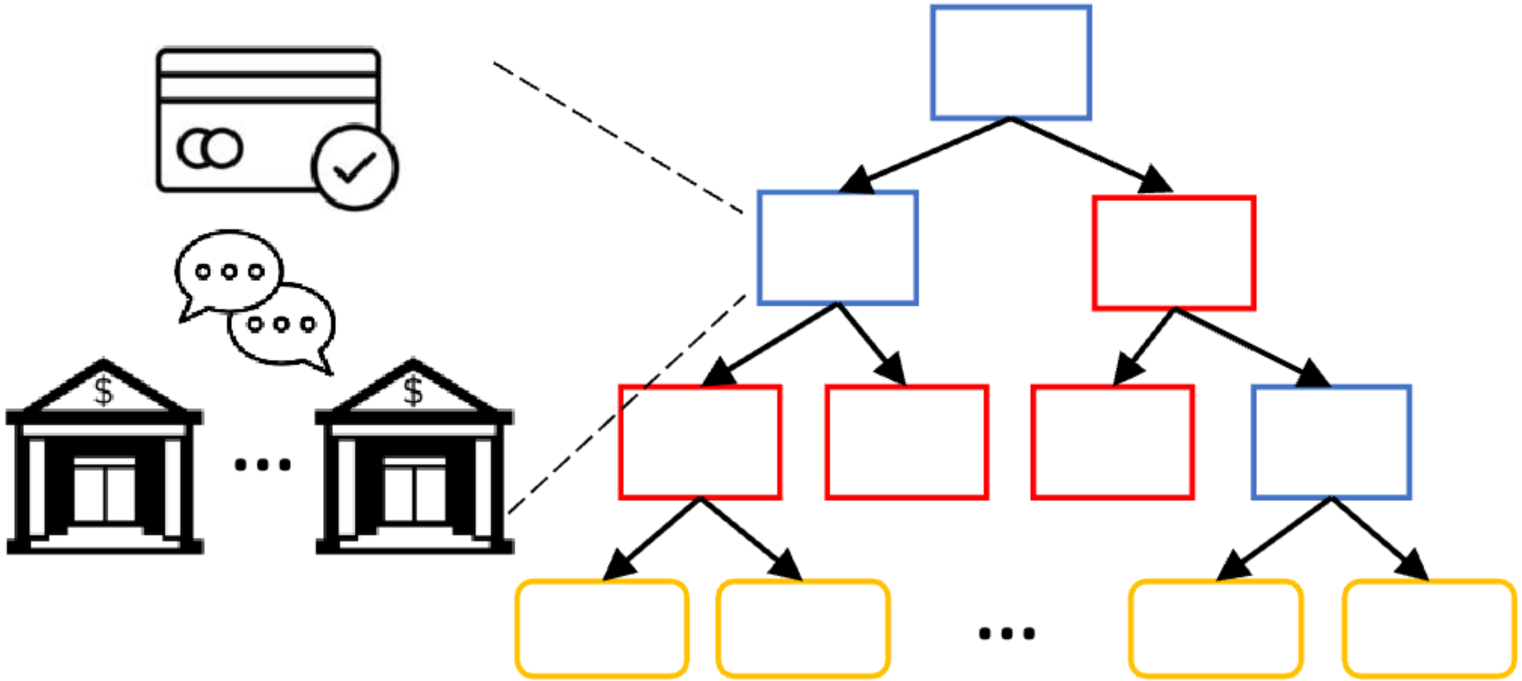
Hybrid

Host party: a payment system (e.g., SWIFT)
Guest party: bank

Node-level solution

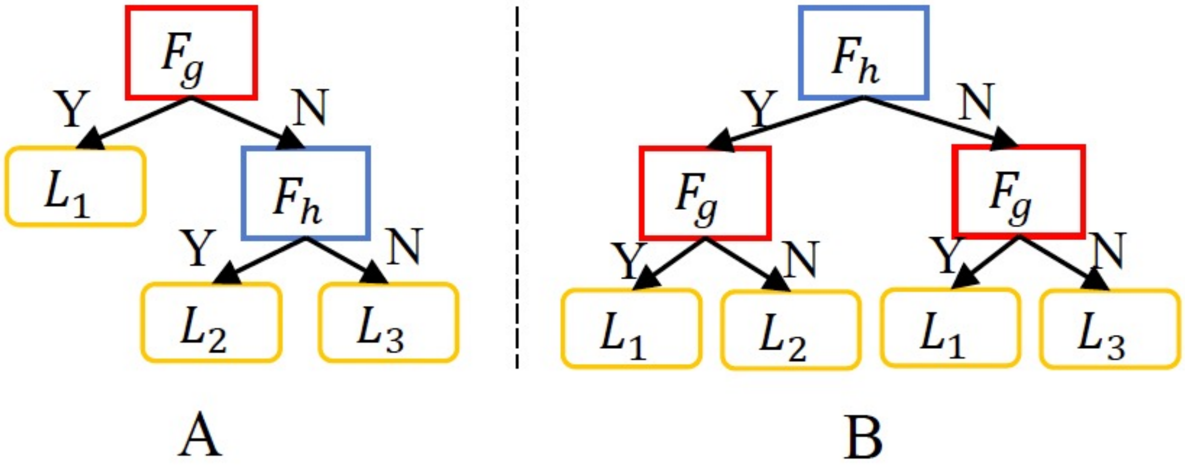
- Aggregating gradients in each node using cryptographic methods.

Huge computation cost



Tree Transformation

Theorem 2. *Suppose F_g is a meta-rule in Tree A. For any input instance $\boldsymbol{x} \in \mathcal{D}$, we have $E[f(\boldsymbol{x};\theta_A)] = E[f(\boldsymbol{x};\theta_B)]$, i.e., the expectation of prediction value of Tree A and Tree B are the same.*

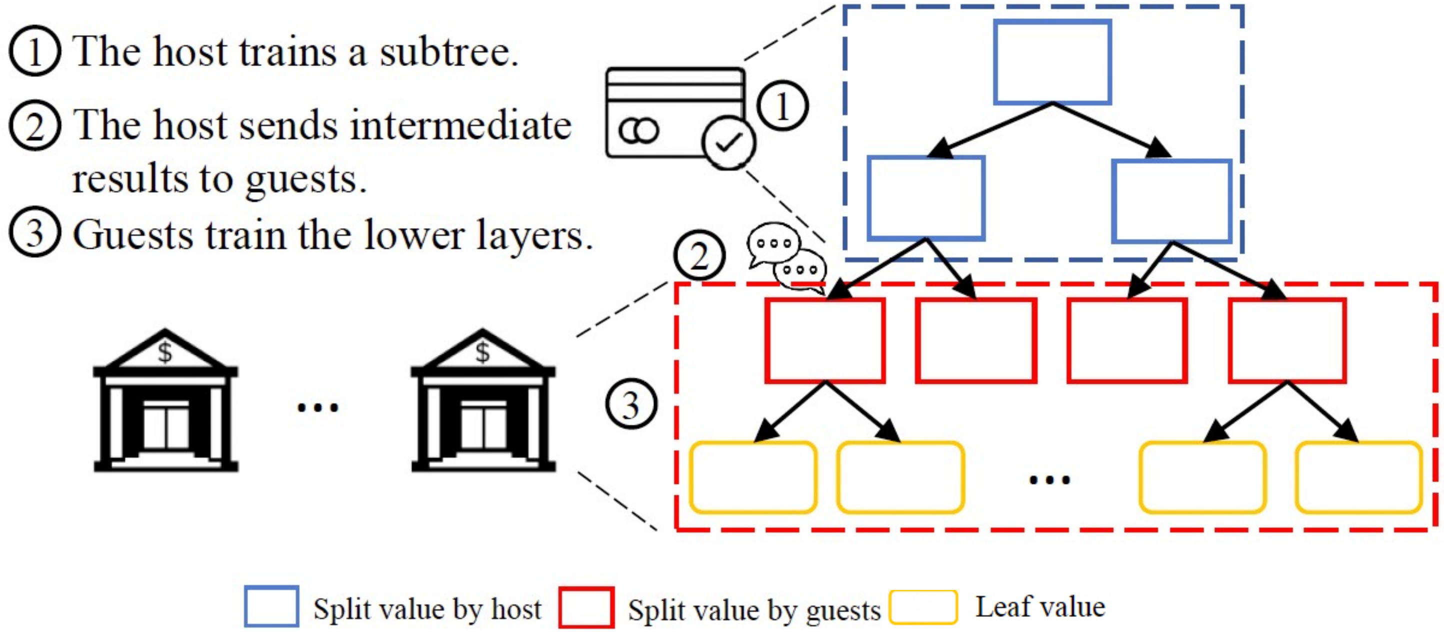


Theorem 3. *Suppose $S_m := F_h \cap \ldots \cap F_g$ is a meta-rule in tree θ_A where F_g is a split condition using the feature from the guests. For any tree path in tree θ_A involving the split nodes in S_m, we can always reorder the split nodes in the tree path such that F_g is in the last layer. Moreover, naming the tree after the reordering as θ_B, we have $E[f(\boldsymbol{x};\theta_A)] = E[f(\boldsymbol{x};\theta_B)]$ for any input instance $\boldsymbol{x} \in \mathcal{D}$.*

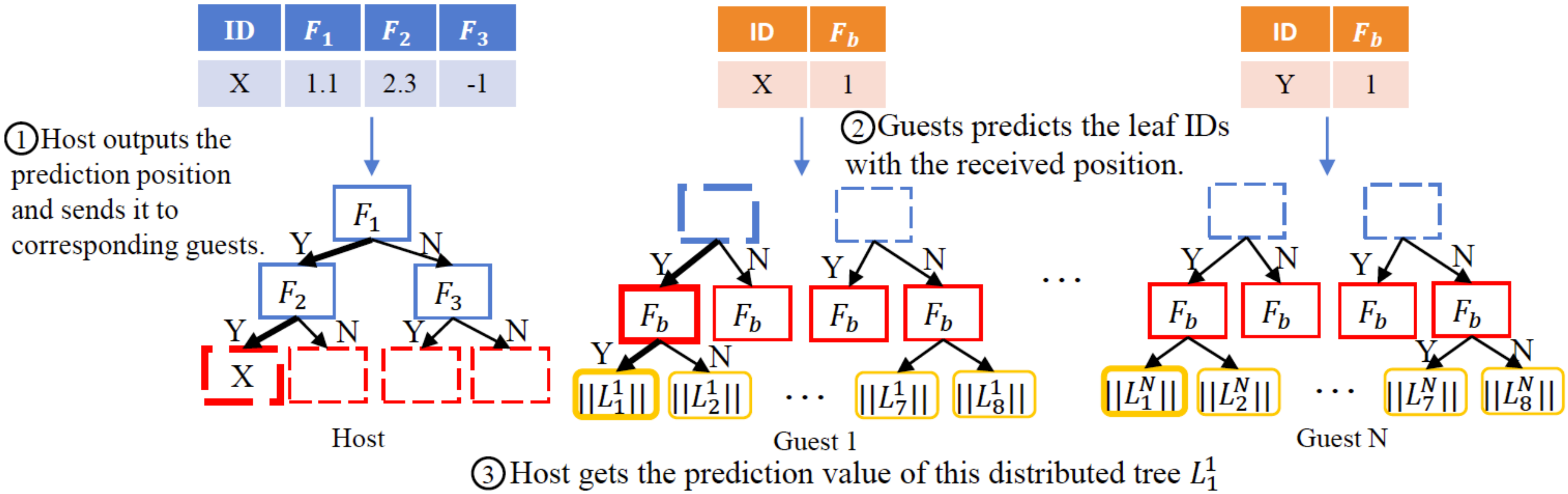
Move the split feature of the guest parties to the last layer.

Layer-wise Training

- Each party trains a subtree individually
- No gradient aggregation in each node



Inference



ID | F_1 | F_2 | F_3
X | 1.1 | 2.3 | -1
ID | F_b
X | 1
ID | F_b
Y | 1
①Host outputs the prediction position and sends it to corresponding guests.
②Guests predicts the leaf IDs with the received position.
F_1
F_2
F_3
F_b
Y
N
X
$||L_1^1||$ $||L_2^1||$ … $||L_7^1||$ $||L_8^1||$
$||L_1^N||$ $||L_2^N||$ … $||L_7^N||$ $||L_8^N||$
Host
Guest 1
Guest N
③Host gets the prediction value of this distributed tree L_1^1

Experiments

- Datasets: 1) Hybrid datasets: AD, DEV-AD; 2) Simulated datasets: Adult, Cod-rna
- Approaches:
- 1) ALL-IN: centralized training
- 2) SOLO: local training
- 3) Two-party VFL: FedTree, SecureBoost, Pivot
- 4) TFL: tree-level aggregation

Effectiveness

	HybridTree	SOLO	FedTree	SecureBoost	Pivot	TFL	ALL-IN
AD	**0.689**	0.492	0.537-0.566	0.537-0.566	0.534-0.561	0.530	0.703
DEV-AD	**0.553**	0.111	0.412-0.462	0.412-0.462	0.414-0.468	0.397	0.574
Adult	**0.832**	0.653	0.764-0.788	0.764-0.788	0.755-0.778	0.773	0.853
Cod-rna	**0.927**	0.690	0.805-0.863	0.805-0.863	0.811-0.870	0.884	0.931

Efficiency

	Communication size (GB)					Training time (s)				
	HybridTree	FedTree	SecureBoost	Pivot	speedup	HybridTree	FedTree	SecureBoost	Pivot	speedup
AD	223.6	1363.9	1389.2	1420.3	**6.1x**	84.1	595.6	3212.7	316823	**7.1x**
DEV-AD	142.6	770.1	681.9	792.2	**5.4x**	58.2	464.9	2856.6	284235	**8.0x**
Adult	1.55	9.74	14.6	11.9	**6.3x**	2.0	8.6	71.1	9234	**4.3x**
Cod-rna	2.84	15.92	20.4	18.5	**5.6x**	1.0	5.3	24.3	3845	**5.3x**