How to Learn Collaboratively - Federated Learning to Peer-to-Peer Learning and What's at Stake

Atul Sharma, Joshua C. Zhao, Wei Chen, Qiang Qiu, Saurabh Bagchi, Somali Chaterji
Purdue University
{sharm438,zhao1207,chen2732,qqiu,sbagchi,schaterji}@purdue.edu

***Abstract*—**

Standard ML relies on training using a centrally collected dataset, while collaborative learning techniques such as Federated Learning (FL) enable data to remain decentralized at client locations. In FL, a central server coordinates the training process, reducing computation and communication expenses for clients. However, this centralization can lead to server congestion and heightened risk of malicious activity or data privacy breaches. In contrast, Peer-to-Peer Learning (P2PL) is a fully decentralized system where nodes manage both local training and aggregation tasks. While P2PL promotes privacy by eliminating the need to trust a single node, it also results in increased computation and communication costs, along with potential difficulties in achieving consensus among nodes. To address the limitations of both FL and P2PL, we propose a hybrid approach called Hubs-and-Spokes Learning (HSL). In HSL, hubs function similarly to FL servers, maintaining consensus but exerting less control over spokes. This paper argues that HSL's design allows for greater availability and privacy than FL, while reducing computation and communication costs compared to P2PL. Additionally, HSL maintains consensus and integrity in the learning process.

1. Introduction

1.1. Federated Learning (FL)

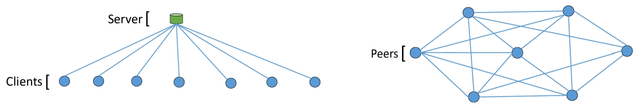


Figure 1: Left: Federated Learning (FL) network; Right: Peer-to-Peer Learning (P2PL) network. In FL, clients receive a global model, train it locally, and send model gradients to the server for aggregation and global model updates. In P2PL, clients exchange trained models with neighbors and independently aggregate received models.

An FL system, as shown in Figure 1, consists of a server that communicates periodically with n client nodes. The server initiates training by distributing the same model to all clients. Clients (all or a subset) train the model locally on their private data and periodically share model updates with the server, which aggregates them and returns the global model. This iterative process optimizes the global objective function $\frac{1}{n}\sum_i^n f_i(x)$, where f_i represents the expectation of client i's local objective, averaged over data batches in its dataset D_i, using the model x received from the server. The server ensures *exact* consensus among all nodes at the synchronization points by sharing the updated global model to all clients. The FedSGD algorithm optimizes FL in its most basic form [1].

Since its introduction in 2017 [1], FL has attracted research interest in various areas, such as theoretical ML [2], systems [3], and security [4]. The learning community has studied different learning objectives [5] and optimizers [6] to improve the convergence of the learning process. Computer systems studies have also proposed multiple aggregation algorithms [7], [8] to handle heterogeneous clients in asynchronous systems. A large portion of the literature is also devoted to the study of FL systems under defined threat models [9], [10], [11].

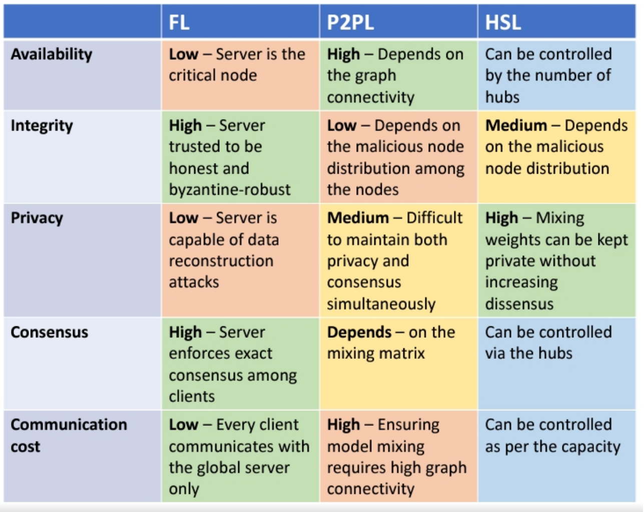
1.2. Peer-to-peer Learning (P2PL)

P2PL is a fully decentralized system where the nodes communicate as peers without the need for a central server, as illustrated in Figure 1. This eliminates the need for a single point of trust in the system, enabling greater personal control and transparency at the nodes. As edge devices become increasingly computationally powerful, model aggregation can be delegated to the learning nodes themselves rather than being restricted to a single server. The Decentralized-SGD (D-SGD) [12] algorithm optimizes vanilla P2PL. In each learning round, connected node pairs gossip, exchange models, and update their models to the average of the received models, including their own. Gossip averaging helps maintain approximate consensus among nodes, achieving exact consensus only when every pair of nodes is connected. However, Assran *et al.* [13] demonstrated that exact consensus is not necessary, and nodes can collaboratively learn with differing models. To avoid chaos, consensus distance among nodes must be bounded, ensuring that local models are relatively similar and in approximate agreement on the learning trajectory. Kong *et al.* [14] recommends controlling consensus by incorporating multiple gossip averaging steps in each communication round. This ensures that consensus remains within calculated bounds for different phases of training.

Mixing matrix W for gossip learning. - In gossip learning, nodes exchange and combine their model information with others. A "mixing matrix" (W) helps determine how much importance a node gives to each received model. The mixing weights in the matrix add up to 1, ensuring that the combined model maintains the appropriate scale. The mixing weights for node i are placed in the i^{th} row of a matrix, forming the mixing matrix W. If X is a matrix where row i represents the model weights of node i, the updated model weights after a single gossip round can be represented by the matrix WX. When nodes participate in multiple gossip rounds, the mixing matrix is raised to the power of the number of rounds (g). (Two rounds of gossip would result in $W(WX) = W^2X$.) As the number of rounds increases, the matrix becomes denser, leading to better mixing of local models. This helps nodes maintain an approximate agreement, even when they do not have the exact same model information [14].

Column stochasticity of W. - The sum of column j in the mixing matrix W represents node j's contribution in a single gossip round, based on the weights assigned to j's model by all nodes. Recent works [12], [14] recommend constraining W to be doubly stochastic. This means that both the row and column sums of matrix W should equal 1, ensuring fairness and balance in the gossip learning process. By requiring both row and column sums to equal 1, the doubly stochastic constraint guarantees that: Each node's updated model is an equal combination of the received models, ensuring that no single model dominates the learning process. Each node's contribution to the system is equal, preventing any node from disproportionately influencing the overall learning process. Further, this constraint of column stochasticity makes it simpler to mathematically analyze the complex learning process in a decentralized setting, even when the network topologies are changing [12]. Specifically, a doubly stochastic W has the property that $\mathbf{1}^T W = \mathbf{1}^T$, where $\mathbf{1}$ is a column vector of one with the same dinension as the number of rows in W. The operator $\mathbf{1}^T$ is useful in analyzing sum-like functions such that the average model weights after gossiping can be expressed as $\frac{1}{n}\mathbf{1}^T WX = \frac{1}{n}\mathbf{1}^T X$. If W is column stochastic, the operator eliminates W.

Although column stochasticity is useful in analyzing convergence in simplified cases, it is not a necessary condition for a P2PL system's convergence, as evident from practical experiments. This is significant because imposing column stochasticity means constraining the contribution of every node to be equal, which would involve artificially increasing the contribution of less popular nodes and limiting that of more popular nodes. In a malicious setting or a non-IID setting, it is not advisable to require equal contributions from all nodes, as this may not be the most optimal approach.



	FL	P2PL	HSL
Availability	**Low** – Server is the critical node	**High** – Depends on the graph connectivity	Can be controlled by the number of hubs
Integrity	**High** – Server trusted to be honest and byzantine-robust	**Low** – Depends on the malicious node distribution among the nodes	**Medium** – Depends on the malicious node distribution
Privacy	**Low** – Server is capable of data reconstruction attacks	**Medium** – Difficult to maintain both privacy and consensus simultaneously	**High** – Mixing weights can be kept private without increasing dissensus
Consensus	**High** – Server enforces exact consensus among clients	**Depends** – on the mixing matrix	Can be controlled via the hubs
Communication cost	**Low** – Every client communicates with the global server only	**High** – Ensuring model mixing requires high graph connectivity	Can be controlled as per the capacity

TABLE 1: A comparison of Federated Learning (FL), Peer-to-Peer Learning (P2PL), and the proposed HSL Hubs-and-Spokes Learning (HSL) architecture for collaborative learning. FL provides high integrity but limited privacy and availability assurances. P2PL delivers high availability and privacy but incurs high communication costs and challenges in consensus maintenance. HSL allows configurable levels of availability, consensus, and communication cost while preserving privacy and integrity.

2. Enlarged Attack Surface of P2PL

Here we first describe the fundamental reasons that enable attacks on an FL system, and then move forward to describe how it is even more difficult to maintain security in P2PL because of inexact consensus. Table 1 summarizes the features of FL and P2PL and compare them with HSL.

FL availability. - Federated Learning (FL) systems have a critical vulnerability in that the server node represents a single point of failure in the network topology. To reduce congestion and prevent failure, servers use intelligent client selection algorithms [15]. While necessary for system availability, this approach can negatively impact the speed of convergence.

FL integrity. - The server also holds a high level of trust relative to the client nodes with clients that do not trust one another, resulting in communication solely with the server. Clients must trust the server to be both benign and capable of maintaining byzantine-robustness in the face of malicious client nodes. If a server succumbs to a poisoning attack [16], it could disseminate the infected model to all clients, potentially derailing the entire training process. Current research focuses on maintaining byzantine-robust model aggregation algorithms, ensuring the integrity of the global model at the server. State-of-the-art defense techniques [4], [11] defend against directed deviation attacks [16], [17], which are the most advanced model poisoning attacks. While FL system integrity is currently considered a solved problem, the ongoing cycle of attack and defense may require increasingly robust aggregation techniques in the future.

FL privacy. - Privacy risks arise when clients train the server-provided model on their local data, as a curious server could attempt to reconstruct a client's local data, breaching privacy. Such attacks are possible when an entity, like the server, has knowledge of a client's initial and final model states during local training. In the case of FL, this is directly accessible to the server since the initial model is sent by the server and the update is also sent back to the server. Optimization-based data reconstruction attacks [18], [19], [20] can recover data samples during the FedSGD process, while analytic data reconstruction attacks [21], [22], [23] can work in the FedAvg setting and under secure aggregation [24]. Adding random noise [25] can help obscure reconstruction but degrades utility. Homomorphic encryption techniques [26] can also enhance privacy but are limited to mean aggregation, which is not byzantine-resilient.

The root cause of privacy attacks is the high level of trust enjoyed by the server, which affords it sufficient power to potentially extract privacy-sensitive information from clients. P2PL provides an opportunity to remove this level of trust from a single node and distribute it among multiple neighboring nodes, provided that neighbors do not collectively collude maliciously.

P2PL availability. - P2PL systems inherently lack a central server, eliminating a single point of failure. While the graph formed by collaborating nodes may have critical edges, P2PL generally offers higher availability than FL at the cost of transferring the aggregation responsibility to participating nodes, which communicate directly with each other.

P2PL integrity. - In P2PL, maintaining the integrity of the local models is each node's responsibility. Nodes can opt to use byzantine-robust aggregations from the FL domain, but extending such aggregations to P2PL is non-trivial and has not yet been explored in the literature. Some FL algorithms [11], [27] preserve byzantine-robustness by assuming an upper bound on the fraction f_{max} of potentially malicious or compromised client nodes, beyond which no guarantees can be made for model integrity. However, in the P2P setting, the distribution of malicious nodes does not need to be uniform. Given a non-uniform distribution of malicious nodes, conservatively setting f_{max} higher than the actual fraction of malicious nodes in the entire collaborating population may still leave some nodes unable to maintain byzantine-robustness due to a higher concentration of malicious nodes in their neighborhood. This problem is more easily addressed in FL, where the server communicates with all other nodes. In P2PL, collaborating nodes, limited by the number of connections they can have, must cope with a variable fraction of malicious nodes in each neighborhood. Other FL algorithms [4], [28] solve the integrity problem by computing some anomaly statistic for the nodes and detecting variable number of malicious nodes based on certain rules. However, controlling consensus in such systems is challenging. As shown in [11], these algorithms can be overly conservative, leading to high false positive malicious detection rates, reduced mixing, and increased dissensus among nodes, which eventually classify an increasing number of non-consensus nodes as malicious.

Freedom vs control in P2PL. - When a node cannot trust any neighboring nodes, it requires freedom of choice in assigning mixing weights. Such freedom requires relaxation of at least the doubly stochastic constraint on the mixing matrix W, which makes mathematical analysis complex. Unrestricted freedom in selecting mixing weights can result in a W that exacerbates dissensus and potentially leads to chaos. Maintaining byzantine-robustness becomes even more difficult when each node demands the freedom to choose weights. Further research in P2PL is needed to identify the permissible properties matrix W should possess for the system to be byzantine-robust and for benign nodes to achieve approximate consensus. To summarize, maintaining consensus and byzantine-robustness simultaneously in P2PL is difficult due to the absence of a controlling structure. We use this argument to motivate a two-layered HSL structure discussed later in the paper.

P2PL privacy. - P2PL offers a means to maintain local data privacy at the nodes. Each node iteratively trains the locally aggregated model on its data before gossiping with neighbors. By not disclosing their mixing weights, nodes can conceal their locally aggregated models after each gossiping step. Although the final state of the model after local training is still shared with neighbors, concealing the mixing weights hides the initial model state, preventing data reconstruction attacks and privacy breaches. However, this reintroduces the challenge of guaranteeing consensus when nodes have unrestricted freedom to choose their mixing weights. The extent of mixing occurring after each gossip round depends on the matrix W, specifically its spectral gap [14]. Without oversight on consensus or on W, it cannot be ensured that the models for each node will not diverge and that they will mix well to learn the same model. It is evident that diverging models are negatively affected by collaboration unless sufficient mixing happens. Thus, we understand that ensuring consensus is an important aspect for maintaining both byzantine-robustness and privacy in any form of collaborative learning. Bearing these considerations in mind, we propose a promising direction to address some of these challenges.

3. Promising Solution Direction

We have now seen that FL and P2PL are two extreme forms of collaborative learning, each with its own limitations. In FL, a powerful server enforces exact consensus and Byzantine-robustness. However, this same authority also heightens the risk of breaching client privacy. On the other hand, the completely decentralized network topology in P2PL makes maintaining integrity and privacy challenging due to the non-zero consensus distance among peer nodes.

To overcome these challenges, we propose a hybrid Hubs-and-Spokes-Learning (HSL), as shown in Figure 2. HSL is composed of two layers: a layer of client-like nodes (spokes) and another of server-like nodes (hubs). This inno-

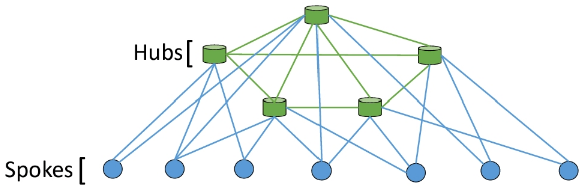


Figure 2: This figure illustrates **HSL**'*s Hubs-and-Spokes Learning (HSL) topology for collaborative learning. Each spoke communicates with two or more hubs, while hubs gossip with each other. Spokes do not directly communicate with one another. Hubs receive locally trained models from the spokes, engage in gossip for model mixing, and transmit aggregated models to connected spokes.*

vative architecture fosters collaboration while maintaining privacy and integrity.

Privacy in HSL. - In HSL, spokes communicate exclusively with their parent hubs, while hubs can gossip with one another. A many-to-many connection exists between the hubs and spoke layers. Each spoke receives models from its parent hubs (with a recommended minimum of two) and locally aggregates them using private mixing weights. This approach prevents a single hub from dictating its model to a child spoke, which is the root cause of data leakage attacks. By keeping mixing weights private, spokes can hide their model's initial state and share the final state after local training without any privacy concerns. We assume that the hubs are not fully connected, because an exact consensus among hubs will result in identical models at all hubs, irrespective of the mixing weights used by spokes.

Hubs, similar to FL servers, do not perform local training, but are responsible for Byzantine-robust aggregation of models received from spokes. Hubs primarily act as a conduit for spokes to exchange information effectively. After aggregation, hubs gossip among themselves to reach approximate consensus.

Consensus control. - The collaborative HSL architecture can be viewed as multiple, interconnected FL systems, where the number of hubs can be orders of magnitude smaller than the number of spokes. The fundamental concept used in our design is that consensus among spokes can be controlled through consensus among hubs. As every spoke's model is a weighted mean of hub models, consensus among hubs bounds the consensus among spokes without imposing constraints on the spokes' mixing weights. Achieving consensus through repeated gossiping among a smaller population of nodes (hubs)compared to a larger one (spokes) also significantly lowers computation and communication costs.

Figure 3 demonstrates the improvement in collaborative learning with HSL using 64 spokes and 5 hubs, with 5 edges among hubs and 128 edges between spokes and hubs (satisfying the minimum requirement of 2 hubs per spoke) compared to a P2PL system with 150 randomly sampled edges. This improvement, observed with a 0.5 non-IID bias and 1 gossip step per round on CIFAR-10, is attributable to better consensus control in HSL.

Availability and communication cost. - It is important to note that HSL with a single hub is functionally equivalent to FL, while HSL with n spokes and n hubs with one-to-one connections functionally represents P2PL. HSL serves as a more generalized framework that encompasses FL and P2PL. By adjusting the number of hubs, HSL can be configured to meet specific availability and communication cost objectives, given finite individual budgets.

Integrity in HSL. - HSL incorporates three levels of Byzantine-robust aggregation. Although hubs may become compromised if malicious spokes are concentrated in certain neighborhoods, we propose a gossip mechanism that allows hubs to provide feedback to each other. This mechanism enables hubs to increment their f_{max} if they suspect a higher number of malicious spokes. Failure to act on the feedback could result in a hub being identified as malicious by its peer hubs. In addition, the spokes have access to their own model as a benign ground truth for robust aggregation. Since consensus among hubs ensures consensus among spokes, the system is less likely to spiral into chaos, unlike a P2PL system.

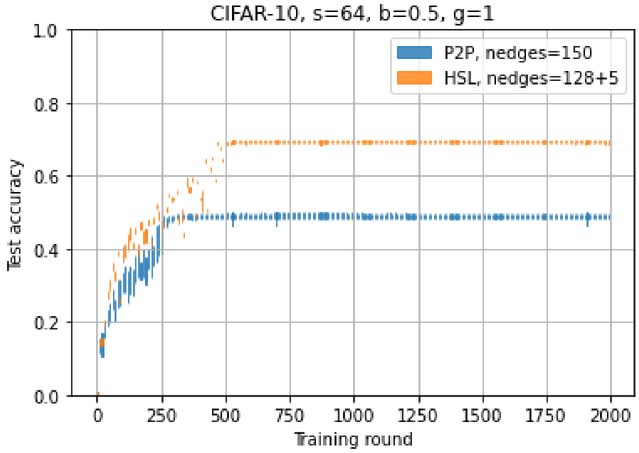


Figure 3: Comparison of the average performance of HSL (5 hubs, 133 edges) and P2PL (150 edges), illustrating test accuracy candles for 64 spokes training ResNet-18 on CIFAR-10 with a single gossip step per round. HSL achieves significant improvement due to superior consensus, even at a lower communication cost compared to P2PL.

4. Conclusion

In this paper, we have highlighted the inherent structural and functional vulnerabilities within the two most prevalent forms of decentralized machine learning, Federated Learning (FL) and Peer-to-Peer Learning (P2PL). We argue that these two forms represent the extremes on a spectrum, motivating our hybrid architecture as a potential resolution to these existing challenges. We have discussed with logical arguments how this proposed architecture addresses the current issues in FL and P2PL. With this discussion, we aim to inspire further research toward more pragmatic forms of collaborative learning, steering away from the extremes.

Further, we have also emphasized the need and possibility for such solutions to be secure in terms of privacy, integrity, and availability for harnessing the potential of large volumes of data and ubiquitous computing.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under Grant Numbers CNS-2146449 (NSF CAREER award), CNS-2038986, and CCF-1919197. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[2] A. Reisizadeh, F. Farnia, R. Pedarsani, and A. Jadbabaie, "Robust federated learning: The case of affine distribution shifts," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 554–21 565, 2020.

[3] F. Lai, Y. Dai, S. Singapuram, J. Liu, X. Zhu, H. Madhyastha, and M. Chowdhury, "Fedscale: Benchmarking model and system performance of federated learning at scale," in *International Conference on Machine Learning*. PMLR, 2022, pp. 11 814–11 827.

[4] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," *Network and Distributed System Security Symposium*, pp. 1–18, 2021.

[5] R. Pathak and M. J. Wainwright, "Fedsplit: An algorithmic framework for fast federated optimization," *Advances in neural information processing systems*, vol. 33, pp. 7057–7066, 2020.

[6] H. Yuan and T. Ma, "Federated accelerated stochastic gradient descent," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5332–5344, 2020.

[7] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection." in *OSDI*, 2021, pp. 19–35.

[8] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, and D. Huba, "Federated learning with buffered asynchronous aggregation," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 3581–3607.

[9] R. Guerraoui, S. Rouault *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3521–3530.

[10] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5650–5659.

[11] A. Sharma, W. Chen, J. Zhao, Q. Qiu, S. Bagchi, and S. Chaterji, "Flair: Defense against model poisoning attack in federated learning," *Proceedings of the ACM Asia Conference on Computer and Communications Security*, 2023.

[12] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, "A unified theory of decentralized sgd with changing topology and local updates," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5381–5393.

[13] M. Assran, N. Loizou, N. Ballas, and M. Rabbat, "Stochastic gradient push for distributed deep learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 344–353.

[14] L. Kong, T. Lin, A. Koloskova, M. Jaggi, and S. Stich, "Consensus control for decentralized deep learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5686–5696.

[15] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 10 351–10 375.

[16] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *Proceedings of the 29th USENIX Conference on Security Symposium*, 2020, pp. 1623–1640.

[17] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *NDSS*, 2021.

[18] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 937–16 947, 2020.

[19] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 337–16 346.

[20] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[21] L. Fowl, J. Geiping, W. Czaja, M. Goldblum, and T. Goldstein, "Robbing the fed: Directly obtaining private data in federated learning with modified models," *arXiv preprint arXiv:2110.13057*, 2021.

[22] J. C. Zhao, A. Sharma, A. R. Elkordy, Y. H. Ezzeldin, S. Avestimehr, and S. Bagchi, "Secure aggregation in federated learning is not private: Leaking user data at large scale through model modification," *arXiv preprint arXiv:2303.12233*, 2023.

[23] J. C. Zhao, A. R. Elkordy, A. Sharma, Y. H. Ezzeldin, S. Avestimehr, and S. Bagchi, "The resource problem of using linear layer leakage attack in federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[24] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[25] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[26] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "Batchcrypt: Efficient homomorphic encryption for cross-silo federated learning," in *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC 2020)*, 2020.

[27] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "Faba: an algorithm for fast aggregation against byzantine attacks in distributed neural networks," in *IJCAI*, 2019.

[28] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings." in *RAID*, 2020, pp. 301–316.