Robust and Privacy-Preserving Collaborative Learning: A Comprehensive Survey

Shangwei Guo*, Xu Zhang*, Fei Yang†§, Tianwei Zhang‡, Yan Gan*, Tao Xiang*, and Yang Liu‡
*College of Computer Science, Chongqing University, Chongqing, China
†Zhejiang Lab, Hangzhou, China
‡School of Computer Science and Engineering, Nanyang Technological University, Singapore

***Abstract*—With the rapid demand of data and computational resources in deep learning systems, a growing number of algorithms to utilize collaborative machine learning techniques, for example, federated learning, to train a shared deep model across multiple participants. It could effectively take advantage of resource of each participant and obtain a more powerful learning system. However, integrity and privacy threats in such systems have greatly obstructed the applications of collaborative learning. And a large amount of works have been proposed to maintain the model integrity and mitigate the privacy leakage of training data during the training phase for different collaborate learning systems. Compared with existing surveys that mainly focus on one specific collaborate learning system, this survey aims to provide a systematic and comprehensive review of security and privacy researches in collaborative learning. Our survey first provides the system overview of collaborative learning, followed by an brief introduction of integrity and privacy threats. In an organized way, we then detail the existing integrity and privacy attacks as well as their defenses. We also list some open problems in this area and opensource the related papers on GitHub: https://github.com/csl-cqu/awesome-secure-collebrative-learning-papers.**

***Index Terms*—Collaborative Learning, Byzantine, Backdoor, Privacy-Preserving**

I. Introduction

Deep learning (DL) has demonstrated its tremendous success in multiple fields including computer vision, natural language processing, bioinformatics, and board game programs. DL systems adopt deep neural networks (DNNs) to improve automatically through experience on huge training datasets [1]–[4]. To efficiently train a DL model, a learning system mainly relies on two components: a large number of high-quality training samples and high-performance GPUs. However, the training datasets and GPUs may be distributed at different parties due to various reasons. Consider the following two examples [5]–[7]:

Medical Image Classification. *A hospital wants to learn a lung cancer detector model to assist its doctors in identifying lung cancer patients from their computed tomography (CT) images. Because the hospital has only received a limited number of lung cancer patients, learning a highly accurate model is difficult for it. To guarantee the accuracy of the diagnosis, the hospital unites with other hospitals to collaboratively learn a shared model together. All hospitals need to keep their CT images locally by considering the privacy of patients.*

§Fei Yang is the corresponding author (email: yangf@zhejianglab.com).

Mobile Keyboard Prediction. *As users increasingly shift to mobile devices, Gboard, the Google keyboard, indents to provide reliable and fast mobile input methods such as next-word predictions. Although publicly available datasets can be used for such task, the distribution of such datasets often does not match that of users. Thus, Gboard requires user-generated texts for better performance without causing users to be uncomfortable with the collection and remote storage of their personal data.*

Collaborative learning has recently been popular as a promising solution for such application scenarios [8]–[14]. Specifically, collaborative learning allows two or more participants to collaboratively train a shared global DL model while keeping their training datasets locally. Each participant trains the shared model on his own training data and exchanges and updates model parameters with other participants. Collaborative learning can improve the training speed and the performance of the shared model while protecting privacy of the participants' training datasets. Thus, it is a promising technique for the scenarios where the training data is sensitive (e.g., medical records, personally identifiable information, etc.). Several learning architectures have been proposed for collaborative learning: with and without a central server, with different ways of aggregating models, etc [15]–[22]. An important branch of collaborative learning is federated learning [23] that enables mobile phones to collaboratively learn a shared prediction model while keeping all the training data on device, decoupling the ability to do machine learning from the need to store the data in the cloud.

Although each participant stores his training dataset locally and only shares the updates of the global model at each iteration, adversaries can also conduct attacks to break model integrity and data privacy during the training process [24]–[27]. One of the most severe threats is the model integrity that can be easily compromised when some of participants are not trustworthy [28], [29]. For example, malicious participants poison their training datasets with some carefully crafted malicious triggers. Then, at each iteration, they generate malicious updates with the triggers and gradually inject such triggers as backdoors into the global model by sharing the malicious updates to earn extra profit or increase their advantages [30], [31]. Adversaries can also disguise as participants to join the collaborative learning process and destroy the learning process by sending malicious updates to their neighborhoods

or parameter servers [25], [32], [33]. Blanchard et al. [28] and Guo et al. [29] show that only one malicious participant can control the whole collaborative learning process.

Other than the model integrity threats, another crucial challenge is to protect the data privacy of each participant. It have been demonstrated that although participants do not share the raw training samples with others, the shared updates are generated from the samples and also leak information about the training datasets indirectly. For instance, Melis et al. [34] found that one can capture the membership and unintended feature leakage from the shared gradients during the training process. More seriously, Zhu et al. [26] proposed an optimization method that can reconstruct training samples from the corresponding updates.

To address the above integrity and privacy threats, many methods are proposed to defend these attacks [24], [26], [28], [35]–[48], [48], [49], [49]–[66]. For instance, to achieve byzantine-resilient collaborative learning, Blanchard et al. [28] use statistic tools to inspect the updates of participants at each iteration and abandon potential malicious updates when aggregating updates. In terms of privacy protection, Gao et a. [67] proposed to search privacy-preserving transformation functions and pre-process the training samples with such functions to defend reconstruction attacks as well as preserving the accuracy of the trained DL models. Several defenses [68]–[72] also proposed robust and privacy-preserving defenses to defend both integrity and privacy threats.

Several survey works [27], [73]–[80] have also summarized some of the threats and defenses in collaborative learning. However, they have certain drawbacks. First, most of them only consider some specific branches of collaborative learning and lack systematic and comprehensive analysis towards other collaborative learning systems. . For example, Second, several surveys [73], [74], [79] mainly target on the threats and defenses in federated learning. Vepakommacite et al. vepakomma2018no summarize the privacy problems and defenses in distributed learning systems. Second, existing surveys do not focus on the training process of collaborative learning systems (the most important stage) and selectively introduce existing threats and defenses, which makes them unable to summarize state-of-the-art methods well.

In this paper, we focus on the integrity and privacy attacks and defenses during the training process of collaborative learning and present a comprehensive survey of the state-of-the-art solutions. Specifically, we systematically introduce different types of collaborative learning systems from various perspectives (Section II). Then, we summarize summarizes the privacy and integrity threats in collaborative learning in Section III. One the one hand, we exhibits existing attacks and the corresponding defenses in Section IV and V, respectively. On the other hand, we shows the state-of-the-art integrity privacy attacks and the corresponding defenses in V Section VI, respectively. We summarize hybrid defense methods to achieve robust and privacy-preserving collaborative learning and adversarial training algorithms to improve the robust of model inference. We illustrate some open problems and future solutions in collaborative learning in Section IX, followed by Section X that concludes this paper. We also opensource the paper list of the attack and defense methods on GitHub: https://github.com/csl-cqu/awesome-secure-collebrative-learning-papers.

II. System Overview

A. Machine Learning Model

In general, machine learning is categorized as supervised learning [81], unsupervised learning [82] and reinforcement learning [83], [84]. In this survey, machine learning is mostly referred to as supervised learning. Supervised learning is the process of optimizing a function from a set of labeled samples such that, given a sample, the function would calculate an approximation of the label.

We use a *dataset* $\mathcal{D}$ to denote a probability distribution of data; $z{\sim}\mathcal{D}$ denotes a random sampled variable z from $\mathcal{D}$, and $\mathbb{E}_{z\sim\mathcal{D}}[f(\xi)]$ denotes the expected value of $f(\xi)$ for a random variable ξ. For a deep learning model, we use $w \in \mathbb{R}^d$ be the d-dimensional parameter vector to estimate the model; $L_{\mathcal{D}}(f)$ be the loss calculated by f on dataset $\mathcal{D}$; l be the loss function of an individual sample. Therefore, we abstract the machine learning task by the following optimization problem:

$$w^* = \underset{w\in\mathbb{R}^d}{argmin} L_{\mathcal{D}}(f_w) = \underset{w\in\mathbb{R}^d}{argmin} \underset{\xi\sim\mathcal{D}}{\mathbb{E}}[l(w,\xi)] \ . \quad (1)$$

There are many different approaches [85] to minimize the loss function, such as gradient descent, second-order methods, evolutionary algorithms, etc. In machine learning, optimization is majored performed via gradient descent. By using randomly sampled data in each iteration, we can apply *Stochastic Gradient Descent* (SGD) [86] to optimize Eq. 1.

B. Dimensions of Parallelism

Machine learning is growing rapidly in the recent decade due to the growth of the sizes of models and datasets. Machine learning algorithms improved a lot thanks to the more and more complicated models, as well as larger and larger datasets. Therefore, parallelism is introduced to provide the scalability of machine learning algorithms. As illustrated in Fig. 1, parallel training allows users to partition the data and computation tasks onto multiple computational resources such as cores and devices. We introduce four prominent partitioning strategies, categorized by the dimension of parallelism, which are data parallelism, model parallelism, pipelining and hybrid parallelism.

1) Data Parallelism: For data parallelism [87], as shown in the top figure of Fig. 1 (a), the approach is to partition the samples from the dataset among multiple computational resources (cores or devices). This method is the dominant distributed deep neural network training strategy.

2) Model Parallelism: Data parallelism (the bottom figure of Fig. 1 (a)) sometimes suffers from very large models since the memory required to store parameters and activations and the time to synchronize parameters makes data parallelism impossible or inefficient. Model parallelism [88] is introduced to solve the above issue. The strategy of model parallelism is to split the model into multiple computational resources. It divides the computational tasks according to the neurons in

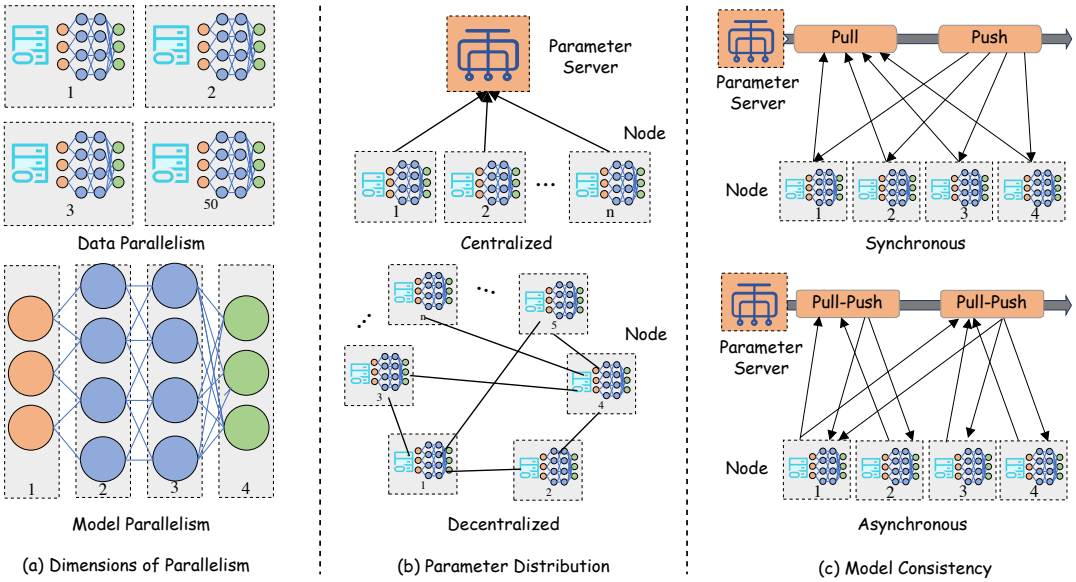

Fig. 1. System Overveiw

each layer. Moreover, the sample minibatch is copied to all processors and different parts of the model are computed on different processors.

3) Pipelining: Pipelining in machine learning can either refer to overlapping computations between layers or partitioning the DNN according to depth and assigning layers to specific processors. Therefore, pipelining is both a form of data parallelism as samples are processed by the network in parallel, and a form of model parallelism as models are partitioned by layers.

The pipelining strategy can be used to overlap the procedure of forward evaluation, backpropagation and weight updates. This approach minimizes the processor idle time. Meanwhile, pipelining could also be treated as partitioning the layers, each processor handles a fixed layer and the data flow is determined throughout the whole procedure.

4) Hybrid Parallelism: Hybrid parallelism combines multiple parallelism schemes. For instance, in AlexNet, a successful approach is to apply data parallelism to the convolutional layer where most computations are performed, and model parallelism to the fully connected layer where most of the parameters are stored.

C. Parameter Distribution

From now, we shall always refer to data parallelism in this paper unless otherwise specified, as it is the most widely used and mostly discussed parallelization method. And the communication manner along multiple devices is given in Fig. 1 (b), including centralized and decentralized.

1) Centralized: The classical topology of distributed learning is centralized. A typical centralized architecture is *Parameter Server* (PS) [15]. In a PS architecture, there exists a single or multiple nodes as masters, and multiple nodes as workers. Every worker node keeps a copy of the model and a part of the dataset. Within a training iteration, the master node distributes the weights of the model to the workers, then every worker node randomly samples a batch of data from its data partition and calculates the gradient of the weights upon the samples. Finally, all workers send their computing results to the master and the master updates the weights of the model according to the aggregated gradients before the training turns to the next iteration. We illustrate the centralized distributed learning in Algorithm 1.

Algorithm 1: Distributed subgradient descent.

Task Scheduler:
  issue `LoadData` () to all workers
  for *iteration t=0,...,T* **do**
    issue `WorkerIterate` (*(t)*) to all workers
Worker *r=1,...,*:
  Function `LoadData` *()*:
    load a part of training data $\{y_{ik}, x_{ik}\}_{k=1}^{n_r}$
    pull the working set $w_r^{(0)}$ from servers
  Function `WorkerIterate` *(t)*:
    gradient $g_r^{(t)} \leftarrow \sum_{k=1}^{n_r} \partial l(x_{ik}, y_{ik}, w_r^{(t)})$
    push $g_r^{(t)}$ to servers
    pull $g_r^{(t+1)}$ from servers
Servers:
  Function `ServerIterate` *(t)*:
    aggregate $g^{(t)} \leftarrow \sum_{r=1}^{m} g_r^{(t)}$
    $w^{(t+1)} \leftarrow w^{(t)} - \eta(g^{(t)} + \partial\Omega(w^{(t)}))$

2) Decentralized: Centralized distributed learning suffers from a communication bottleneck on the master node, therefore, the scalability of such an architecture is limited. Decentralized network topology is proposed to solve such a problem. Here, we categorise the mainstream decentralized approach to ring topology and general decentralized topology.

Ring topology is inspired by the ring all-reduce algorithm from network community, then it is used introduced to decentralized distributed learning to implement the all-

Algorithm 2: Decentralized parallel stochastic gradient descent on the i-th node

Input: initial point $x_{0,i} = x_0$, step length γ, weight matrix W, and the number of iterations K

for $t = 0, \ldots, K-1$ **do**
- Randomly sample $\xi_{t,i}$ from local data of the i-th node
- $\forall i$ Compute the local stochastic gradient $\partial l(x_{it}, y_{it}, w_r^{(t)})$
- Compute the neighborhood weighted average by fetching optimization variables from neighbors: $x_{t+\frac{1}{2},i} = \sum_{j=1}^{n} w_{ij} x_{t,j}$
- Update the local optimization variable $x_{t+1,i} \leftarrow x_{t+\frac{1}{2},i} - \gamma \partial l(x_{it}, y_{it}, w_r^{(t)})$.

Output: $\frac{1}{n}\sum_{i=1}^{n} x_{K,i}$

reduce operation by Baidu[1]. Later, Nvidia also successfully use ring all-reduce in its GPU collective communication library (NCCL)[2].

General decentralized topology can be illustrated using a weighted undirected graph (V, W), where $V = \{1, 2, \ldots, n\}$ denotes the set of nodes, and $W \in \mathbb{R}^{n \times n}$, satisfying $w_{i,j} \in [0, 1]$, $w_{ij} = w_{ji}$ and $\sum_j w_{ij} = 1$. The decentralized learning can be abstracted as an optimization problem that minimizes the average expectation of the loss function over all nodes, as follows:

$$\underset{w \in \mathbb{R}^d}{argmin} f(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\xi \sim \mathcal{D}_i} F_i(x; \xi) \ . \tag{2}$$

Decentralized parallel stochastic gradient descent (D-PSGD) [89] is the most widely utilized algorithm in decentralized distributed learning. We illustrate the D-PSGD in Algorithm 2.

D. Model Consistency

In collaborative learning, the goal is to train a single copy of model parameter w from multiple participants. However, there may be multiple instances of SGD running independently on different nodes, hence the model parameter is updated by different nodes simultaneously as shown in Fig. 1 (c). Therefore, some strategies are applied to ensure the consistency of the model.

1) Synchronous: A straightforward approach is to use a *synchronized* strategy to update the model. For every training iteration, all the participants synchronize their parameters. For instance, in Spark [90], a master node aggregates the parameters after all the working nodes finish their computation for one batch of samples. This strategy ensures a strong consistency of the model, however, it also leads to a low usage to the computational capacity since a node that finishes early has to wait until all the other nodes finish their computation.

[1]https://github.com/baidu-research/baidu-allreduce

[2]https://github.com/NVIDIA/nccl

2) Asynchronous: An *asynchronized* model updating strategy greatly enhances the usage of the computational resources. For instance, in Parameter Server [15], a working node pushes its result to the server and pulls the current parameter without waiting for other nodes. Consequently, the strategy avoids the waiting time of a node.

3) Stale-Synchronous: However, for large-scale and heterogeneous clusters, the model consistency for asynchronized strategy suffers from the *staleness* problem. In a heterogeneous environment, there exist some stale nodes with computing speed slower than others. Therefore, using asynchronized strategy sometimes aggregates gradients always from stale nodes with faster nodes, thus breaking the consistency of the model. In order to provide correctness guarantees in spite of asynchrony, Stale-Synchronous Parallelism (SSP) [91] proposes a compromise between consistent and inconsistent models. In SSP, a global synchronization step is forced before a bound of gradient staleness of one of the nodes is reached. This approach performs well in heterogeneous environments.

E. Federated Learning

Federated learning [23] is a rapidly growing research area in the past years. It is a machine learning technique that trains an algorithm across multiple centralized or decentralized edge devices or servers holding local data samples, without exchanging data. In federated learning, data are not supposed to be uploaded to servers, and local data samples are not assumed to be identically distributed. Federated learning enables multiple nodes to build a common, robust machine learning model without sharing data, thus it is addressed to critical issues such as data privacy, data security, data access rights and heterogeneous data.

Since federated learning inherits the architecture of collaborative learning, it inherits the security threats in collaborative learning naturally. We will also elaborate on the security threats, privacy issues, attack and defense methods for federated learning in later sections.

III. Threat in Collaborative Learning

Collaborative learning has shown remarkable achievements in many fields while it still faces serious security and privacy risks due to the complexity of the learning system and the untrustworthy of participants or parameter servers. These security problems are worse than standalone learning systems because underlying adversaries in thousands of participants are rougher to detect and defend. We classify existing threats during the training process into two categories according to the objective of adversaries: integrity and privacy threats.

A. Integrity Threats

Model integrity requires the accuracy and completeness of trained models, which presents the efforts to change or manipulate the models. It is the core requirement during the training and applying deep learning in practice. However, recent researches have shown that only a single malicious participant can influence or even control the whole model training process in collaborative learning scenarios [28], [29].

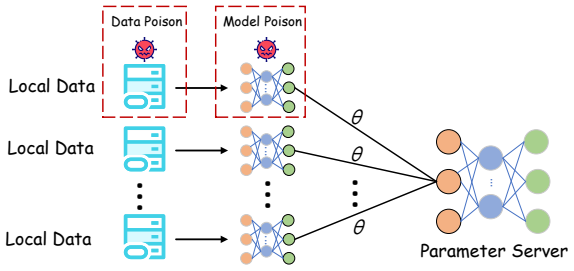

Fig. 2. Two types of attacks: data and model poisoning.

Compromise vs. Backdoor. Attacks for breaking the integrity of collaborative learning can be categorized as compromise and backdoor attacks according to the corresponding adversarial goals. Compromise attack aims to reduce or destroy the trained model performance by changing model parameters, which normally makes the shared model not converge to a satisfactory one during the training phase. It can also be caused by system problems, like system failures, network congestion and here we only focus on adversarial manipulations in the following sections. Such adversarial goals can be achieved by Byzantine attacks [25], [28], [33], [38], [92], where some participants inside the collaborative learning system can conduct inappropriate behaviors, and propagate wrong information, leading to the failure of the learning system.

On the other hand, backdoor attacks try to inject predefined malicious training samples, i.e., backdoors, into a victim model while maintaining the performance of the primary task [31], [93]–[102]. The backdoors would be activated if a input sample contains the injected triggers. Because of the secrecy of triggers, it is difficult to identify backdoor attacks as a backdoored model performs normally on normal samples.

Data Poisoning vs. Model Poisoning. Adversaries can attack the collaborative learning systems in two types: data and model poisoning. In data poisoning, attackers can poison the training datasets of some participants with malicious samples with carefully crafted triggers [53]. For instance, backdoor attacks for the image classification task poison training datasets with trigger-attached images with incorrect labels, with which the collaborative learning system eventually learns a shortcut from the triggers to the labels. Thus, images with the injected triggers would be classified into the predefined labels. For model poisoning, attackers compromise some participants and completely control their behaviour during the training. Then, attackers could directly alter the local model updates to influence the global model [92]. Fig. 2 illustrates the two types of poisoning.

B. Privacy Threats

Compared with standalone learning systems, one significant advantage of collaborative learning is tha teach participant only sends the local model update to the parameter server to protect the privacy of training data. However, since the updates are computed from the training samples, they still carry sensitive information, which makes collaborative learning systems vulnerable to many inference attacks. For instance, attackers can recover pixel-wise accuracy for image and token-wise matching for texts from the exchanged gradients at each iteration [26].

Membership vs. Property vs. Sample. According to different attack goals, we can classify existing attacks into three categories: membership, property, and samples inference attacks. Given a data record and black-box access to a model or updates, a membership inference attack determines whether the record was in the model's training dataset [103]. With the membership inference ability, an attacker can infer the presence of a specific data sample in a training dataset, which is a serious privacy threat, especially when the dataset contains sensitive samples. For example, if multiple hospitals work collaboratively to train a shared model on the records of patients with a certain disease, a participant or the parameter server can launch a membership inference attack to infer a specific patient's health condition, which directly affects his or her privacy.

Property inference attacks in collaborative learning [34], [104], [105] aim to infer properties of participants' training data that are class representatives or properties that characterize the training classes. Some attacks even allow an attacker to infer when a property appears and disappears in the dataset during the training process [34]. Sample inference attacks [106], [107] try to extract both the training data and their labels when attackers obtain model updates during the training phase. Recent work first adopt generate a dummy sample, then gradually reduce the distance between dummy simple and the grand truth through optimization algorithm [26], [108].

Passive vs. Active. On the basis of the behavior of adversaries, we classify privacy attacks in collaborative learning into two categories: passive and active attacks. In the passive mode, the attacker can only monitor the genuine computations by the training algorithm and the model or observe the updates and performs the aggregation operator without changing anything in the collaborative training procedure. In the active model, the attacker is allowed to do anything during the training procedure. For instance, as a participant, the attack can adversarially modify his parameter uploads. He can also send fake information to the parameter server(s) or his neighborhoods to increase his weights during the aggregation. A global attacker (a parameter server) can control the participants for the update at each iteration and adversarially modify the aggregate parameters that are sent to the target participant(s) in active mode. The active attacker can be further classified based on whether he has conspirators: Single attacker who carries out attacks by himself and Byzantine attacker who communicates and shares information with his conspirators. Byzantine attackers can collaborate to make the optimal attacks. The attackers can be the participants that have common interests or are controlled by a malicious adversary.

IV. Integrity Attacks

In this section, we summarize collaborative learning attacks that compromise the integrity of the trained global models.

We elaborate on two types of classical attacks: Byzantine and backdoor attacks. We list the most representative integrity attack algorithms in Table I.

A. Byzantine Attacks

Although data poisoning has shown huge impact on stand-alone model training systems [32], [109], researches recently show that model poisoning is much more effective than data poisoning on Byzantine attacks in the setting of collaborative learning scenarios [25], [33]. The intuition is that model poisoning and data poisoning both aim to modify the weights of local models. Obviously, the former has more direct impact.

Byzantine attacks assume that the attacker has the permission to access and modify the updates from a number of participants in a collaborative learning system. We call the modified updates malicious updates. In the averaging collaborative learning algorithm, it is straightforward to implement a Denial-Of-Service attack by sending a linear combination of a malicious update and other benign updates [28], which forces the averaged update to follow the malicious update. However, this attack could be simply filtered out as the magnitude of the linear combination often differs from benign ones. Alternatively, since model updates form a high dimensional vector, a feasible solution is to craft malicious updates by drifting benign update with a constrained value. Baruch et al. [33] demonstrate that slight perturbation is enough to circumvent magnitude-based defense policies. In their experiment, it shows a nearly 50% accuracy decline with one-fifth malicious clients. Moreover, to strengthen the effect of the attack, Bhagoji et al. [25] formalize this process as an optimization problem, which aims to find a suitable boosting value of malicious updates.

In a more relaxed setting, attackers could launch a more damaging version of updates if they know the aggregation rules of the server [38], [92]. This setting is reasonable in various scenarios, for example, the provider of the server may make the aggregation rule public for attracting potential participants [112]. The above attacking mathods have different costs in the number of iterations they perform. Fang et al. [92] terminate the optimization process once crafted updates bypass the aggregation rule, while Shejwalkar [38] try to find an approximate maximum in fruitful updates. Although Shejwalkar [38] achieves a slightly more serious accuracy decline from in the experiment, it usually takes dozens of extra costs of iterations of aggregation.

B. Backdoor Attacks

1) Data Poisoning: We first introduce data poisoning in the stand-alone backdoor attacks. A backdoor could be embedded in the neural networks trained by a compromised dataset [95], [96]. The methods of injecting backdoors by data poisoning assumes that the attacker controls a significant fraction of the training data. Therefore, backdoor attacks change the behaviour of the model only on specific attacker-chosen inputs via data poisoning [93], [96]. These methods could be categorized into two classes, including unclean and clean label stand-alone backdoors.

In an unclean label stand-alone backdoor, the adversary introduces a number of miss-classified data samples into the training set. It poisons the training examples and changes their labels. For example, Gu et al. [93] proposed the BadNets model, which injects a trigger pattern to a set of randomly selected training images. As the visible pattern in the poisoned images is easy to be observed, for achieving invisible backdoor attack, Li et al. [113] add sample-specific noise into the selected images using DNN-based image steganography [114]–[116] and Doan et al. [117] trained a stealthy trigger generator to craft trigger-images in imperceptible ways.

Since the poisoned images are mislabeled, unclean label attacks can be easily detected by simple data filtering or human inspection [102]. Therefore, clean label stand-alone backdoor is proposed. It assumes that the adversary cannot change the label of any training sample and preserves the labels of the poisoned samples. Visually, the tampered samples still look similar to the beginning ones. For example, Shafahi et al. [98] explored poisoning attacks on neural nets and presented an optimization-based feature collision attack method for crafting poisons. The experiments show that just one single poison image can control the classifier behavior when transfer learning is used. However, the method proposed by Shafahi et al. [98] requires a complete or a query access to the victim model. Then, Zhu et al. [110] assumed the victim model is not accessible to the attacker and proposed a new convex polytope attack in which poison images are designed to surround the targeted image in the feature space.

Soon afterward, Huang et al. [94] showed that the feature collision and convex polytopes attacks only work on fine-tuning and transfer learning pipelines, and fail when the victim trains their model from scratch. Furthermore, they are not general-purpose, an attacker could have objectives beyond a limited number of targets. In order to solve these difficulties, Huang et al. [94] proposed a MetaPoison algorithm for crafting poison images that manipulate the victim's training pipeline to achieve arbitrary model behaviors. It is a bi-level optimization problem, where the inner level corresponds to train a network on a poisoned dataset and the outer level corresponds to update those poisons to achieve a desired behavior on the trained model. Further, Turner et al. [111] introduced two techniques to strengthen the backdoor attack, including latent space interpolation using GANs and adversarial perturbations bounded by l_p-norm.

Data poisoning in collaborative learning systems follows the attacks in the stand-alone setting. Tolpegin et al. [100] investigated targeted data poisoning attacks against collaborative learning systems, in which a malicious subset of the participants aim to poison the global model by sending model updates derived from mislabeled data. However, Bagdasaryan et al. [30] pointed out that these attacks in the stand-alone setting are not effective against collaborative learning, where the malicious model is aggregated with hundreds or thousands of benign models. In order to implement a backdoor attack in collaborative learning systems, they [30] proposed a constrain-and-scale technique to inject backdoor in collaborative learning. Compared with previous backdoor attacks, in collaborative learning, the attacker controls the

TABLE I
TAXONOMY OF BYZANTINE AND BACKDOOR ATTACKS.

Methods		Parallelism		Consistency		Poisoning		Framework
		Data	Model	Sync	Async	Data Poisoning	Model Poisoning	
Byzantine	Jagielski [109]	-	-	-	-	✓		Standalone
	Munoz [32]	-	-	-	-	✓		Standalone
	Blanchard [28]	✓		✓			✓	Centralized
	Little [33]	✓		✓			✓	Centralized
	Shejwalkar [38]	✓		✓			✓	Federated
Backdoor	Badnets [93]	-	-	-	-	Unclean		Standalone
	Shafahi [98]	-	-	-	-	Clean		Standalone
	Zhu [110]	-	-	-	-	Clean		Standalone
	MetaPoison [94]	-	-	-	-	Clean		Standalone
	Turner [111]	-	-	-	-	Clean		Standalone
	Zhao [102]	-	-	-	-	Clean		Standalone
	Nguyen [97],Wang [31]	✓		✓		Clean		Federated
	Tolpegin [100]	✓		✓		Unclean		Federated
	DBA [101]	✓		✓		Unclean		Federated
	Sun [99]	✓		✓		Unclean		Federated
	Sun [53]	✓		✓			✓	Federated
	Bagdasaryan [30]	✓		✓			✓	Federated
	Fang [92]	✓		✓			✓	Federated
	Bhagoji [25]	✓		✓			✓	Federated

entire training process, but only for one or a few participants. Based on the above assumption, Nguyen et al. [97] indicated that the collaborative learning based IoT intrusion detection systems are vulnerable to backdoor attacks and proposed a data poisoning attack method. The core idea of this method is that it allows an adversary to implant a backdoor into the aggregated detection model to incorrectly classify malicious traffic as benign traffic. Finally, an adversary can gradually poison the detection model by only using compromised IoT devices to inject small amounts of malicious data into the training process. From another perspective, Wang et al. [31] focused on attacking algorithms that leverage data from the tail of the input data distribution. Then, they established in theory that, if a model is vulnerable to adversarial examples, under mild conditions, backdoor attacks are unavoidable. If backdoors are crafted properly, they are also hard to detect.

Although the backdoor attacks for collaborative learning systems mentioned above have good performance, they do not fully exploit the distributed learning methodology of collaborative learning, as they embed the same global trigger pattern to all adversarial parties [101]. In order to take the full advantage of the distributive nature of collaborative learning, Xie et al. [101] proposed a distributed backdoor attacking (DBA) method. DBA decomposes a global trigger pattern into separate local patterns and embeds them into the training set of different adversarial parties respectively.

2) Model Poisoning: In model poisoning, the training process is proceeded on local devices. Therefore, fully compromised clients can change the local model update completely, thereby altering the global model. For example, Bagdasaryan et al. [30] pointed out that a single or multiple malicious participants can use model replacement to introduce backdoor functionality into the joint model, e.g., modify an image classifier so that it assigns an attacker-chosen label to images with certain features, or force a word predictor to complete certain sentences with an attacker chosen word. Then, Bhagoji et al. [25] proposed a model poisoning method, which is carried out by an adversary that controls a small number of malicious agents (usually one), aiming to cause the global model to misclassify a set of chosen inputs with high confidence. Since backdoor attacks are more concealed than target attacks, following the research work of [30] and [25], Sun et al. [53] considered targeted model update poisoning attacks. Specifically, they concerned about backdoor attacks in collaborative learning and allowed non-malicious clients to have correctly labeled samples from the targeted tasks. The goal of an adversary is to reduce the performance of the model on targeted tasks while maintaining good performance on the main task. In addition, to achieve the purpose of destroying the integrity of the learning process in the training phase, Fang et al. [92] focused on the method to create an effective local model poisoning attacks in the Byzantine robust collaborative learning method.

V. INTEGRITY DEFENSES

A. Byzantine Defenses

Byzantine defense aims to filter out malicious participants using experience from updates, which could be mean or median of updates as well as the history of interactions. Therefore, we divide existing Byzantine-tolerant algorithms into two categories: statistic-based and learning-based, with a summation in Table II.

1) Statistic-based Inspection: Statistic-based inspection applies anomaly detection on participants' in each iteration of the training. Existing research takes two criteria: magnitude and performance. Blanchard et al. [28] proposed Krum to compute updates similarity using euclidean distance. The model select the one that minimizes the sum of the distances to all other updates as the global update. However, Krum endures high computational overhead when computing distances of high-dimensional vectors. Hence, Yin et al. [41] used the mean of dimensions to replace the euclidean distance, called Trimmed Mean. It treats each update independently, sorts each dimension of updates and removes β largest and smallest items,

TABLE II
TAXONOMY OF BYZANTINE DEFENSES.

Methods		Parallelism		Consistency		Methodology	Data Distribution		Framework
		Data	Model	Sync	Async		IID	Non-IID	
Statistic	Geometric Median [118], [119]	✓		✓		Median	✓		Centralized
	Krum [28]	✓		✓		Euclidean	✓		Centralized
	Trimmed mean [41]	✓		✓		Mean	✓		Centralized
	Bulyan [24]	✓		✓		Euclidean+Median		✓	Centralized
	Cao [35]	✓		✓		Euclidean		✓	Centralized
	Zeno [39]	✓		✓		Loss		✓	Centralized
	Zeno++ [40]	✓			✓	Loss		✓	Centralized
	Dnc [38]	✓		✓		SVD		✓	federated
	FAIR [120]	✓		✓		Loss	✓		federated
	BASGD [121]	✓			✓	Median/Mean		✓	Centralized
	Romoa [122]	✓			✓	Lookahead Similarity Measurement		✓	Decentralized
	El-Mhamdi [123]	✓			✓	Minimum-diameter Averaging/Median		✓	Decentralized
	UBAR [29]	✓			✓	Loss	✓		Decentralized
Learning	AFA [36]	✓		✓		Hidden Markov Model	✓		federated
	RLR [56]	✓		✓		Robust Learning Rate		✓	federated
	Justinian's GAAvernor [37]	✓		✓		Reinforcement Learning	✓		Centralized
	DeepSA [124]	✓		✓		DNN	✓		federated
	karimireddy [125]	✓		✓		Worker Momentum		✓	Centralized

then calculates the mean of remaining values as the global update. In addition, Krum is easily influenced by a single parameter. Therefore, Mhamdi et al. [24] proposed Bulyan, a combination of Krum and Trimmed Mean. It first runs Krum for several iterations to select a certain number of candidates, then it applies a variant of Trimmed Mean to calculate the global update. Moreover, there are also many median-based updates estimators, such as geometric median [118], [126], marginal median, mean around median [127], median of means(MOM) [119] and mean of median [128]. Furthermore, some researchers applied more sophisticated statistics techniques to compute updates similarity. Muñoz-González et al. [36] computed the weighted average of all updates and compute the cosine similarities between the averaged update to each update. Then, it removes updates with similarities out of a certain threshold. Shejwalkar et al. [38] presented Dnc, which uses Singular value decomposition (SVD) and dimensionality reduction to discard outliers.

All aforementioned magnitude methods can only deal with the scenario in which less than half of the participants are compromised. Some researchers expect to break through the above limitation using performance evaluation. [35], [39], [120]. As a compromise, these methods typically require a clean dataset. Xie [39] proposed Zeno in which the server sorts the updates by a stochastic descendant score. The score is composed of the estimated descendant of the loss function and the magnitude of the update, which roughly indicates how trustworthy each participant is. The server aggregates the updates with the highest score. Zeno requires that at least one benign update from all updates for proving the convergence of SGD for non-convex problems. Cao et al. [35] proposed an aggregation algorithm that can defense an arbitrary number of Byzantine attackers. It computes a benign update using the clean dataset and compares the updates from each participant with the benign update. The benign update is very noisy because the scale of the clean dataset could be quite small, while it is enough to filter out malicious information in experiments. Deng et al. [120] used loss reduction between the global model and the local models to evaluate the quality of the update from each participant. Guo et al. [29] proposed a Uniform Byzantine-resilient Aggregation Rule (UBAR) to select the useful parameter updates and filter out the malicious ones in each training iteration. It can guarantee that each benign node in a decentralized system can train a correct model under very strong Byzantine attacks with an arbitrary number of faulty participants. Furthermore, the above algorithms also inspire Byzantine robust solutions in asynchronous distributed learning [40], [121]–[123].

2) Learning-based Inspection: The learning-based inspection identifies malicious participants according to historical interactions. Lupu et al. [36] adopted a Hidden Markov Model to specify and learn the quality of model updates provided by each participant during training, which could enhance the accuracy and efficiency of detecting malicious updates. Pan et al. [37] proposed Justinian's GAAvernor, a gradient aggregation agent which learns to be robust against Byzantine attacks via reinforcement learning. It views the historical interactions as the experience and the relative decrease of loss on a clean dataset as the reward. It defines the credits of participants as the objective policy and optimizes the current policy after receiving the reward of the global update through reinforcement learning. Karimireddy et al. [125] observed that Byzantine updates have a significant deviation for certain rounds. Inspired by [129], they introduced momentum into computing benign updates and used simple iterative clipping to aggregate updates. Similarly, Ma et al. [124] used a crafted DNN to learn the correlation of benign updates in multiple rounds, which differs from Byzantine updates. Then, the DNN is treated as a classifier to sort out Byzantine updates.

B. Backdoor Defenses

To avoid or mitigate the effects of backdoor attacks on collaborative learning systems, several backdoor defense methods have been proposed [74], [130]–[132]. We divide existing methods into two categories based on the subject of inspection: data and the model inspection.

Data inspection methods mainly check whether the input data contains triggers through anomaly detection or just remove the abnormal samples during the inference process. Thus, existing data inspection methods for standalone learning [42], [44]–[47], [132] are appliable for well-trained models by collaborate learning systems. Thus, we summarize model

inspection defenses as below, especially the defenses for collaborative learning systems.

Data inspection defenses try to distinguish poisoned data from normal ones, while the model inspection approach [48], [49] relies on anomaly technique to distinguish abnormal behaviour of the models caused by backdoors [130]. These defenses can be carried out during or after the training processing. For model inspection for well-trained models, Wang et al. [50] proposed Neural Cleanse to detect whether a DNN model has been subjected to a backdoor attack or not prior to deployment. Taking advantage of output explanation techniques, Huang et al. [51] proposed Neuron Inspect to identify backdoor attacks by outlier detection based on the heatmap of the output layer. Liu et al. [49] proposed Artificial Brain Stimulation to detect backdoors by analyzing the inner neuron behaviors through a stimulation method. Nevertheless, Chen et al. [52] pointed out that it is indispensable to inspect whether a pre-trained DNN has been trojaned before employing a model. Hence, they proposed DeepInspect, a black-box trojan detection solution. It learns the probability distribution of potential triggers from the queried model using a conditional generative model. Ma et al. [48] pointed out that existing detection techniques only work well for specific attacks under various strong assumptions. They proposed NIC by checking the provenance channel and the activation value distribution channel. They extract DNN invariants and use them to perform run-time adversarial sample detection including trigger input detection.

In addition to detecting backdoor or backdoored models after the training processing, several backdoor defenses are proposed to mitigate the impact of backdoor during the collaborative training processing. For example, Sun et al. [53] studied backdoor and defense strategies in collaborative learning and showed that norm clipping and weak differential privacy can mitigate the attacks without hurting the overall model performance. Liu et al. [54] introduced additional training layers at the active party for backdoor defense. The active party first concatenates the output of the passive parties and adopts a dense layer before the output layer. Zhao et al. [55] and Andreina et al. [133] presented defense schemes to detect anomalous updates in both IID and non-IID settings with a key idea of realizing client-side cross-validation, where each update is evaluated over the local data from other participants. Jingwei el al. [134] proposed a more challenging task that defending backdoor attacks on participants when the global model is polluted. They designed a client-based defense named FL-WBC to perturb the parameter space where long-lasting backdoor attacks resides. In addition, recent works [26] shown that gradient sparsification is an effective approach to defend backdoor attacks in collaborative learning, as well as to achieve a robust learning rate [56]. Wu et al. [135] proposed a federated pruning method to remove redundant neurons of the shared model and then adjust the extreme weight values to mitigating backdoor attacks in federated learning systems.

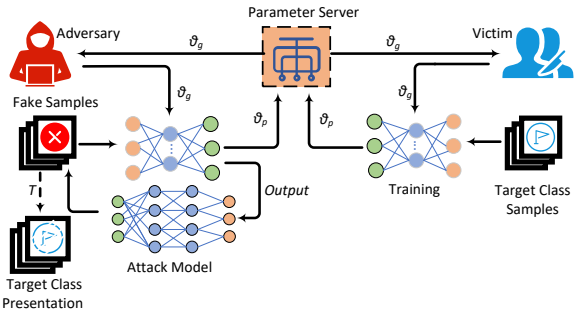

Fig. 3. Property inference attack architecture in collaborative learning systems.

VI. Privacy Attacks

A. Threat model

As shown in Section III-B, privacy attacks aims to infer private information about the training samples of workers. Figure 3 illustrates the property inference attack architecture, where some participating nodes are potential attackers. They use the aggregated parameters to gradually generate the target class representations of other participants. Malicious participants can also implement privacy attacks to gain the membership information or training samples of others. On the other hand, the parameter server that obtains the updated gradients of all participants at each iteration can also be the attacker, from which he can infer membership, properties or even training samples from the updates. According to the background knowledge of aggregated model, privacy attacks can be divided into two categories: white-box and black-box. The attackers can only access model outputs in the black-box mode, while in the white-box mode, attackers know the model structure and all parameters of the model. We summarize popular privacy attacks in collaborative learning systems in Table III.

B. Membership Inference

For stand-alone learning, an attacker can only observe the final target model that is trained by only one participant. Prior work has presented passive and active membership inference attacks against stand-alone DL models [146]–[149] but collaborative learning presents interesting new avenues for such inference attacks. In collaborative learning systems, the attacker can be the parameter server or any of the participant nodes. While the parameter server observes individual updates over time and can control the view of all participants on the global parameters, each participant observes the global parameter updates and can control its parameter uploads. Therefore, compared with the attacks in stand-alone learning, the parameter server and participants have more information of the updates of each iteration and are easier to carry out membership inference attacks.

Melis et al. [34] proposed a membership inference attack for learning tasks on text record datasets. In particular, at each iteration, the attacker, i.e., an honest-but-curious participant, receives the current aggregated updates, from which he can get the aggregated updates from other participants. Melis et

TABLE III
PRIVACY ATTACKS IN COLLABORATIVE LEARNING SYSTEMS

Methods		Parallelism		Consistency		Knowledge		Identity		Action		Framework	Task
		Data	Model	Sync	Async	White-box	Black-box	Participant	Server	Passive	Active		
Membership	Melis [34]	✓		✓			✓	✓		✓	✓	Federated	NLP
	Nasr [136]	✓		✓		✓		✓	✓	✓	✓	Federated	CV
	Zhang [137]	✓		✓		✓		✓		✓		Federated	CV
	Yuan [138]	✓			✓	✓		✓		✓	✓	Federated	NLP
Property	Hitaj [104]	✓		✓		✓		✓		✓		Federated	CV
	Wang [105],Song [139]	✓		✓		✓			✓	✓	✓	Federated	CV
	Zhang [140]	✓		✓			✓	✓		✓		Centralized	NLP
Sample	Zhu [26]	✓		✓			✓		✓	✓		(De)centralized	CV
	Zhao [108]	✓		✓			✓		✓	✓		Centralized	CV
	Geiping [106]	✓		✓			✓		✓	✓		Centralized	CV
	Yin [141]	✓		✓			✓		✓	✓		Centralized	CV
	Dang [142]	✓		✓			✓		✓	✓		Centralized	CV
	Jin [143]	✓		✓			✓		✓	✓		Federated	CV
	Fu [144]	✓		-	-	-	✓	✓		✓		Federated	CV
	He [145]		✓	✓		-	-	-	-		✓	Federated	CV

al. observe that the aggregated gradient of an embedding layer is sparse with respect to the training text. Given a batch of training text, the embedding layer transforms the inputs into a lower-dimensional vector representation and the corresponding parameters are updated only with the words that appear in the batch. The gradients of the other words are zeros. Thus, the aggregated updates/gradients directly reveal which words occur in the training texts used by other honest participants during the collaborative learning process.

Unfortunately, the membership inference attack [34] works exclusively for the learning tasks whose models use explicit word embeddings with small training mini-batches. Nasr et al. [136] presented a more standard and comprehensive framework for the privacy analysis in collaborative learning systems. Specifically, Nasr et al. proposed white-box membership inference attacks by analyzing the privacy leakage from the stochastic gradient descent algorithm and evaluated the attacks under various adversarial models with different types of prior knowledge and abilities. Nasr et al. show that in collaborative learning, the update history on the same training datasets would reveal privacy information and boost the inference attack accuracy. A local passive attacker can perform membership inference attacks against other participants with the maximum inference accuracy of 79.2%. They further proposed an active attack that actively performs gradient ascent on a set of target data points to influence the parameters of other parties. This magnifies the presence of the data points in others' training sets. The attacker judges whether the target points are members by observing the reacts of the gradients on them. The accuracy of the active inference attack would be boosted by a significant increase under a global attacker.

Zhang et al. [137] focused on the scenario that the attack is launched by one of the participants and proposed a passive attack using the generative adversarial network (GAN). The attack uses GAN to enrich attack data and increase data diversity that is used to query the target collaborative learning model. The models trained using the new sample-label pairs are vulnerable to membership inference attacks. Yuan et al. [138] explore record data leakage against NLP in asynchronous distributed learning which would cause imbalanced performance of training across participants. Through eavesdropping on the selection of participants or injecting a single watermark into the victim, they can successfully obtain the privacy records and reveal the identifies of participants.

C. Property Inference

With the aggregated updates from server, attackers could gradually determine the class representation (i.e. property) of participants' training data. For example, Hitaj et al. [104] proposed a GAN-based attack to extract class representation information from honest participants in collaborative learning systems. The attack employs a GAN to generate instances that are visually similar to the samples from a targeted class of a victim participant. In particular, the attack first generates some fake samples from the targeted class that are injected into the training dataset as another class. In this way, the victim participant would reveal privacy information about the targeted class since he has to distinguish between the two classes. Taking the advantage of the knowledge about the targeted class and the density estimation of GAN, the attacker can learn the distribution of the targeted class without accessing the training points of the victim participant directly. The attack is effective against the collaborative learning tasks with convolutional neural networks, even when the parameters are obfuscated via differential privacy techniques.

The GAN-based class representation attack infers only properties of the entire targeted class and assumes that the victim participant owns the entire training points of the targeted class. In contrast, Melis et al. [34] release the constrained assumptions and proposed property inference attacks to extract unintended information about participants' training data from the update history. Specifically, at each training iteration, the attacker saves the snapshot of the aggregated update parameters. The difference between the consecutive snapshots is equal to the aggregated updates from all participants. This difference directly reveals privacy information in the training batches of the honest participants during collaborative learning. Melis et al. proposed property attacks in both passive and active modes:

- Passive property inference: consider that the attacker has auxiliary data consisting of the data points that have the property of interest and data points that do not have the property. The intuition behind the attack is that the adversary can leverage the snapshots of the global model

to generate aggregated updates based on the data with the property and updates based on the data without the property. This produces labeled examples that enable the adversary to train a binary batch property classifier that determines if the observed updates are based on the data with or without the property.
- Active property inference: the active attacker can perform a more powerful attack using multi-task learning. The adversary extends his local copy of the collaboratively trained model with an augmented property classifier connected to the last layer. He trains this model to simultaneously perform well on the main task and recognize batch properties.

Similar to [104], Wang et al. [105], [139] proposed GAN-based attacks against collaborative learning systems to target client-level privacy. In the proposed attack, the parameter server is malicious and cannot access the target data. Since GANs could generate conditioned samples, the attacker trains GANs conditioned on the updates from the victim participants, thus it could generate victim-conditioned samples which contain client-level privacy information. They also consider both passive and active modes.

- Passive inference: the malicious server is assumed to be honest-but-curious and only analyzes the updates from the participants by training GANs.
- Active inference: the active attacker isolates the victim participants from the others, i.e., training GANs on the victim alone by sending a special version of the aggregated model to the victim participants.

The above methods require updates information during the training process, i.e., in the white-box mode. Instead, Zhang et al. [140] suppose the adversary can only black-box access to the global model. Depending on the correlation relationship sensitive attributes with other attributes or labels, they train a sequence of shadow networks and a meta-classifier to learn the distribution of sensitive attributes in a few queries.

D. Sample Inference

Collaborative learning systems use the gradient sharing framework to avoid data leakage of participants, which is less effective in recent sample inference attacks. Zhu et al. [26] first pointed out that the sharing gradients can leak private training data. They presented an optimization algorithm, deep leakage from gradients (DLG), that can obtain both the training inputs and the labels in just a few iterations. The attack first randomly generates a pair of "dummy" inputs and labels and then derives the dummy gradients from the dummy data. The attack optimizes the dummy inputs and labels to minimize the distance between dummy gradients and real gradients. The private training data will be fully revealed by matching the gradients makes the dummy data close to the original ones.

Although DLG works, Zhao et al. [108] found that it is not able to reliably extract the ground-truth labels or generate good quality training samples. Zhao et al. proposed a simple yet efficient sample inference attack to extract the ground-truth labels from the shared gradients. They demonstrate that the gradient of the classification loss can distinguish correct label from others by derivation. With such observation, the attacker can identify the ground-truth labels based on the shared gradients. Then, the attacker can significantly simplify the DLG attack and extract good-quality training samples.

Later, numerous sample reference attacks are devoted to improve the effectiveness of the revealing training samples and labels [141]–[144], [150]. For example, Yin et al. [141] present GradInvision to recover a single image from the averaged gradients. In particular, GradInvision first performs label revealing from the gradients of the fully-connected layer and then optimizes random inputs to match the target gradients using fidelity regularization and get a better quality of the reconstructed image. Dang et al. [142] consider that participants compute updates with reasonable small batch size and proposed RLG (Revealing Labels from Gradients) that reconstructs training samples from only the gradient of the last layer. Meanwhile, Chen et al. [150] and Fu et al. [144] investigate the large-batch data leakage in vertical federated learning and He et al. [145] explore the sample reconstruction in the model parallelism architecture.

The above sample inference attacks maily rely on two components: the euclidean cost function and optimization via LBFGS. Geiping et al. [106] argue that these choices are not optimal for more realistic architectures and especially arbitrary parameter vectors and proposed to use a cost function based on angles, i.e. cosine similarity. One the one hand, the magnitude measures local optimality of the data point and only captures information about the state of training. On the other hand, the angle quantifies the change in prediction at one data point when taking a gradient step towards another.

VII. Privacy Defenses

Inspired by the privacy attacks, a large amount of privacy defenses are proposed to protect the training samples from being inferred. According to the commonly used privacy-preserving techniques, we can classify existing privacy defenses into three categories: differentially private, privacy-preserving and cryptographic privacy-preserving collaborative learning. We elaborate state-of-the-art privacy defenses as below.

A. Differentially Private Collaborative Learning

To mitigate membership inference attacks in collaborative training systems, one promising solution is Differential Privacy (DP), which is a rigorous mathematical framework to preserve the privacy of individual data records in a database when the aggregated information about this database is shared among untrusted parties [151], [152]. A number of studies have applied DP to enhance the privacy of DL training in different environments [57]–[62]. Most existing DP-SGD algorithms adopt additive noise mechanisms by adding random noise to the estimates in every training iteration. There exists a trade-off between privacy and usability, determined by the noise scale added during training: adding too much noise can meet the privacy requirements, at the cost of the drop in model accuracy. As a result, it is critical to identify the minimal

amount of noise that can provide desired privacy protection as well as maintain acceptable model performance.

Two common approaches were devised to optimize the DP mechanisms and balance the privacy-usability trade-off. The first one is to carefully restrict the sensitivity of randomized mechanisms. For example, Abadi et al. [58] bounded the influence of training samples on gradients by clipping each gradient in l_2 norm below a given threshold. Yu et al. [61] optimized the model accuracy by adding decay noise to the gradients over the training time since the learned models converge iteratively. The second approach is to precisely track the accumulated privacy cost of the training process using composition techniques such as the strong composition theorem [152] and moments account (MA) [58], [153]–[155]. In the following, we first illustrate exiting frequently used DP techniques and then summarize differentially private solutions for collaborative learning systems.

1) DP Techniques: For any two neighboring datasets that only differ in one single record, a randomized mechanism $\mathcal{M}$ is differentially private if its outputs are almost the same on the two datasets. The formal definition of DP is illustrated as follows.

Definition 1. *((ϵ, δ)-DP) A randomized mechanism $\mathcal{M} : D \rightarrow R$ with domain D and range R satisfies (ϵ, δ)-DP if for any two neighboring datasets D_1, D_2 and any subset of outputs $S \subseteq R$, the following property is held:*

$$Pr[\mathcal{M}(D_1) \in S] \leq e^{\epsilon} Pr[\mathcal{M}(D_2) \in S] + \delta. \tag{3}$$

$\mathcal{M}$ can satisfy DP is restricted by two parameters: ϵ and δ. ϵ is the privacy budget to limit the privacy loss of individual records. δ is a relaxation parameter that allows the privacy budget of $\mathcal{M}$ to exceed ϵ with probability δ. It has been proven that differential privacy satisfies a composition property: when with privacy budgets ϵ_1 and ϵ_2 are performed on the same data, the privacy budget of the combined of the two mechanisms equals to the sum of the two privacy budgets, i.e., $\epsilon_1 + \epsilon_2$.

Relaxed Definitions. Due to the composition property, composing multiple differentially private mechanisms leads to a linear increase in the privacy budget and the scale of the corresponding noise increases to maintain a fixed total privacy budget. To get a better privacy-usability trade off, multiple DP techniques reduce the this linear composition bound at the cost of slightly increasing the failure probability. There are two commonly used relaxations of differential privacy that exhibits better accuracy than (ϵ, δ)-DP, which use different versions of divergences to calculate the distributional difference between the outputs of $\mathcal{M}$ in adjacent datasets: Concentrated Differential Privacy (CDP) and Rényi Differential Privacy (RDP). CDP uses the sub-Gaussian divergence to restrict the mean and standard deviation of the privacy loss variable. It get better accuracy and any ϵ-DP algorithm satisfies $(\epsilon \cdot (e^{\epsilon} - 1)/2, \epsilon)$-CDP. Rényi DP (RDP) [156] is a natural relaxation of DP based on the Rényi divergence and allows tighter analysis of tracking cumulative privacy loss. The instantiation of RDP is MA, which keeps track of a cumulative bound on the moments of the privacy loss.

2) DP-SGD for Collaborative Learning: For single-party learning, there are two common candidates for where to add random noise: the objective function [57], [157] and gradients [58], [61]. For the first approach, Chaudhuri et al. [57] perturb the objective function before optimizing over classifiers and show that the objective perturbation is DP if certain convexity and differentiability criteria hold. Phan et al. [157] attempt to use the objective perturbation by replacing the non-convex function with a convex polynomial function. To this end, Phan et al. [157] design a convex polynomial function to approximate the non-convex one, which however would change the learning protocol and even sacrifice the model performance. On the other hand, adding random noise to the gradients is a simpler and popular approach in single-party learning. For example, Abadi et al. [58] bound the influence of training samples on gradients by clipping each gradient in l_2 norm below a given threshold to restrict the sensitivity of randomized mechanisms. Yu et al. [61] focus on differentially private model publishing and optimize the model accuracy by adding decay noise to the gradients over the training time since the learned models converge iteratively.

Another way to improve the model usability lies in precisely tracking the cumulative privacy cost of the training process. For example, Shokri et al. [158] and Wei et al. [159] composed the additive noise mechanisms using the advanced composition theorem [152], leading to a linear increase in the privacy budget. Some DP-SGD methods [58], [153]–[155] use MA to reduce the added noise during the training process. Other algorithms [61], [160], [161] were designed to improve the model usability using (zero) concentrated DP [162].

Some works [122], [153]–[155], [158], [161], [163]–[167] applied the DP techniques from the standalone mode to the distributed systems to preserve the privacy of the training data for each agent. For example, Shokri et al. [158] proposed a privacy-preserving distributed learning algorithm by adding Laplacian noise to each agent's gradients to prevent indirect leakage. Kang et al. [155] adopted weighted aggregation instead of simply averaging to reduce the negative impact caused by uneven data scale in collaborative learning systems.

In terms of the accumulated privacy loss, Kang et al. [155] employed MA to track the overall privacy cost of the collaborative training process. Wei et al. [159] and Wei [164] perturbed agents' trained parameters locally by adding Gaussian noise before uploading them to the server for aggregation and bounded the sensitivity of the Gaussian mechanism by clipping in federated learning systems. Shokri et al. [158] and Wei et al. [159] composed the additive noise mechanisms using the strong composition theorem [152], leading to a linear increase in the privacy budget. In order to reduce aggregated noise in local updates, Han et al. [163]dynamically adjust batch size and noise level according to the rate of critical input data and the sensitivity estimation.

B. Cryptographic Privacy-preserving Collaborative Learning

Although DP techniques are widely used in collaborative learning as its clear theory and concise algorithm, they are designed against membership inference attacks and hard to

defend property and sample inference attacks. Additionally, the noise added to the updates would reduce the performance of the trained models, especially when the participants are extremely sensitive to privacy leakage. Due to the drawbacks of DP techniques, several privacy-preserving collaborative learning methods are proposed using cryptographic tools as we elaborate below.

Collaborative Learning with Homomorphic Encryption. Homomorphic Encryption(HE) permits users to perform arithmetic operations directly on ciphertext, which is equivalent to executing the same operations on the corresponding plaintext. Since HE techniques only require participants to share encrypted data, they can provide cryptographic privacy protection in collaborative learning scenarios. There are two types of HE schemes: Fully Homomorphic Encryption(FHE) and Partially Homomorphic Encryption(PHE). FHE allows addition and multiplication on encrypted data, while PHE only support one of them. Correspondingly, FHE is much more computationally expensive than PHE. Several privacy-preserving collaborative learning approaches have been proposed to use PHE to ensure the privacy of individual model updates [63], [168]–[171]. For example, Aono [63], Phong [168], and PPFDL [169] perform the addition operation over encrypted updates to protect the privacy of the updates during the aggregation process.

To reduce the cost of homomorphic linear computation, Zhang et al. [170] considers homomorphic linear computation as a sequence addition operations of addition, multiplication, and permutation and then greedy chooses the least expensive operation for every computation step. Froelicher et al. proposed SPINDLE [172] that preserves data and model confidentiality and enables the execution of a cooperative gradient-descent and the evaluation of the obtained model even when there are colluding participants. Stripelis et al. [173] proposed a secure federated learning framework FL using FHE techniques to protect training data and the shared updates. [173] However, HE has some limitations. For example, the memory and arithmetic cost of encrypted data is much higher than that of the original plaintext. And HE has to use polynomial approximations to handle common non-linear operations in collaborative learning systems.

Collaborative Learning with Secure Muti-Party Computation. Another widely applied cryptographic method is secure multi-party computation (SMC), which allows mutual distrust participants to jointly compute a function over their inputs and preserve the privacy of inputs [65], [66], [174], [175]. Bonawitz et al. [65] proposed a communication-efficient, failure-robust secure aggregation of high dimensional model updates without learning each participant's sensitive information with SMC, which can defend both passive and active adversaries. Li et al. [66] proposed a privacy-preserving collaborative learning framework based on the chained SMC technique. With such framework, adversaries can not obtain any privacy of participants as the output of a single participant is dissimulated with its prior. Comparing to HE, SMC has less computation cost and communication overhead, but it is still not applicable to large-scale collaborative learning, especially collaborative learning systems with thousands of participants.

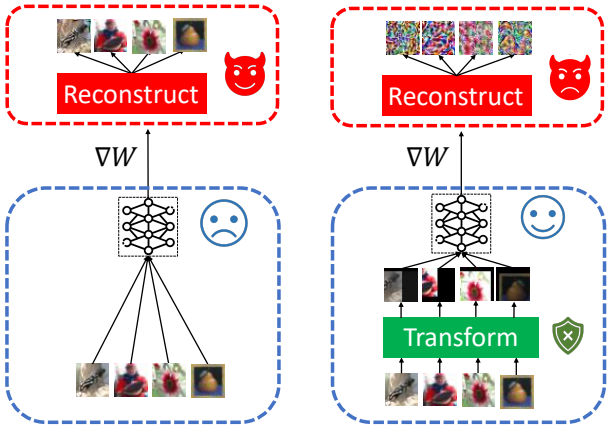

Fig. 4. Automatic transformation search against deep leakage from gradients [67].

C. Practical Privacy-preserving Collaborative Learning

Besides the above privacy defenses whose security can be theoretically guaranteed, a large amounts of privacy-preserving collaborative learning methods are proposed to protect the privacy of participants in real collaborative learning scenarios. Similar to integrity defenses, these privacy defenses also focus on processing training data or model updates to experimentally guarantee privacy information against inference attacks. Figure 4 illustrates a privacy-preserving collaborative learning method [67] using automatic transformation search against deep leakage from gradients. The method transforms original local data samples into related samples for disabling sample inference attacks by searching specific transformation.

Such approach is much more efficient than the defenses with cryptographic techniques to thwart inference attacks. Zhao et al. [176] proposed a framework that transfers sensitive samples to public ones with privacy protection, based on which the participants can collaboratively update their local models with noise-preserving labels. Fan et al. [177] designed a secret polarization network for each participant to produce secret losses and calculate the gradients. Huang et al. [178] proposed to combine existing sample inference defenses in an appropriate manner to enhance the protection performance.

VIII. Hybrid Defenses and Beyond

A. Hybrid Defenses

Existing investigations [70] have shown that defenses against one type of attacks cannot be directly applied to the other type of attacks. Thus, besides the defenses that aim to prevent one type of threats, several methods [68], [69], [71], [72], [179]–[181] are proposed to defend both integrity and privacy attacks and build robust and privacy-preserving collaborative learning systems. Theses hybrid defenses mainly take advantage of techniques against both integrity and privacy attacks. We elaborate state-of-the-art hybrid defenses as below.

One main design strategy of hybrid defenses [68], [69], [72] are to merge existing defenses for integrity and privacy

defenses together to achieve secure collaborative learning systems. For example, Ma et al [68] utilize an existing Byzantine-robust federated learning algorithm and distributed Paillier encryption and zero-knowledge proof to guarantee privacy and filter out anomaly parameters from Byzantine participants. Qi et al [71] achieve hybrid defense using blockchain and differential privacy techniques.

Some hybrid defenses leverage homomorphic encryption techniques that provide both confidentiality and computability over encrypted data. For example, Liu et al [72] proposed a homomorphic encryption scheme that enables privacy protection and provides the parameter server a channel to punish poisoners under ciphertext. Dong et al [180] employ two non-colluding servers and proposed an oblivious defender for private Byzantine-robust federated learning using additive homomorphic encryption and secure two-party computation primitives. However, homomorphic encryption based defenses requires a large amount of computations resources. Domingo et al [181] offer privacy to the participants as well as robustness against Byzantine and poisoning attacks via unlinkable anonymity, which can detect bad model updates while reducing the computational overhead compared to homomorphic encryption based defenses.

B. Collaborative Adversarial Training

Deep models are vulnerable to adversarial examples that are maliciously constructed to mislead the models to output wrong predictions but visually indistinguishable from normal samples [182]–[185]. Adversarial training [186]–[188] is one of the most effective approaches to defend deep models against adversarial examples and enhance their robustness. Its main idea is to augment training data with existing adversarial example generation methods during the training process. Thus, the adversarially trained models are more robust against adversarial examples during the inference process. Numerous of adversarial training methods [189] have been proposed for standalone training systems. For example, Shafahi et al [186] proposed an efficient adversarial training algorithm that recycles the gradient information computed at each iteration to eliminate the overhead cost of generating adversarial examples. Wong et al [188] propose to utilize Fast Gradient Sign Method (FGSM) [182] during the adversarial training process. They introduce random initialization points to improve the effectiveness the projected gradient descent based training.

Although standalone adversarial training achieves great success, challenges rise when extending these standalone algorithms to collaborative learning systems. One main challenge is that participants in collaborative systems only have limited training data and computational resources and cannot support the data-hungry and costly adversarial training. To this end, several adversarial training algorithms [190]–[192] have been proposed specifically for collaborative learning systems. For instance, Hong et al [190] proposed an effective propagation methods that transfers adversarial robustness from high-resource participants that can afford adversarial training to low-resource participants. Zhou et al [191] conduct collaborative adversarial training by composing the aggregation error of the parameter server(s) into bias and variance and using the bias-variance adversarial examples to improve model robustness. Shah et al [192] consider communication constrained federated learning environments and proposed an dynamic adversarial training methods to improve both adversarial robustness and model convergence speed.

IX. Open Problem

Although extensive research has been proposed to address the integrity and privacy threats in collaborative learning, some interesting and important issues remain to be fully explored. Here, we present several open problems and potential research directions below to stimulate further research:

Non-IID or Noisy Scenarios in Byzantine Attacks and Defenses. Byzantine attacks and defenses is an arms race between attackers and defenders: attackers intend to design malicious updates that are indistinguishable from normal ones, while defenders try to identify potential Byzantine updates and ensure the integrity of the trained models. Most of existing Byzantine resilient algorithms only consider IID training scenarios, where the training datasets of benign participants are IID. However, the training datasets are Non-IID in most real cases because the quality and distribution of each training dataset is different. Thus, it is more difficult for defenders to distinguish benign and malicious updates. For example, a malicious participant can disguise as a node with poor training data quality and generate updates that are indistinguishable from normal ones but fatal to model integrity. Although several works [39], [193] attempt to propose Byzantine resilient aggregation rules in Non-IID scenarios, they fail to defend advance Byzantine attacks or only consider few types of Non-IID scenarios [193].

Certified Backdoor Defenses. Existing backdoor defenses for collaborative learning mainly focus on empirically identifying or removing backdoors. Such defenses work well for existing backdoor attacks but can hardly identify or remove new advance attacks. Therefore, certified backdoor defenses for collaborative learning are urgently needed and provide provable protection against backdoor attacks. Unfortunately, most of existing certified backdoor defenses [194], [195] are proposed for standalone machine learning systems and few of them [196] are designed for collaborative learning.

Privacy-performance Tradeoff in Differential Privacy. Differential privacy techniques require adding noise onto the updates/models to defend membership inference attacks. Although several relaxation methods have been proposed to reduce the scale of noise, the performance is still unsatisfactory, especially when the parameters of the trained neural networks are large [103], [164]. One possible research direction is to utilize the system features of collaborative learning systems to reach a better privacy-performance tradeoff.

Basis Datasets in Property Inference Attacks. Several attacks [34], [104] utilize local datasets to infer the property of other participants. Such local datasets are assumed to have the same distribution with victim participants and critical to the inference attacks. However, such IID datasets limit the threat of these attacks because adversaries may not know

the distribution of the training datasets of victim participants. Thus, how to conduct property inference attacks with basis datasets is worth of in-depth study.

Performance Improvement in Sample Inference Defenses. Samples inference defenses [67], [178] can protect training samples from being inferred by existing attacks. However, some defenses such as adding noise or pruning parameters would harm the performance of the collaboratively trained models. Thus, it is necessary to design new defenses that can enhance both privacy and performance of collaborative learning.

X. Conclusion

In this survey, we systematically introduce state-of-the-art integrity and privacy threats in collaborative learning systems, which mainly include byzantine and backdoor attacks as well as three kinds of data inference attacks. And we also detailed describe corresponding defensive policies, including model and data based inspection against integrity attacks and differential privacy and encryption techniques against privacy attacks. The tradeoff between security and performance is the crucial point of new defense schemes. Additionally, we discuss a number of open problems of current defense methods, hoping it could help researchers identify and solve issues more quickly in the area of robust and privacy- preserving collaborative learning.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[3] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE international conference on acoustics, speech and signal processing*, 2013, pp. 6645–6649.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[5] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[6] M. Gawali, C. Arvind, S. Suryavanshi, H. Madaan, A. Gaikwad, K. B. Prakash, V. Kulkarni, and A. Pant, "Comparison of privacy-preserving distributed deep learning methods in healthcare," in *Annual Conference on Medical Image Understanding and Analysis*, 2021, pp. 457–471.

[7] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[8] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang *et al.*, "Large scale distributed deep networks," *Advances in neural information processing systems*, vol. 25, pp. 1223–1231, 2012.

[9] D. Peteiro-Barral and B. Guijarro-Berdiñas, "A survey of methods for distributed machine learning," *Progress in Artificial Intelligence*, vol. 2, no. 1, pp. 1–11, 2013.

[10] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, "Federated learning for keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 6341–6345.

[11] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large cnns at the edge," *arXiv preprint arXiv:2007.14513*, 2020.

[12] W. Zheng, L. Yan, C. Gou, and F.-Y. Wang, "Federated meta-learning for fraudulent credit card detection." in *IJCAI*, 2020, pp. 4654–4660.

[13] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[14] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[15] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *USENIX Symposium on Operating Systems Design and Implementation*, 2014, pp. 583–598.

[16] P. Moritz, R. Nishihara, I. Stoica, and M. I. Jordan, "Sparknet: Training deep networks in spark," *arXiv preprint arXiv:1511.06051*, 2015.

[17] M. Liu, W. Zhang, Y. Mroueh, X. Cui, J. Ross, T. Yang, and P. Das, "A decentralized parallel algorithm for training generative adversarial nets," *arXiv preprint arXiv:1910.12999*, 2019.

[18] T. Sun, D. Li, and B. Wang, "Stability and generalization of the decentralized stochastic gradient descent," *arXiv preprint arXiv:2102.01302*, 2021.

[19] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," *arXiv preprint arXiv:1812.06127*, vol. 3, p. 3, 2018.

[20] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," *arXiv preprint arXiv:2003.00295*, 2020.

[21] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *arXiv preprint arXiv:2007.07481*, 2020.

[22] Y. Lu and C. De Sa, "Optimal complexity in decentralized training," in *International Conference on Machine Learning*, 2021, pp. 7111–7123.

[23] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

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

[25] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *International Conference on Machine Learning*, 2019, pp. 634–643.

[26] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 747–14 756.

[27] D. Zhang, X. Chen, D. Wang, and J. Shi, "A survey on collaborative deep learning and privacy-preserving," in *IEEE International Conference on Data Science in Cyberspace*, 2018, pp. 652–658.

[28] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, 2017.

[29] S. Guo, T. Zhang, H. Yu, X. Xie, L. Ma, T. Xiang, and Y. Liu, "Byzantine-resilient decentralized stochastic gradient descent," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[30] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*, 2020, pp. 2938–2948.

[31] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos, "Attack of the tails: Yes, you really can backdoor federated learning," *arXiv preprint arXiv:2007.05084*, 2020.

[32] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 27–38.

[33] M. Baruch, G. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," *arXiv preprint arXiv:1902.06156*, 2019.

[34] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *IEEE Symposium on Security and Privacy*, 2019, pp. 691–706.

[35] X. Cao and L. Lai, "Distributed gradient descent algorithm robust to an arbitrary number of byzantine attackers," *IEEE Transactions on Signal Processing*, vol. 67, no. 22, pp. 5850–5864, 2019.

[36] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *arXiv preprint arXiv:1909.05125*, 2019.

[37] X. Pan, M. Zhang, D. Wu, Q. Xiao, S. Ji, and Z. Yang, "Justinian's gaavernor: Robust distributed learning with gradient aggregation agent," in *USENIX Security Symposium*, 2020, pp. 1641–1658.

[38] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," *Internet Society*, p. 18, 2021.

[39] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *International Conference on Machine Learning*, 2019, pp. 6893–6901.

[40] ——, "Zeno++: Robust fully asynchronous sgd," in *International Conference on Machine Learning*, 2020, pp. 10 495–10 503.

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

[42] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Advances in Neural Information Processing Systems*, 2018, pp. 8000–8010.

[43] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," *arXiv preprint arXiv:1811.03728*, 2018.

[44] A. Chan and Y.-S. Ong, "Poison as a cure: Detecting & neutralizing variable-sized backdoor attacks in deep neural networks," *arXiv preprint arXiv:1911.08040*, 2019.

[45] E. Chou, F. Tramèr, G. Pellegrino, and D. Boneh, "Sentinet: Detecting physical attacks against deep learning systems," *arXiv preprint arXiv:1812.00292*, 2018.

[46] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "Strip: A defence against trojan attacks on deep neural networks," in *Computer Security Applications Conference*, 2019, pp. 113–125.

[47] L. Truong, C. Jones, B. Hutchinson, A. August, B. Praggastis, R. Jasper, N. Nichols, and A. Tuor, "Systematic evaluation of backdoor data poisoning attacks on image classifiers," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 788–789.

[48] S. Ma and Y. Liu, "Nic: Detecting adversarial samples with neural network invariant checking," in *Network and Distributed System Security Symposium*, 2019.

[49] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "Abs: Scanning neural networks for back-doors by artificial brain stimulation," in *ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1265–1282.

[50] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *IEEE Symposium on Security and Privacy*, 2019, pp. 707–723.

[51] X. Huang, M. Alzantot, and M. Srivastava, "Neuroninspect: Detecting backdoors in neural networks via output explanations," *arXiv preprint arXiv:1911.07399*, 2019.

[52] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks." in *IJCAI*, 2019, pp. 4658–4664.

[53] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" *arXiv preprint arXiv:1911.07963*, 2019.

[54] Y. Liu, Z. Yi, and T. Chen, "Backdoor attacks and defenses in feature-partitioned collaborative learning," *arXiv preprint arXiv:2007.03608*, 2020.

[55] L. Zhao, S. Hu, Q. Wang, J. Jiang, S. Chao, X. Luo, and P. Hu, "Shielding collaborative learning: Mitigating poisoning attacks through client-side detection," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[56] M. S. Ozdayi, M. Kantarcioglu, and Y. R. Gel, "Defending against backdoors in federated learning with robust learning rate," *arXiv preprint arXiv:2007.03767*, 2020.

[57] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research*, vol. 12, no. 3, 2011.

[58] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[59] X. Zhang, M. M. Khalili, and M. Liu, "Improving the privacy and accuracy of ADMM-based distributed algorithms," in *International Conference on Machine Learning*, 2018.

[60] C. Li, P. Zhou, L. Xiong, Q. Wang, and T. Wang, "Differentially private distributed online learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 8, pp. 1440–1453, 2018.

[61] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," in *IEEE Symposium on Security and Privacy*, 2019, pp. 332–349.

[62] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *USENIX Security Symposium*, 2019, pp. 1895–1912.

[63] Y. Aono, T. Hayashi, L. Trieu Phong, and L. Wang, "Scalable and secure logistic regression via homomorphic encryption," in *The Sixth ACM Conference on Data and Application Security and Privacy*, 2016, pp. 142–144.

[64] M. Kim, Y. Song, S. Wang, Y. Xia, X. Jiang *et al.*, "Secure logistic regression based on homomorphic encryption: Design and evaluation," *JMIR medical informatics*, vol. 6, no. 2, p. e8805, 2018.

[65] 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 *ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[66] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng, "Privacy-preserving federated learning framework based on chained secure multiparty computing," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6178–6186, 2020.

[67] W. Gao, S. Guo, T. Zhang, H. Qiu, Y. Wen, and Y. Liu, "Privacy-preserving collaborative learning with automatic transformation search," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 114–123.

[68] X. Ma, Y. Zhou, L. Wang, and M. Miao, "Privacy-preserving byzantine-robust federated learning," *Computer Standards & Interfaces*, vol. 80, p. 103561, 2022.

[69] M. Grama, M. Musat, L. Muñoz-González, J. Passerat-Palmbach, D. Rueckert, and A. Alansary, "Robust aggregation for adaptive privacy preserving federated learning in healthcare," *arXiv preprint arXiv:2009.08294*, 2020.

[70] M. Naseri, J. Hayes, and E. De Cristofaro, "Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy," *arXiv preprint arXiv:2009.03561*, 2020.

[71] Y. Qi, M. S. Hossain, J. Nie, and X. Li, "Privacy-preserving blockchain-based federated learning for traffic flow prediction," *Future Generation Computer Systems*, vol. 117, pp. 328–337, 2021.

[72] X. Liu, H. Li, G. Xu, Z. Chen, X. Huang, and R. Lu, "Privacy-enhanced federated learning against poisoning adversaries," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4574–4588, 2021.

[73] L. Lyu, H. Yu, X. Ma, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and robustness in federated learning: Attacks and defenses," *arXiv preprint arXiv:2012.06337*, 2020.

[74] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," *arXiv preprint arXiv:2003.02133*, 2020.

[75] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021.

[76] D. Liu, Z. Yan, W. Ding, and M. Atiquzzaman, "A survey on secure data analytics in edge computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4946–4967, 2019.

[77] P. Vepakomma, T. Swedish, R. Raskar, O. Gupta, and A. Dubey, "No peek: A survey of private distributed deep learning," *arXiv preprint arXiv:1812.03288*, 2018.

[78] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[79] D. Enthoven and Z. Al-Ars, "An overview of federated deep learning privacy attacks and defensive strategies," *arXiv preprint arXiv:2004.04676*, 2020.

[80] Z. Yang, A. Gang, and W. U. Bajwa, "Adversary-resilient distributed and decentralized statistical inference and machine learning: An overview of recent advances under the byzantine threat model," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 146–159, 2020.

[81] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *International Conference on Machine Learning*, 2006, pp. 161–168.

[82] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[83] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[84] K. Chen, S. Guo, T. Zhang, X. Xie, and Y. Liu, "Stealing deep reinforcement learning models for fun and profit," in *ACM Asia Conference on Computer and Communications Security*, 2021, pp. 307–319.

[85] R. Battiti, "First-and second-order methods for learning: Between steepest descent and Newton's method," *Neural Computation*, vol. 4, no. 2, pp. 141–166, 1992.

[86] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[87] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1106–1114.

[88] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. W. Senior, P. A. Tucker, K. Yang, and A. Y. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1232–1240.

[89] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 5330–5340.

[90] P. Moritz, R. Nishihara, I. Stoica, and M. I. Jordan, "Sparknet: Training deep networks in spark," in *International Conference on Learning Representations*, 2016.

[91] Q. Ho, J. Cipar, H. Cui, S. Lee, J. K. Kim, P. B. Gibbons, G. A. Gibson, G. R. Ganger, and E. P. Xing, "More effective distributed ML via a stale synchronous parallel parameter server," in *Advances in Neural Information Processing Systems*, 2013, pp. 1223–1231.

[92] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *USENIX Security Symposium*, 2020, pp. 1605–1622.

[93] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[94] W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein, "Metapoison: Practical general-purpose clean-label data poisoning," *arXiv preprint arXiv:2004.00225*, 2020.

[95] Y. Ji, X. Zhang, and T. Wang, "Backdoor attacks against learning systems," in *IEEE Conference on Communications and Network Security*, 2017, pp. 1–9.

[96] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," 2017.

[97] T. D. Nguyen, P. Rieger, M. Miettinen, and A.-R. Sadeghi, "Poisoning attacks on federated learning-based iot intrusion detection system," in *Proc. Workshop Decentralized IoT Syst. Secur*, 2020, pp. 1–7.

[98] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 6103–6113.

[99] G. Sun, Y. Cong, J. Dong, Q. Wang, and J. Liu, "Data poisoning attacks on federated machine learning," *arXiv preprint arXiv:2004.10020*, 2020.

[100] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *European Symposium on Research in Computer Security*, 2020, pp. 480–501.

[101] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2019.

[102] S. Zhao, X. Ma, X. Zheng, J. Bailey, J. Chen, and Y.-G. Jiang, "Clean-label backdoor attacks on video recognition models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 443–14 452.

[103] S. Guo, T. Zhang, G. Xu, H. Yu, T. Xiang, and Y. Liu, "Topology-aware differential privacy for decentralized image classification," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[104] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: information leakage from collaborative deep learning," in *ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 603–618.

[105] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE Conference on Computer Communications*, 2019, pp. 2512–2520.

[106] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients–how easy is it to break privacy in federated learning?" *arXiv preprint arXiv:2003.14053*, 2020.

[107] M. Lam, G.-Y. Wei, D. Brooks, V. J. Reddi, and M. Mitzenmacher, "Gradient disaggregation: Breaking privacy in federated learning by reconstructing the user participant matrix," *arXiv preprint arXiv:2106.06089*, 2021.

[108] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[109] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *IEEE Symposium on Security and Privacy*, 2018, pp. 19–35.

[110] C. Zhu, W. R. Huang, A. Shafahi, H. Li, G. Taylor, C. Studer, and T. Goldstein, "Transferable clean-label poisoning attacks on deep neural nets," *arXiv preprint arXiv:1905.05897*, 2019.

[111] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attacks," 2018.

[112] 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*, 2017, pp. 1273–1282.

[113] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 463–16 472.

[114] S. Baluja, "Hiding images in plain sight: Deep steganography," *Advances in Neural Information Processing Systems*, vol. 30, pp. 2069–2079, 2017.

[115] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 657–672.

[116] M. Tancik, B. Mildenhall, and R. Ng, "Stegastamp: Invisible hyperlinks in physical photographs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2117–2126.

[117] K. Doan, Y. Lao, W. Zhao, and P. Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11 966–11 976.

[118] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, pp. 1–25, 2017.

[119] J. Tu, W. Liu, X. Mao, and X. Chen, "Variance reduced median-of-means estimator for byzantine-robust distributed inference," *Journal of Machine Learning Research*, vol. 22, no. 84, pp. 1–67, 2021.

[120] Y. Deng, F. Lyu, J. Ren, Y.-C. Chen, P. Yang, Y. Zhou, and Y. Zhang, "Fair: Quality-aware federated learning with precise user incentive and model aggregation," in *IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[121] Y.-R. Yang and W.-J. Li, "Basgd: Buffered asynchronous sgd for byzantine learning," in *International Conference on Machine Learning*, 2021, pp. 11 751–11 761.

[122] Y. Mao, X. Yuan, X. Zhao, and S. Zhong, "Romoa: Robust model aggregation for the resistance of federated learning to model poisoning attacks," in *European Symposium on Research in Computer Security*, 2021, pp. 476–496.

[123] E. M. El-Mhamdi, S. Farhadkhani, R. Guerraoui, A. Guirguis, L.-N. Hoang, and S. Rouault, "Collaborative learning in the jungle (decentralized, byzantine, heterogeneous, asynchronous and nonconvex learning)," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[124] C. Ma, J. Li, M. Ding, K. Wei, W. Chen, and H. V. Poor, "Federated learning with unreliable clients: Performance analysis and mechanism design," *IEEE Internet of Things Journal*, 2021.

[125] S. P. Karimireddy, L. He, and M. Jaggi, "Learning from history for byzantine robust optimization," in *International Conference on Machine Learning*, 2021, pp. 5311–5319.

[126] J. Feng, H. Xu, and S. Mannor, "Distributed robust learning," *arXiv preprint arXiv:1409.5937*, 2014.

[127] C. Xie, O. Koyejo, and I. Gupta, "Generalized byzantine-tolerant sgd," *arXiv preprint arXiv:1802.10116*, 2018.

[128] X. Fan, Y. Ma, Z. Dai, W. Jing, C. Tan, and B. K. H. Low, "Fault-tolerant federated reinforcement learning with theoretical guarantee," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[129] E. M. El Mhamdi, R. Guerraoui, and S. L. A. Rouault, "Distributed momentum for byzantine-resilient stochastic gradient descent," in *9th International Conference on Learning Representations (ICLR)*, no. CONF, 2021.

[130] Y. Gao, B. G. Doan, Z. Zhang, S. Ma, A. Fu, S. Nepal, and H. Kim, "Backdoor attacks and countermeasures on deep learning: A comprehensive review," *arXiv preprint arXiv:2007.10760*, 2020.

[131] H. Qiu, Y. Zeng, S. Guo, T. Zhang, M. Qiu, and B. Thuraisingham, "Deepsweep: An evaluation framework for mitigating dnn backdoor attacks using data augmentation," in *ACM Asia Conference on Computer and Communications Security*, 2021, pp. 363–377.

[132] S. Li, S. Ma, M. Xue, and B. Z. H. Zhao, "Deep learning backdoors," *arXiv preprint arXiv:2007.08273*, 2020.

[133] S. Andreina, G. A. Marson, H. Möllering, and G. Karame, "Baffle: Backdoor detection via feedback-based federated learning," in *International Conference on Distributed Computing Systems*, 2021, pp. 852–863.

[134] J. Sun, A. Li, L. DiValentin, A. Hassanzadeh, Y. Chen, and H. Li, "Fl-wbc: Enhancing robustness against model poisoning attacks in federated learning from a client perspective," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[135] C. Wu, X. Yang, S. Zhu, and P. Mitra, "Mitigating backdoor attacks in federated learning," *arXiv preprint arXiv:2011.01767*, 2020.

[136] 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 *IEEE Symposium on Security and Privacy*, 2019, pp. 739–753.

[137] J. Zhang, J. Zhang, J. Chen, and S. Yu, "Gan enhanced membership inference: A passive local attack in federated learning," in *IEEE International Conference on Communications*, 2020.

[138] X. Yuan, X. Ma, L. Zhang, Y. Fang, and D. Wu, "Beyond class-level privacy leakage: Breaking record-level privacy in federated learning," *IEEE Internet of Things Journal*, 2021.

[139] M. Song, Z. Wang, Z. Zhang, Y. Song, Q. Wang, J. Ren, and H. Qi, "Analyzing user-level privacy attack against federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2430–2444, 2020.

[140] W. Zhang, S. Tople, and O. Ohrimenko, "Leakage of dataset properties in multi-party machine learning," in *USENIX Security Symposium*, 2021, pp. 2687–2704.

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

[142] T. Dang, O. Thakkar, S. Ramaswamy, R. Mathews, P. Chin, and F. Beaufays, "Revealing and protecting labels in distributed training," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[143] X. Jin, P.-Y. Chen, C.-Y. Hsu, C.-M. Yu, and T. Chen, "Cafe: Catastrophic data leakage in vertical federated learning," *arXiv preprint arXiv:2110.15122*, 2021.

[144] C. Fu, X. Zhang, S. Ji, J. Chen, J. Wu, S. Guo, J. Zhou, A. X. Liu, and T. Wang, "Label inference attacks against vertical federated learning."

[145] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference," in *Computer Security Applications Conference*, 2019, pp. 148–162.

[146] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy*, 2017, pp. 3–18.

[147] 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.

[148] Y. Long, V. Bindschaedler, L. Wang, D. Bu, X. Wang, H. Tang, C. A. Gunter, and K. Chen, "Understanding membership inferences on well-generalized learning models," *arXiv preprint arXiv:1802.04889*, 2018.

[149] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro, "LOGAN: Membership inference attacks against generative models," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 1, pp. 133–152, 2019.

[150] S. Chen, M. Kahla, R. Jia, and G.-J. Qi, "Knowledge-enriched distributional model inversion attacks," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 178–16 187.

[151] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2006, pp. 486–503.

[152] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *IEEE Annual Symposium on Foundations of Computer Science*, 2010, pp. 51–60.

[153] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," *arXiv preprint arXiv:1812.00984*, 2018.

[154] N. Hynes, R. Cheng, and D. Song, "Efficient deep learning on multi-source private data," *arXiv preprint arXiv:1807.06689*, 2018.

[155] Y. Kang, Y. Liu, and W. Wang, "Weighted distributed differential privacy ERM: Convex and non-convex," *arXiv preprint arXiv:1910.10308*, 2019.

[156] I. Mironov, "Rényi differential privacy," in *IEEE Computer Security Foundations Symposium*, 2017, pp. 263–275.

[157] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: An application of human behavior prediction," in *AAAI Conference on Artificial Intelligence*, 2016, pp. 1309–1316.

[158] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1310–1321.

[159] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, 2020.

[160] M. Park, J. Foulds, K. Choudhary, and M. Welling, "DP-EM: Differentially private expectation maximization," in *Artificial Intelligence and Statistics*, 2017, pp. 896–904.

[161] B. Jayaraman, L. Wang, D. Evans, and Q. Gu, "Distributed learning without distress: Privacy-preserving empirical risk minimization," in *Advances in Neural Information Processing Systems*, 2018, pp. 6343–6354.

[162] C. Dwork and G. N. Rothblum, "Concentrated differential privacy," *arXiv preprint arXiv:1603.01887*, 2016.

[163] R. Han, D. Li, J. Ouyang, C. H. Liu, G. Wang, D. Wu, and L. Y. Chen, "Accurate differentially private deep learning on the edge," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 9, pp. 2231–2247, 2021.

[164] K. Wei, J. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Transactions on Mobile Computing*, 2021.

[165] W. Wei, L. Liu, Y. Wut, G. Su, and A. Iyengar, "Gradient-leakage resilient federated learning," in *International Conference on Distributed Computing Systems*, 2021, pp. 797–807.

[166] P. Sun, H. Che, Z. Wang, Y. Wang, T. Wang, L. Wu, and H. Shao, "Pain-fl: Personalized privacy-preserving incentive for federated learning," *IEEE Journal on Selected Areas in Communications*, 2021.

[167] Z. Xiong, Z. Cai, D. Takabi, and W. Li, "Privacy threat and defense for federated learning with non-iid data in aiot," *IEEE Transactions on Industrial Informatics*, 2021.

[168] T. P. Le, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, no. 99, pp. 1–1, 2017.

[169] G. Xu, H. Li, Y. Zhang, S. Xu, J. Ning, and R. Deng, "Privacy-preserving federated deep learning with irregular users," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2020.

[170] Q. Zhang, C. Xin, and H. Wu, "Gala: Greedy computation for linear algebra in privacy-preserved neural networks," in *Network and Distributed System Security Symposium*, 2021.

[171] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "Batchcrypt: Efficient homomorphic encryption for cross-silo federated learning," in *USENIX Annual Technical Conference*, 2020, pp. 493–506.

[172] D. Froelicher, J. R. Troncoso-Pastoriza, A. Pyrgelis, S. Sav, J. S. Sousa, J.-P. Bossuat, and J.-P. Hubaux, "Scalable privacy-preserving distributed learning," *Proceedings on Privacy Enhancing Technologies*, vol. 2, pp. 323–347, 2021.

[173] D. Stripelis, H. Saleem, T. Ghai, N. Dhinagar, U. Gupta, C. Anastasiou, G. V. Steeg, S. Ravi, M. Naveed, P. M. Thompson *et al.*, "Secure neuroimaging analysis using federated learning with homomorphic encryption," *arXiv preprint arXiv:2108.03437*, 2021.

[174] J. H. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova, "Secure single-server aggregation with (poly) logarithmic overhead," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1253–1269.

[175] Y. Li, H. Li, G. Xu, T. Xiang, X. Huang, and R. Lu, "Toward secure and privacy-preserving distributed deep learning in fog-cloud computing," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 460–11 472, 2020.

[176] Q. Zhao, C. Zhao, S. Cui, S. Jing, and Z. Chen, "PrivateDL: Privacy-preserving collaborative deep learning against leakage from gradient sharing," *International Journal of Intelligent Systems*, 2020.

[177] L. Fan, K. W. Ng, C. Ju, T. Zhang, C. Liu, C. S. Chan, and Q. Yang, "Rethinking privacy preserving deep learning: How to evaluate and thwart privacy attacks," *arXiv preprint arXiv:2006.11601*, 2020.

[178] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, "Evaluating gradient inversion attacks and defenses in federated learning," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[179] L. Lyu, "Dp-signsgd: When efficiency meets privacy and robustness," in *International Conference on Acoustics, Speech and Signal Processing*, 2021, pp. 3070–3074.

[180] Y. Dong, X. Chen, K. Li, D. Wang, and S. Zeng, "Flod: Oblivious defender for private byzantine-robust federated learning with dishonest-majority," *Cryptology ePrint Archive*, 2021.

[181] J. Domingo-Ferrer, A. Blanco-Justicia, J. Manjón, and D. Sánchez, "Secure and privacy-preserving federated learning via co-utility," *IEEE Internet of Things Journal*, 2021.

[182] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[183] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy*, 2017, pp. 39–57.

[184] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[185] T. Xiang, H. Liu, S. Guo, T. Zhang, and X. Liao, "Local black-box adversarial attacks: A query efficient approach," *arXiv preprint arXiv:2101.01032*, 2021.

[186] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!" *arXiv preprint arXiv:1904.12843*, 2019.

[187] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *arXiv preprint arXiv:1705.07204*, 2017.

[188] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," *arXiv preprint arXiv:2001.03994*, 2020.

[189] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, "Recent advances in adversarial training for adversarial robustness," *arXiv preprint arXiv:2102.01356*, 2021.

[190] J. Hong, H. Wang, Z. Wang, and J. Zhou, "Federated robustness propagation: Sharing adversarial robustness in federated learning," *arXiv preprint arXiv:2106.10196*, 2021.

[191] Y. Zhou, J. Wu, and J. He, "Adversarially robust federated learning for neural networks," 2020.

[192] D. Shah, P. Dube, S. Chakraborty, and A. Verma, "Adversarial training in communication constrained federated learning," *arXiv preprint arXiv:2103.01319*, 2021.

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

[194] M. Weber, X. Xu, B. Karlaš, C. Zhang, and B. Li, "Rab: Provable robustness against backdoor attacks," *arXiv preprint arXiv:2003.08904*, 2020.

[195] B. Wang, X. Cao, N. Z. Gong *et al.*, "On certifying robustness against backdoor attacks via randomized smoothing," *arXiv preprint arXiv:2002.11750*, 2020.

[196] C. Xie, M. Chen, P.-Y. Chen, and B. Li, "Crfl: Certifiably robust federated learning against backdoor attacks," *arXiv preprint arXiv:2106.08283*, 2021.