

Creative Commons Attribution 4.0 International (CC BY 4.0)

<https://creativecommons.org/licenses/by/4.0/>

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

**Please provide feedback**

Please support the ScholarWorks@UMBC repository by emailing [scholarworks-group@umbc.edu](mailto:scholarworks-group@umbc.edu) and telling us what having access to this work means to you and why it's important to you. Thank you.

---

# LEVERAGING MIXED PRECISION QUANTIZATION TO TACKLE GRADIENT LEAKAGE ATTACKS IN FEDERATED LEARNING

---

**Pretom Roy Ovi**

Department of Information Systems  
University of Maryland, Baltimore County  
povi1@umbc.edu

**Emon Dey**

Department of Information Systems  
University of Maryland, Baltimore County  
edey1@umbc.edu

**Nirmalya Roy**

Department of Information Systems  
University of Maryland, Baltimore County  
nroy@umbc.edu

**Aryya Gangopadhyay**

Department of Information Systems  
University of Maryland, Baltimore County  
gangopad@umbc.edu

## ABSTRACT

Federated Learning (FL) enables collaborative model building among a large number of participants without the need for explicit data sharing. But this approach shows vulnerabilities when privacy inference attacks are applied to it. In particular, in the event of a gradient leakage attack, which has a higher success rate in retrieving sensitive data from the model gradients, FL models are at higher risk due to the presence of communication in their inherent architecture. The most alarming thing about this gradient leakage attack is that it can be performed in such a covert way that it doesn't hamper the training performance while the attackers backtrack from the gradients to get information about the raw data. Two of the most common approaches proposed as solutions to this issue are homomorphic encryption and adding noise with differential privacy parameters. These two approaches suffer from two major drawbacks. They are: the key generation process becomes tedious with the increasing number of clients, and noise-based differential privacy suffers from a significant drop in global model accuracy. As a countermeasure, we propose a mixed-precision quantized FL scheme, and we empirically show that both of the issues addressed above can be resolved. In addition, our approach can ensure more robustness as different layers of the deep model are quantized with different precision and quantization modes. We empirically proved the validity of our method with three benchmark datasets and found a minimal accuracy drop in the global model after applying quantization.

## 1 Introduction

Federated Learning (FL) has emerged as an alternative to the centralized approach of building a machine learning model, which introduces collaborative training among multiple clients while keeping their dataset private. Although the inherent architecture of FL eliminates the need for explicit data sharing, it still shows vulnerabilities during privacy and robustness attacks. Among the present adversaries in the context of FL, the gradient leakage attack is considered one of the most harmful ones because attackers can successfully reconstitute sensitive training data by secretly snooping on gradient updates during iterative training[1, 2], all without affecting model training quality.

Two of the most explored prevention methods in the present literature are differential privacy (DP) and homomorphic encryption (HE). First, in the differential privacy approach [3, 4], to protect the confidentiality of training sample data, Gaussian or laplacian noise is added with gradients during training. But in this method, the expense of accuracy deteriorates below the threshold level sometimes, while preserving privacy. The second one is homomorphic encryption [5, 6], where key-based encryption is proposed. In this approach, either separate keys are allocated for different participants or the same key for all participants has to be set. But generating different keys for each client elevates the computation complexity. On the contrary, to ensure all clients are getting the same key, we have to enable key sharing among the clients, but such communication is not desirable in FL.

Moreover, researchers have been investigating quantization, typically a model compression scheme to reduce the computation resource requirement of deep models. We point out a new use case of the quantization approach in tackling the gradient leakage attack. In quantization, the gradient values are transitioned into a less precise form according to our choice of bit size. Unless the attacker has some knowledge about the range information of the unquantized gradient, it is highly unlikely to retrieve some sensitive raw data information. We have chosen mixed-precision over single-precision quantization to make our resistance algorithm more robust. This is because, in mixed-precision, the number of iterations to estimate a hyperparameter jumps up to the power of the layer number of the model. Thus, it can make the data extracting process significantly resource exhaustive for the attackers.

Our approach can overcome the disadvantages introduced in DP and HE methods in the sense that, in our approach, it is possible to dequantize the shared quantized gradients at the server end prior to aggregation to maintain almost the same performance level. Since dequantization is not a fully reversible operation, the accuracy may decrease slightly, but it is still superior to differential privacy's accuracy. Also, due to the compressed nature of the quantization, the gradient size required for transmission is less, thus countering the size expansion issue of HE. The specific scientific contributions we offer here are:

- We conduct a detailed risk assessment in federated learning (FL) scenario due to a gradient leakage attack and propose a quantization-enabled solution to secure a more robust FL framework. Specifically, our approach is built upon the concept of mixed-precision quantization, which is applied to the gradients during the transmission phase.
- We empirically demonstrate the applicability of our proposed algorithm with three of the most popular FL datasets. In addition, a comprehensive baseline comparison is performed, and we achieve an average 10% increase in accuracy while keeping the attack success rate to a bare minimum.
- We present a pertinent ablation study to determine the impact of different hyperparameters used in our federated framework. We find that along with the traits of attack resiliency and accuracy retention, our method can offer another desirable property of reduced communication cost.

## 2 Related work

In Federated learning, the server and clients exchange gradients after each round of training. According to [7, 8], gradients reveal some properties of the training data. Moreover, recent studies [1, 2, 9, 10] demonstrated some approaches that can completely steal the training data from gradients with gradient leakage attack. And existing solutions against gradient leakage can be divided into two groups- one is differential privacy [3, 11, 4] and other one is encryption [6, 12].

Differential privacy offers statistical guarantees against the information an attacker can infer from the output of a randomized algorithm. Differential privacy on a client level is feasible, and high model accuracy can be reached when sufficiently many clients (around 1000) are involved [11], but specially in cross-silo federated setup, there may not be thousands of clients. DP is effective when each client has access to a large dataset [13], but FL setup can't guarantee large training datasets for each client. Furthermore, the quantity of accessible data can vary among clients. Authors in [14] demonstrated FL with Bayesian DP in a context where data is similarly distributed across participating clients. But FL can't guarantee similar data distribution among all clients. While providing a certain level of differential privacy guarantee, DP limits the performance accuracy of deep learning models. In addition, algorithms require careful privacy parameter selection; otherwise, gradient leakage-induced privacy breach is possible.

The encryption algorithms often used in FL can be broadly classified as Homomorphic Encryption (HE) [5, 6] and Secure Multi-party Computing (SMC) [12, 15]. While preserving the privacy of training data samples, HE theoretically ensures no performance loss in terms of model convergence [5, 6]. However, the effectiveness of HE comes at the expense of computation and memory [16], which limits its application. Then the data-size of the encrypted models increases linearly with each homomorphic operation [17]. Thus, the encrypted models are significantly larger which increases the communication costs. On the other hand, Secure Multi-party Computing (SMC) in FL scenario requires each worker to coordinate with each other during the training process, which is usually impractical.

## 3 Methodology

In this section, we describe the detailed working procedure of our proposed federated learning approach along with integration of mixed precision quantization.

We describe the individual operations carried out on clients (data owner) and server (model owner), the two components of any FL system. Our proposed built on top of the FedAvg[18] algorithm and some of the notations used in this

description are  $\mathcal{N} = \{1, \dots, N\}$  signify the set of  $N$  clients, each of which has their own dataset  $D_{k \in \mathcal{N}}$ . Each of them trains a local model using their own dataset and only shares the model gradients to the FL server. Then, the global model formation takes place with all the local model gradient  $\Delta W_G$  updates which can be denoted by  $\Delta W = \cup_{k \in \mathcal{N}} \Delta W_k$ . The complete pseudocode of our method is shown in algorithm 1 and described below:

---

**Algorithm 1** Proposed Federated Learning algorithm
 

---

**Require:** Clients number  $n$  per iteration, local epoches number  $E$ , and learning rate  $\mu$ , Local minibatch size  $B$ , Total number of iteration  $T$

**Ensure:** Global model  $\mathbf{W}_G$ .

- 1: [Step 1](Server)
- 2: Initialize  $\mathbf{w}_G^0$
- 3: [Step 2](Client)
- 4: **LocalTraining**( $i, \Delta W$ ) :
- 5: Split local dataset  $D_i$  to minibatches of size  $B$  which are included into the set  $\mathcal{B}_i$ .
- 6: **for** each local epoch  $j$  from 1 to  $E$  **do**
- 7:     **for** each  $b \in \mathcal{B}_i$  **do**
- 8:          $\mathbf{W} \leftarrow \mathbf{W} - \mu \Delta W(\mathbf{W}; b)$   $\triangleright$  ( $\mu$  is the learning rate,  $\Delta W$  is the gradient of  $L$  on  $b$ )
- 9:     **end for**
- 10: **end for**
- 11: **Gradient Quantization:**  $\Delta W_q \leftarrow \text{Quantize}(\Delta W)$
- 12: [Step 3](Server)
- 13: **Gradient Dequantization:**  $\Delta W^t \leftarrow \text{Dequantize}(\Delta W_q^t)$
- 14:  $\Delta W_G^t = \frac{1}{\sum_{i \in \mathcal{N}} D_i} \sum_{i=1}^N D_i \Delta W_i^t$   $\triangleright$  (Aggregation through average)
- 15: **Updater**():
- 16:
- 17: **for** each iteration  $t$  from 1 to  $T$  **do**
- 18:     Randomly choose a subset  $\mathcal{S}_t$  of  $m$  clients from  $\mathcal{N}$
- 19:     **for** each client  $i \in \mathcal{S}_t$  **parallelly do**
- 20:          $\Delta W_i^{t+1} \leftarrow \text{LocalTraining}(i, \Delta W_G^t)$
- 21:     **end for**
- 22: **end for**
- 23: **Gradient Quantization:**  $\Delta W_{qG}^t \leftarrow \text{Quantize}(\Delta W_G^t)$
- 24: [Step 4](Client)
- 25: **Gradient Dequantization:**  $\Delta W_G^t \leftarrow \text{Dequantize}(\Delta W_{qG}^t)$
- 26: Repeat from step 2.

---

### 1. Executed at the client level

- *Local training and update transmission:* Each client at their side uses their own data to learn parameters trained on the received global model during the first training round. The client tries to minimize the loss function [19]  $L(\Delta W_k^t)$  and searches for optimal hyperparameters  $\Delta W_k^t$ .

$$\Delta W_k^{t*} = \Delta W_k^t \arg \min L(\Delta W_k^t) \quad (1)$$

As soon as the training is completed at the clients side, the model gradients are quantized to  $\Delta W_{qG}^t$ , where  $t$  stands for each iteration index, using mixed-precision algorithm and transmitted to the server. These step prevents the framework from the gradient leakage attack. Even if the gradients are leaked, The permutations required to retrieve the raw data from quantized gradients make the process very arduous for the attackers.

### 2. Executed at the server side

- *Weight initialization:* The global model  $\mathbf{w}_G^0$  and its hyperparameters are disseminated from the server side. In our case, the application we chose is image classification and LeNet-5 model is initiated with randomized weight during the first training round. As soon as the first round finishes, the aggregated gradients achieved from the clients are continued.
- *Aggregation and global update:* The server first dequantizes the quantized gradients sent from the participants of each training round. The pseudo code for the dequantization process is given in 2. As soon as the dequantization is done, the aggregation process is carried out to get the aggregated update

$\Delta W_G^{t+1}$  from the local model gradients of the participants. The server wants to minimize the global loss function [19]  $L(\Delta W_G^t)$ , i.e.

$$L(\Delta W_G^t) = \frac{1}{N} \sum_{k=1}^N L(\Delta W_k^t) \quad (2)$$

Before sending the updated gradients back to the data owners, the updates are quantized to  $\Delta W_{qG}^{t+1}$  using the algorithm described in 2. This process is repeated until the global loss function converges or a desirable training accuracy is achieved. The Global Updater function runs on the SGD [20] formula for weight update. The formal equation of global loss minimization formula by the averaging aggregation at the  $t^{th}$  iteration is given below:

$$\Delta W_G^t = \frac{1}{\sum_{k \in \mathcal{N}} D_k} \sum_{k=1}^N D_k \Delta W_i^t \quad (3)$$

---

### Algorithm 2 Quantization and Dequantization

---

**Require:**  $min\_T$  (min value of target tensor),  $max\_T$  (max value of target tensor),  $max\_float$  (max value of input tensor of type float32)

```

1: [Quantization]
2: if ( $min\_T \times min\_range > 0$ ) then
3:    $min\_scale\_factor = min\_T / min\_range$ 
4: else
5:    $min\_scale\_factor = max\_float$ 
6: end if
7: if ( $max\_T \times max\_range > 0$ ) then
8:    $max\_scale\_factor = max\_T / max\_range$ 
9: else
10:   $max\_scale\_factor = max\_float$ 
11: end if
12:  $scale\_factor = \min(min\_scale\_factor, max\_scale\_factor)$ 
13:  $min\_range = min\_T / scale\_factor$ 
14:  $max\_range = max\_T / scale\_factor$ 
15: Quantized_output =  $\text{round}(\min(max\_range, \max(min\_range, input)) \times scale\_factor)$ 
16: [Dequantization]
17: if ( $\min(T) == 0$ ) then
18:    $scale\_factor = max\_range / max\_T$ 
19: else
20:    $scale\_factor = \max(min\_range / min\_T, max\_range / max\_T)$ 
21: end if
22: Dequantized_output =  $\text{Quantized\_output} * scale\_factor$ 

```

---

### 3. Quantization and Dequantization

In the method of quantization that we have outlined, scaling factor determination is the key factor during the quantization process. Since we have implemented mixed-precision quantization, the bit size that is used for each layer has a different range of options available to it. The scaling factor is set to have the highest feasible value in order to make sure that all of the values that fall within the range extending from the minimum range to the maximum range are able to be represented by values of our selected data type of output tensor. After that, the scale factor is utilized to make the modifications to the minimum and maximum values. As soon as these steps are completed, quantized version of the input tensor can be obtained by clipping the values to the minimum range and maximum range and then multiplying by the scaling factor.

The dequantization process uses the same set of parameters to select the maximum and minimum range values and looks for the maximum value between the two ranges if the quantized tensor value is not zero. The complete pseudocode for this approach can be found in algorithm 2.

## 4 Experiment and Result Analysis

Server-clients based federated learning framework is implemented on Keras, which is built on TensorFlow backend. The server sets the pace of the training, determines the number of epochs per round, and how many rounds of overall

training are to be conducted. In mixed-precision quantization, the gradients of different layers of deep model are quantized with different precision and quantization bits. We utilized the combination of int8 and int16 bit quantization keeping the accuracy in the loop. We conducted the proposed federated training framework with 15 clients on MNIST [21], fashion mnist[22], CIFAR-10 [23] dataset.

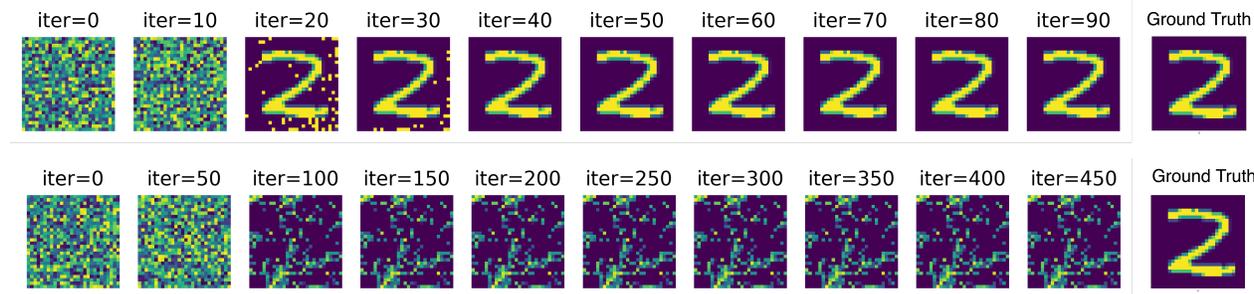


Figure 1: Gradient leakage attack. Training sample retrieved within 30 iterations (above), no leakage from gradients after mixed precision applied (below).

At first, we launched the gradient leakage attack by extracting the training samples from the gradients. To launch the attack, we attempted to match the gradients generated by the dummy data and the real ones. The difference between the dummy and real data is decreased by minimizing the distance between gradients. Figure 1 and 2 depicted the gradient leaking process that how training samples could be retrieved from gradients. And we found out that recovering monochromatic images with a clean background (MNIST) is easier, whereas recovering relatively complex images (CIFAR-10) requires more iterations.

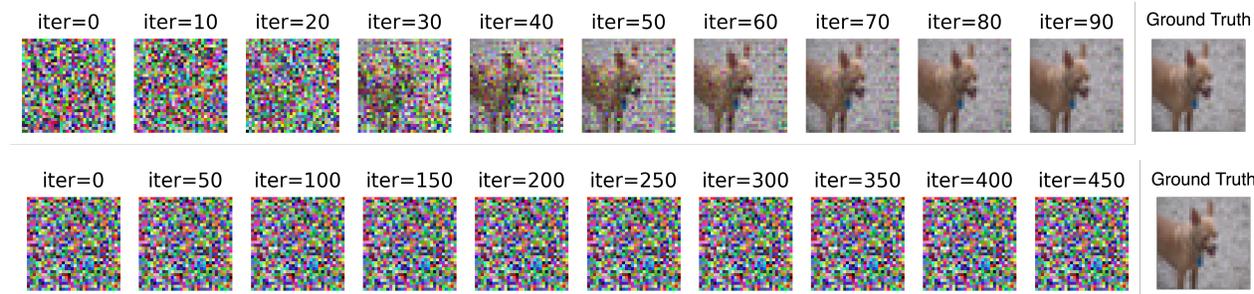


Figure 2: Gradient leakage attack. Training sample retrieved within 40 iterations (above), no leakage from gradients after mixed precision applied (below).

Secondly, we showed the effectiveness of our proposed mixed precision quantized approach against the gradient leakage attack. From figure 1 and 2, we can see that training images could not be retrieved from the quantized gradients even after 450 iterations of distance minimization, whereas we retrieved both mnist and cifar-10 samples within 40 iterations from unquantized gradients (figure 1 and figure 2).

Table 1: Top-1 accuracy comparison table

Methods	Dataset		
	Mnist	Fashion Mnist	Cifar10
Base FL (vulnerable to attack)	97.05	86.8	60.04
FL with DP (epsilon=1,delta=1e-5)	89.94	79.49	38.39
Mixed Precision and Quantization (Proposed FL)	96.67	85.22	58.9

Moreover, we compared the performance efficiency of our proposed approach with differential privacy based FL. We illustrated the gaussian differential privacy approach with epsilon=1 and delta=1e - 5 as optimum level of noise to provide data privacy. In comparison with base FL (model with no defense against gradient leakage attack), our proposed approach achieved almost the same level of performance for mnist dataset and 1.5% degradation for fashion mnist dataset, whereas DP FL degrades around 7% on performance for both mnist and fashion mnist. Similarly, DP FL has 21% accuracy drop on cifar-10, whereas our approach has only 1% drop, demonstrated in table 1.

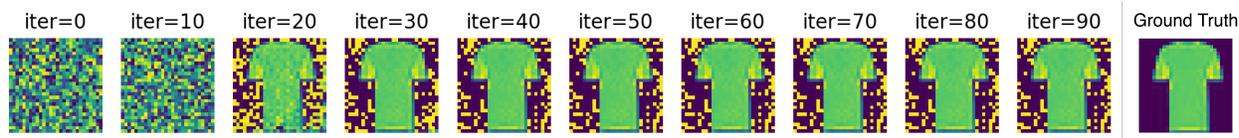


Figure 3: Recovered image from dequantized gradients with negligible noise in the background.

While ensuring the defense against gradient leakage attacks, the proposed approach has almost same level performance compared to base FL. Since we dequantized the quantized gradients at the server side prior to update global model. Since dequantization is not a fully reversible process, the recovered image from dequantized gradients has a few negligible noises in the background pixels (figure 3) but this process does not necessarily impact the model’s performance.

## 5 Ablation Study

In this section, we analyze our framework from two different points of view. Firstly, we provide our findings on the impact of specific quantization modes. While implementing our quantized framework, we employ a hyperparameter named ‘Quantization mode’. The content determines which calculation procedure will be used to determine the modified maximum and minimum data range. The quantization mode hyperparameter is varied across layers keeping accuracy in the loop. In the server side, the mode that was used for quantization must be selected for dequantization to get the ground truth gradients. Otherwise, dequantized gradients will have a significant mismatch that will largely impact the performance. To showcase the mismatch, we illustrated in figure 4 that ground truth image cannot be recovered even from dequantized gradients if generated by a different mode.

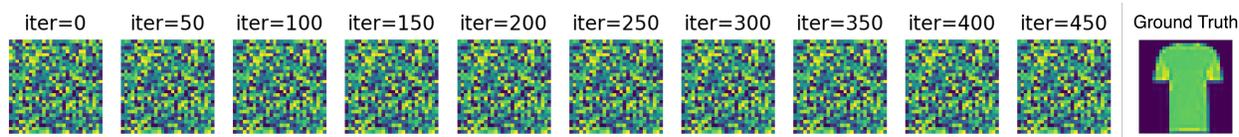


Figure 4: Failure in image recovery when dequantized mode mismatched with quantized mode

Secondly, the mixed-precision quantization ensures robustness against the attack where an attacker aims to reveal the actual gradient by dequantize the quantized weights. The set of operations that is needed to quantize, the same set of reverse operations is needed to dequantize it. If we assume that the attacker somehow reveals the min max range of float32 bit actual gradients, the attacker may dequantize it by trying all  $m$  combinations for single bit quantization where  $m$  is the number of quantization mode. On the other hand, for mixed bit quantization, the attacker will need to try  $m^L$  combinations to crack it which is a large number of combinations, where  $L$  is the number of layers in any deep model.

## 6 Discussion

Our proposed approach can also be seen as a communication efficient federated framework. In our approach, the low bit quantized gradients are being shared between server and clients rather than transmitting the gradients of float32. For instance, converting the precision of activation and gradients from 32-bit floats (174 kB) to 8-bit integers (44 kB) results in  $4\times$  data reduction, which eventually requires 4 times less transmission bandwidth. Quantized gradient reduces the downstream and upstream communication cost and thus speeds up training.

## 7 Conclusion

The use of federated learning is motivated by data privacy and communication efficiency. Our proposed mixed precision quantized FL ensures data privacy and low communication cost. Experiments on three benchmark datasets demonstrate that our approach outperforms the differential privacy approach in terms of accuracy and is better than the encryption process in terms of communication cost and computational overheads. To conclude, the proposed federated framework has high resilience against gradient privacy leakage attacks with competitive performance accuracy and a strong data privacy guarantee.

## References

- [1] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.
- [2] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.
- [3] Wenqi Wei, Ling Liu, Yanzhao Wut, Gong Su, and Arun Iyengar. Gradient-leakage resilient federated learning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pages 797–807. IEEE, 2021.
- [4] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning. *arXiv preprint arXiv:2009.03561*, 2020.
- [5] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2017.
- [6] Haokun Fang and Quan Qian. Privacy preserving machine learning with homomorphic encryption and federated learning. *Future Internet*, 13(4):94, 2021.
- [7] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.
- [8] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 691–706. IEEE, 2019.
- [9] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating client privacy leakages in federated learning. In *European Symposium on Research in Computer Security*, pages 545–566. Springer, 2020.
- [10] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems*, 33:16937–16947, 2020.
- [11] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.
- [12] Yong Li, Yipeng Zhou, Alireza Jolfaei, Dongjin Yu, Gaochao Xu, and Xi Zheng. Privacy-preserving federated learning framework based on chained secure multiparty computing. *IEEE Internet of Things Journal*, 8(8):6178–6186, 2020.
- [13] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *2013 IEEE global conference on signal and information processing*, pages 245–248. IEEE, 2013.
- [14] Aleksei Triastcyn and Boi Faltings. Federated learning with bayesian differential privacy. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2587–2596. IEEE, 2019.
- [15] Jun Liu, Yuan Tian, Yu Zhou, Yang Xiao, and Nirwan Ansari. Privacy preserving distributed data mining based on secure multi-party computation. *Computer Communications*, 153:208–216, 2020.
- [16] Mohammad Saidur Rahman, Ibrahim Khalil, Mohammed Atiquzzaman, and Xun Yi. Towards privacy preserving ai based composition framework in edge networks using fully homomorphic encryption. *Engineering Applications of Artificial Intelligence*, 94:103737, 2020.
- [17] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4):1–35, 2018.
- [18] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.
- [19] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.
- [20] Zengpeng Li, Vishal Sharma, and Saraju P Mohanty. Preserving data privacy via federated learning: Challenges and solutions. *IEEE Consumer Electronics Magazine*, 9(3):8–16, 2020.
- [21] Yann LeCun. The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>, 1998.
- [22] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.