MIA-BAD: An Approach for Enhancing Membership Inference Attack and its Mitigation with Federated Learning

Soumya Banerjee*, Sandip Roy*, Sayyed Farid Ahamed*, Devin Quinn†, Marc Vucovich†,
Dhruv Nandakumar†, Kevin Choi†, Abdul Rahman†, Edward Bowen†, Sachin Shetty*
*Virginia Modeling, Analysis and Simulation Center, Old Dominion University, Virginia, USA
{s1banerj, sroy, saham001, sshetty}@odu.edu
†Deloitte & Touche LLP
{devquinn, mvucovich, dnandakumar,kevchoi, abdulrahman, edbowen}@deloitte.com

***Abstract*—The membership inference attack (MIA) is a popular paradigm for compromising the privacy of a machine learning (ML) model. MIA exploits the natural inclination of ML models to overfit upon the training data. MIAs are trained to distinguish between training and testing prediction confidence to infer membership information. Federated Learning (FL) is a privacy-preserving ML paradigm that enables multiple clients to train a unified model without disclosing their private data. In this paper, we propose an enhanced Membership Inference Attack with the Batch-wise generated Attack Dataset (MIA-BAD), a modification to the MIA approach. We investigate that the MIA is more accurate when the attack dataset is generated batch-wise. This quantitatively decreases the attack dataset while qualitatively improving it. We show how training an ML model through FL, has some distinct advantages and investigate how the threat introduced with the proposed MIA-BAD approach can be mitigated with FL approaches. Finally, we demonstrate the qualitative effects of the proposed MIA-BAD methodology by conducting extensive experiments with various target datasets, variable numbers of federated clients, and training batch sizes.**

***Index Terms*—Federated Learning, Membership Inference Attack, Privacy, Security.**

I. Introduction

In recent years, federated learning (FL) has emerged as a popular privacy-preserving machine learning (ML) paradigm, allowing multiple clients to collaboratively develop a unified and consolidated model while protecting their individual training data [1]. Unlike typical centralized ML, FL does not require users to send their original data to the centralized server, which might compromise users' privacy and security. The basic procedure involves training localized models on individual client data, followed by the exchange of model updates among federated clients, and finally building a unified model that is shared by all clients [2]. Because collaborative learning doesn't involve sharing data, FL shows the potential to solve problems with data and user privacy that are quite prevalent in traditional centralized ML [3].

Despite the fact that FL is designed to secure personal information [4], recent research has shown that FL models are vulnerable to attacks that leak critical training dataset information through source inference, model inversion, and reconstruction attacks [5]. In this paper, we focus on membership inference attacks (MIAs), which are aimed against the FL model, and attempt to infer whether or not the target sample was included in the model's training data [6]. Successful MI attacks have the potential to compromise the security of federated clients [7]. As an example, knowing the target sample could reveal the patient's medical condition and treatment history, particularly when an FL model is trained on data sourced from diverse medical institutions [8]. An attacker can launch a membership inference attack in a "Black-Box" environment (Restrictive Knowledge of the model) by developing a binary attack model that, when fed the confidence score vector from the target model, returns the likelihood that the given data sample is part of the training dataset [6].

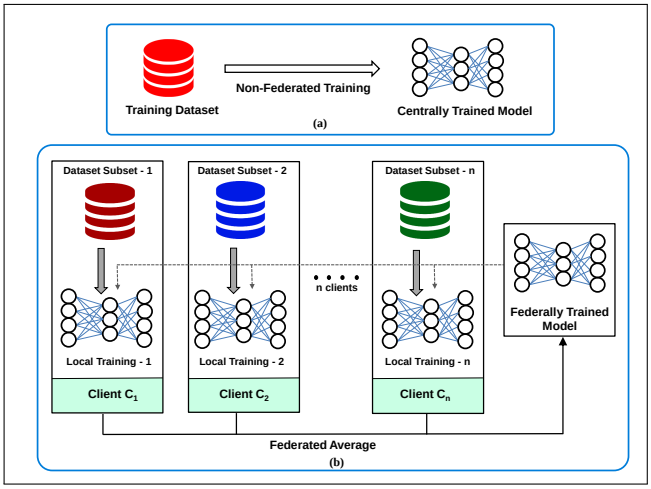

Fig. 1. Training procedure of an ML model, centrally vs FL. (a) Centrally or non-federated training of an ML model. (b) Training of an ML model in FL environment consisting of n clients.

Given a sample of data, MIA in FL detects its participation in the FL training process. A breach of privacy transpires when an adversary has any knowledge about the use of a certain data element for the purpose of training FL [6]. For example, if the data includes device information, we may deduce if the

device is participating in the FL training process, exposing the device's privacy [9].

In this paper, our aim is twofold. Initially, we show how an adversary can more accurately launch MIA attack into FL models for various datasets and then develop new insights for resisting such attacks in FL environments. We demonstrate that the FL paradigm while providing only limited protection to MIA, has a distinct advantage in reducing the potency of the proposed MIA-BAD approach.

The main contributions of this paper are as follows:

- We demonstrate how federated training can reduce the potency of membership inference attacks and how the number of federated clients affects this result.
- We propose a novel modification to the membership inference attacks paradigm, MIA-BAD, showing that the MIA is more accurate when the attack dataset is generated batch-wise.
- We demonstrate how the attacker's advantage of the proposed MIA-BAD can be minimized with FL. Through detailed experiments with different target datasets, and a varying number of clients and batch sizes, we document the the qualitative effects of the proposed MIA-BAD approach.

The paper is organized as follows. In section II, we introduce the relevant theoretical background and present a brief literature survey. Section III defines the threat model and the attacker's goals, knowledge, and capabilities. Section IV defines the framework for implementing the membership inference attack. We propose the MAI-BAD approach in section V. The experimental results are provided and analyzed in section VI. Finally, we conclude our work and discuss future research scope in section VII.

II. Preliminary

A. Federated Learning

FL is a decentralized ML training approach that allows multiple clients to collaboratively train a shared machine-learning model without accessing the local data of the clients [10]. A conventional FL system consists of n federated clients $(C_1, C_2, C_3, ..., C_n)$, in proximity with a central server denoted as S. The central server S is responsible for coordinating the FL training process and through iterative training generates a converged global FL model $\mathcal{M}$. The training process of the ML model $\mathcal{M}$ at the r^{th} training round (where r is an element of set R, representing the number of FL training rounds) is comprised of the following four steps:

- Step 1. The centralized server denoted as S, distributes the current global FL model, denoted as $\mathcal{M}^r$, to the federated clients participating in the process, denoted as C_i (where i ranges from 1 to n)
- Step 2. In parallel, each client C_i independently trains and improves the model $\mathcal{M}^r$, using its own dataset D_i. Once the local training process has been concluded, each client C_i transmits the updated model parameters, denoted by U_i^r, to the central server.
- Step 3. The central server S accumulates the updated parameters $U^r = [U_1^r, U_2^r, \cdots, U_n^r]$ from each of the clients that are participating in the process.
- Step 4. The central server S then updates the global model M^r by aggregating the collected parameter updates U^r. During the subsequent training iteration, the recently updated model M^{r+1} will be distributed to federated clients.

The training process in FL is executed iteratively by the central server and clients until a termination criterion is met. This criterion can be a maximum number of iterations or a threshold for model accuracy. Subsequently, the central server converges FL model $\mathcal{M}$, which is then distributed to each and every client of the system. Figure 1 contrasts the centralized and federated training of ML models.

In order to enhance privacy protection, recent advancements in the field of FL have incorporated various privacy protection mechanisms such as differential privacy[1] [11] and secure aggregation[2] [12]. Prior studies have primarily concentrated on two approaches for achieving differential privacy: centralized differential privacy, which relies on a central trusted party [11], [13], and local differential privacy, where each user perturbs their updates randomly before transmitting them to an untrusted aggregator [12], [14]. While FL has been recognized as a potentially effective approach that prioritizes privacy, a limited number of recent studies have demonstrated the susceptibility of FL to MIAs [7], [15].

B. Membership Inference Attack

Our work focuses on investigating the potential of membership privacy breaches through MIA and studying the attack's accuracy. In this section, we provide a brief overview of the background of existing works of MIAs as they pertain to ML models. An attacker intends to ascertain the membership property of a target sample x, i.e., whether or not this sample was used to train the target model, by exploiting the prediction behavior of an ML model $\mathcal{T}_m$ (referred to as the target model). MIA may compromise the privacy of training data used in ML models, thereby introducing additional risks for the producers of such training data.

The MIA can be expressed in a more formal way as:

$$\mathcal{A}(\mathcal{T}_m, \Omega, x) \rightarrow \textbf{In} \,/\, \textbf{Out} \tag{1}$$

where the attack is represented by $\mathcal{A}$, **In** means that x is a member, **Out** means x is not a member of training data $\mathcal{T}_m$. Ω refers to any additional knowledge about the target model and its training data that $\mathcal{A}$ can receive.

[1]**Differential privacy** guarantees that an individual's data is modified or perturbed in a controlled manner to provide a privacy guarantee, ensuring that the overall statistical outcomes of computations are stable and independent of whether their specific information is included or altered.

[2]**Secure aggregation** ensures that data collected from multiple sources is combined using cryptographic techniques to maintain individual data privacy and confidentiality, enabling collaborative computations without revealing the raw data.

In 2017, Shokri et al. [6] introduced an MIA algorithm for ML models, that assumes that the adversary has "Restrictive Knowledge" access to the model[3] and similar sample data as a priori knowledge. The experiments showed that the adversary can obtain significant information about the training data by attacking the model. In an FL environment, Truex et al. [16] gave insights that MIA may also result in the exposure of private membership information. Salem et al. [17] showed that the criteria for the execution of an MIA may be relaxed, which allows the attack to be carried out in a wider range of settings. Xu et al. provided a method for conspiring attackers to purposefully alter training data in order to enhance or decrease the weight of a dimension in the aggregation model [18]. If the aggregation model's weights or parameters reflect a recognizable pattern that generates relevant signals, sensitive data may be exposed.

Synthesis: Recently, there have been a few efforts to deploy various MIA variants in both centralized and federated ML environments. However, the effect of attack accuracy on different batch sizes of training to build the attack model for different numbers of clients in the FL has not been well studied.

III. Threat Model

In this section, we discuss the basic threat model, where we explain the attacker's knowledge of the environment, the attacker's goal, and the capability [6], [16], [17].

Adversary's knowledge: We consider the challenging situation in which the adversary can only acquire restricted knowledge about the target model and knowledge about the distribution of the target dataset. The training goals and model architecture are shared by all FL participants. As a result, this level of attacker knowledge is realistic. However, the adversary is unable to get any knowledge on the global training process (centralized or FL), or the distribution of the training data among the clients (if applicable).

Attacker's Goals: For the ML model, the attacker guesses or infers data from the initial training set. Following training, the attacker develops an attack model that infers private data from query-level access to the target model. The attacker does not alter the parameters of the model and does not need any additional information [6].

Attacker's Capability: The attacker is presumed to be an honest but curious user with query access to the target model, but cannot access its internal weights and gradients. The attacker uses this query access to implement the membership inference attack.

IV. Framework Overview

A. Attack Overview

Our study is focused on a framework that is predicated on the notion that, given training vs. non-training data, MIA may recognize the difference in an ML model's behavior. The inference attacker employs a two-step strategy. To begin, a collection of shadow models with behaviors comparable to the target model is built and trained. Then, the shadow models are utilized to generate an attack dataset for training the attack model. The trained attack model can infer whether a sample belongs to the training set of the target model data.

When the FL training procedure is completed, a global model is produced. This global model comprises the training outcomes of all participants' data that comes from many rounds of aggregation. To the adversary, this federally trained model is indistinguishable from a centrally trained model and thus makes no difference for MIA purposes.

The attacker trains the shadow models locally, each with behavior comparable to the global model. Having trained on the shadow model outcomes, the attack models may effectively infer the global model's "in" and "out" dataset.

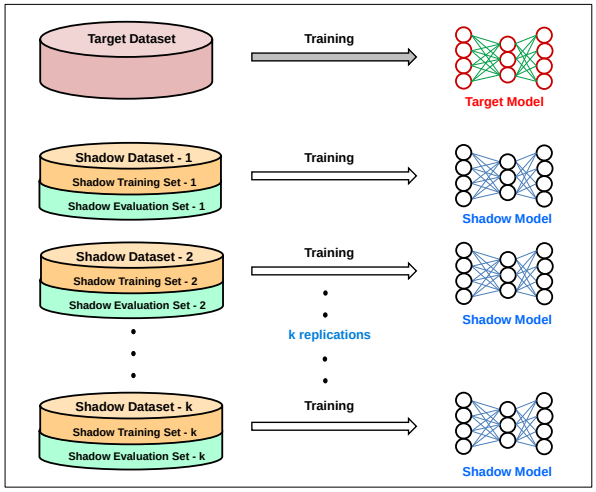

Fig. 2. MIA's shadow model training technique. Shadow models are trained in the same manner as the target model using k shadow-training and shadow-evaluation datasets.

B. Shadow Model

The concept of shadow model training involves using identically distributed data on the as similar as possible model architecture to train models that evaluate data similarly [6]. The attacker utilizes knowledge about the target dataset's distribution to build multiple shadow datasets with similar styles and distributions. The attacker trains shadow models, with a similar architecture to the target model, on these shadow datasets. Once trained, the shadow model is used to produce training data for the attack model. Figure 2 summarizes how the shadow models are trained.

C. Attack Model

The attack model is a binary classifier that is trained to predict if a given record is part of the model's training set or not. For each record in the shadow model's training dataset, the adversary creates a query, retrieves the output, marks the resultant vectors as "in" and finally adds them to the attack

[3]**Restrictive Knowledge** is achieved by an adversary for a black-box access or query access. It implies that the attacker can query the ML model to get a prediction on a given sample, but has no other access to the ML model and its weights.

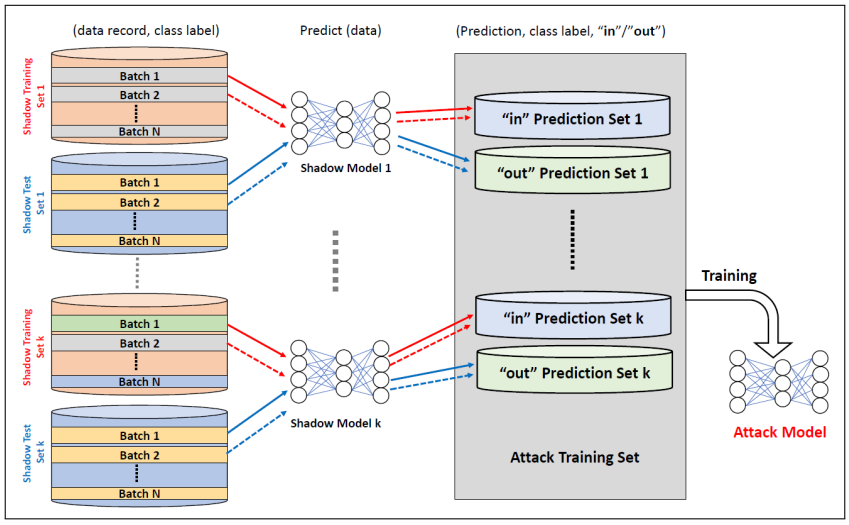

Fig. 3. Training process of the proposed MIA-BAD approach. The attack model is trained from batch-wise generation of the attack dataset.

model's training dataset. Similarly, the attacker queries and records output labeled as "out" using a disjoint test dataset of the shadow model. The attacker now possesses a dataset comprising records, as well as the appropriate outputs of the shadow models and the corresponding in/out labels. The objective of the attack model is to infer the labels from the records and corresponding outputs.

V. The MIA-BAD Approach

In this section, we investigate the effect of modifications on how the attack model is trained. Normally, as described in section IV-C, the attack dataset is built by evaluating on the shadow model to enable tabulation of the sample-wise loss in the attack dataset. This approach produces a dataset equal to the size of all the shadow datasets combined. This gives us more than enough data to train the attack model.

However, it is a well-known phenomenon that ensembling improves performance [19]. We generally train ML models in batches because larger batch sizes can provide better model convergence [20]. In an MIA, the shadow models as well as the attack models are trained in batches, however, the attack dataset is built sample-wise to retain the size of the attack dataset.

We propose, building the attack dataset by evaluating batch-wise with shadow models on the corresponding shadow dataset. This will significantly decrease the size of the attack dataset. Nevertheless, we hypothesized that the qualitative improvement through the implied ensembling would compensate for the quantitative loss, section VI-C backs our hypothesis. Figure 3 demonstrates the MIA-BAD approach for attack model training.

A. The Overall Attack Paradigm

In this section, we first present the basic steps of the proposed MIA-BAD approach. Next, then provide the algorithm (in pseudo-code format), in Algorithm 1, for implementing the proposed attack.

1) Define a master shadow dataset by sampling from a similar distribution as the target dataset.
2) Sample k overlapping but significantly different shadow dataset from the master shadow dataset. Split each shadow dataset into training and test subsets.
3) Train the k shadow models on the training subsets of its corresponding shadow dataset.
4) For every shadow model, evaluate the entire shadow dataset (train or test, represented by seen and unseen) batch-wise, and record the batch-wise loss with the seen/unseen label to build the attack dataset.
5) Train a binary classifier on the attack dataset as the attack model.

Algorithm 1 Algorithm for MIA-BAD

Input: Target dataset distribution Ω, Shadow model definition $\mathcal{S}$, Shadow count k, Batch size B
Output: Attack model $\mathcal{A}$

Set, $\mathcal{D}$ = [] ▷ *Attack Dataset.*
Initialize, $\mathcal{A} \leftarrow$ Binary Classifier.
for $i = 1$ to k **do**
  Initialize, $\mathcal{S}_i \leftarrow \mathcal{S}$.
  Sample, $\omega_i \leftarrow \Omega$.
  Train $\mathcal{S}_i$ with ω_i^{train}
  for batch $b \in \omega_i$ **do**
    Set, size of b as B.
    Evaluate $y \leftarrow \mathrm{S}_i(b)$
    if $b \in \omega_i^{train}$ **then**
      Update $\mathcal{D} \leftarrow \langle y, in \rangle$
    else if $b \in \omega_i^{test}$ **then**
      Update $\mathcal{D} \leftarrow \langle y, out \rangle$
    end if
  end for
end for
Train $\mathcal{A}$ with $\mathcal{D}$
return $\mathcal{A}$

TABLE I
PERFORMANCE OF MIA-BAD ON CIFAR10 DATASET FOR DIFFERENT BATCH SIZES.

Training Mode	Sample wise	Batch wise					
		8	16	32	64	128	258
Centrally	0.833	0.853	0.869	0.856	0.853	0.869	0.856
2 clients	0.721	0.751	0.755	0.757	0.753	0.753	0.751
5 clients	0.769	0.769	0.772	0.768	0.771	0.768	0.769
10 clients	0.786	0.765	0.765	0.796	0.753	0.769	0.786

VI. PERFORMANCE EVALUATION

In this section, we first define our experimental setup, then we demonstrate the effect of MIA on federally trained ML as opposed to a centrally trained model. Finally, we demonstrate the effect of batch-wise generation of the attack dataset.

A. Experimental Setup

We define a centralized FL architecture that trains ML models over a wide range of datasets. The architecture distributes the datasets into n disjoint subsets and sends them to n clients. Each client trains a local copy of the model architecture on the locally available data. After e epochs of local training the clients (securely) share the model weights with the orchestrating server which combines all the received model weights through a federated average [**?**] to define the global model, which is sent back to the clients. For the next round, the client completes e epochs of local training on the received model and shares the updated weights with the server. The server conducts r such rounds to arrive at the global model. For a baseline comparison, an ML model with the same architecture is trained for $e \times r$ epochs.

For this work, we evaluated the efficientnet [21] architecture on the CIFAR10 [22], CIFAR100 [22], MNIST [23], and FashionMNIST [24] datasets.

B. Membership Inference Attack on FL

In this section, we present how successful membership inference attack is on ML models trained centrally or in a federated manner over 2, 5, and 10 clients. Figure 4 summarizes the accuracy of the MIA over the trained ML models.

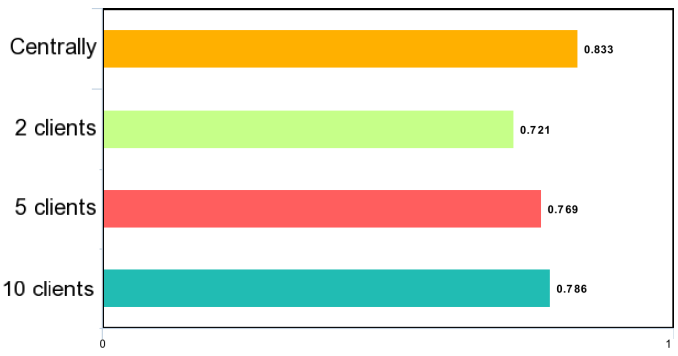

Fig. 4. Effectiveness of MIA in FL environment on CIFAR10 dataset for different number of clients.

When the model is trained centrally, we achieve almost 83.3% attack accuracy. In contrast, for a federally trained model, we get a maximum accuracy of 78.6% for ten clients. Therefore, when the model is trained federally, the attack is less effective. However, as the number of contributing clients increases this effect becomes less pronounced. Another observation, elaborated upon in the next section, training the model over federated architecture can mitigate the attacker's advantage gained by the proposed MIA-BAD approach.

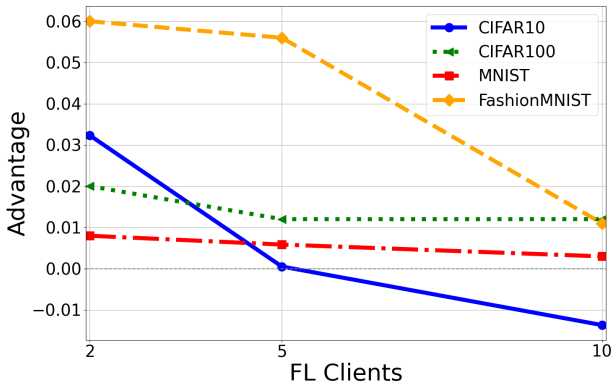

Fig. 5. Comparison of the advantage of the MIA-BAD approach for 2, 5, and 10 clients as performed using CIFAR10, CIFAR100, MNIST, and FashionMNIST datasets.

C. Effect of Generating the Attack Dataset Batch-wise

In this section, we highlight the experimental results of the proposed MIA-BAD approach. Table I shows the effect of generating the attack dataset batch-wise when the target dataset is CIFAR10. From table I we observe that the advantage of the MIA-BAD approach is not affected by the batch size. Table II demonstrates the same result for MNIST, FashionMnist, and CIFAR100 respectively.

Table I demonstrates how batch-wise generation of the attack dataset improves the attacker's advantage. We also observe that this effect is mostly independent of the batch size itself. Therefore, we only consider batch size 32 for the remaining datasets. We hypothesize that the marginal gain over the greater ensemble effect is perfectly countered by the reduced (attack) dataset size.

D. Effect of FL on the Proposed Approach

From tables I and II, we observe that the attacker's advantage over the MIA-BAD approach can be strongly countered

TABLE II
COMPARISON OF ACCURACY OF MIA-BAD ON MNIST, FASHIONMNIST, AND CIFAR100 DATASETS.

Training Mode	MNIST		FashionMNIST		CIFAR100	
	Sample	Batch	Sample	Batch	Sample	Batch
2 clients	0.838	0.846	0.747	0.807	0.701	0.721
5 clients	0.840	0.847	0.749	0.805	0.757	0.769
10 clients	0.849	0.852	0.787	0.798	0.786	0.798

by federated training of the ML model. The attacker's advantage is determined by the difference between batch-wise and sample-wise accuracy. In the case of CIFAR10, we consider the average batch-wise accuracy, for the remaining we consider batch size 32. Empirically, we observe the number of federated clients is inversely proportional to the effect of the MIA-BAD approach. Figure 5 demonstrates how the attacker's advantage through MIA-BAD can be mitigated through the federated training of ML models. In FashionMNIST, we observe the greatest advantage with the MIA-BAD approach, but nevertheless, it is mitigated with a high client count. Furthermore, with 10 clients we observed a negative advantage in the case of CIFAR10, completely mitigating the MIA-BAD advantage.

VII. CONCLUSION AND FUTURE SCOPE

In this paper, we propose a membership inference attack approach, MIA-BAD. We demonstrate that batch-wise attack dataset generation can provide an advantage to the adversary. We also demonstrate how the FL paradigm can be utilized to mitigate this approach. The proposed phenomenon is novel and promising and merits further investigation. In future works, we plan to investigate how the observed trends generalize over different ML models and more diverse datasets. We also plan to investigate if the proposed approach can be utilized with more advanced variants of the membership attacks.

ACKNOWLEDGEMENT

This work is partially funded by the Deloitte AI Center of Excellence.

REFERENCES

[1] H. McMahan, E. Moore, D. Ramage, and B. Arcas, "Federated learning of deep networks using model averaging. arxiv 2016," *arXiv preprint arXiv:1602.05629*.

[2] M. Vucovich, A. Tarcar, P. Rebelo, N. Gade, R. Porwal, A. Rahman, C. Redino, K. Choi, D. Nandakumar, R. Schiller *et al.*, "Anomaly detection via federated learning," *arXiv preprint arXiv:2210.06614*, 2022.

[3] J. Chen, X. Zhang, R. Zhang, C. Wang, and L. Liu, "De-pois: An attack-agnostic defense against data poisoning attacks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3412–3425, 2021.

[4] C. Ma, J. Li, M. Ding, H. H. Yang, F. Shu, T. Q. Quek, and H. V. Poor, "On safeguarding privacy and security in the framework of federated learning," *IEEE network*, vol. 34, no. 4, pp. 242–248, 2020.

[5] H. Hu, Z. Salcic, L. Sun, G. Dobbie, and X. Zhang, "Source inference attacks in federated learning," in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 1102–1107.

[6] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[7] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 739–753.

[8] M. G. Crowson, D. Moukheiber, A. R. Arévalo, B. D. Lam, S. Mantena, A. Rana, D. Goss, D. W. Bates, and L. A. Celi, "A systematic review of federated learning applications for biomedical data," *PLOS Digital Health*, vol. 1, no. 5, p. e0000033, 2022.

[9] M. Ali, F. Naeem, M. Tariq, and G. Kaddoum, "Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey," *IEEE journal of biomedical and health informatics*, vol. 27, no. 2, pp. 778–789, 2022.

[10] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–36, 2021.

[11] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[12] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM workshop on artificial intelligence and security*, 2019, pp. 1–11.

[13] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," *arXiv preprint arXiv:1710.06963*, 2017.

[14] L. Sun, J. Qian, and X. Chen, "Ldp-fl: Practical private aggregation in federated learning with local differential privacy," *arXiv preprint arXiv:2007.15789*, 2020.

[15] H. Lee, J. Kim, S. Ahn, R. Hussain, S. Cho, and J. Son, "Digestive neural networks: A novel defense strategy against inference attacks in federated learning," *computers & security*, vol. 109, p. 102378, 2021.

[16] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei, "Demystifying membership inference attacks in machine learning as a service," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 2073–2089, 2019.

[17] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," *arXiv preprint arXiv:1806.01246*, 2018.

[18] X. Xu, J. Wu, M. Yang, T. Luo, X. Duan, W. Li, Y. Wu, and B. Wu, "Information leakage by model weights on federated learning," in *Proceedings of the 2020 workshop on privacy-preserving machine learning in practice*, 2020, pp. 31–36.

[19] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*. Springer, 2000, pp. 1–15.

[20] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," *arXiv preprint arXiv:1609.04836*, 2016.

[21] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[22] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[23] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[24] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.