Review of Privacy in ML with a focus on federated learning and Differential privacy

Emran Altamimi
dept. Computer science
Qatar University
Doha, qatar

***Abstract*—Privacy is a growing concern across different fields, and with the rise of machine learning, there has been a growing interest to harness the power of it in a privacy preserving manner. Common attacks on machine learning are discussed and the methods to protect against them. The limitations and future research directions of privacy preserving machine learning is discussed. Federated learning and differential privacy are then highlighted in details, with their importance and how they aid in privacy for machine learning.**

***Index Terms*—Machine learning, privacy preserving, federated learning, differential privacy, Attacks**

I. Introduction

Machine learning is becoming more widely used in a multitude of applications. deep - learning (DL), demonstrate big progress in performance, particularly in areas like machine vision, nlp, and voice or sound identification [1]. Arising federated learning is a cooperative machine learning method that allows a strong model to be trained even though the data is distributed across numerous decentralized machines [2] . In a multitude of areas, such as health services, IoT, and intelligent systems, among others,federated learning has demonstrated its ability to deliver on promise [3]. Despite the fact that such systems had already demonstrated big progress in AI- or machine-learning-driven solutions, they nevertheless encounter many challenges, such like:

- a scarcity of computationally powerful resources
- large amounts of data are available for training the model, which is computationally inefficient.

ML systems generally perform better when they have a large amount of training data and computationally intensive resources to sustain both of building the model and predictive stages. Large memory storage capacity and high-performance CPUs and GPUs are just a few of the computing resources needed to support machine learning (ML). To meet this demand, commercial infrastructure service providers such as Amazon, Microsoft, Google, and IBM have invested heavily in constructing IaaS and Machine-Learning-as-a-Service (MLaaS) platforms with rental fees. limited resource users can use ML-related cloud computing services to maintain and build their models first, and then use their applications directly to provide data analytics and predictive services.

There is a further dilemma for machine learning techniques: access to large amounts of data is required. To get the best out of an artificial intelligence model, it is necessary to collect a large amount of training data though in some case scenarios from different providers. Because of this, it is important to note that there are serious privacy concerns regarding the gathering and use of data, and also the creation and use ML models. For example, latest data cyberattacks have heightened concerns about the privacy of large-scale data collection and use [4]. It is also possible to infer confidential details using various inference attacks, like membership attacks, on an ML model that can be exploited by an attacker [5]. Other attacks examples are, model inversion attacks [6], property inference attacks [7], and privacy leakage during the communication of gradients in federated learning [8]. as an example, an adversary could be able to determine if a specific person's data was used in the building of an AIDs-related classifier. Regulators like HIPPA and the GDPR are ever more restricting access to and use of confidentiality data.As a result, the use of ML models in practical uses has seen significant challenges. As a result Privacy preserving machine learning strategies are needed to address growing privacy risks pertaining with the use of ML in uses that collect and manage sensitive personal data, such as digital hospital records, geolocation, etc. Recently, more attention has been paid to privacy preserving machine learning that incorporates established obfuscation methods into the machine learning pipeline or developing innovative techniques and structures for machine learning systems. currently only some aspects of privacy are addressed in each existing approach. This review paper will discuss current approaches and their limitations.

Their are many review papers in the topics of privacy preserving machine learning, in [9] the authors give a comprehensive review of federated learning based on a 5 dimensional privacy concern (trust assumptions, active or passive adversaries, where in the ML pipeline the method is implemented, at which stage (gradient and weight update, or the final model), and the attack assumptions). In [10] all aspects of privacy and machine learning was reviewed, including how privacy can be enhanced with machine learning and how machine learning can be used as a tool to detriment privacy. [11] gave a high level overview of privacy preserving threats and their respective solutions. In [12] the authors reviewed how differential privacy is able to conceal the locations of users and their applications in services that are location-based. In

[13] the authors survey federated learning from a system's point of view, as well as the applications, limitations and future research direction. [14] provides a survey on differential privacy in the context of Deep learning.

Our contribution comes in providing a comprehensive review of all privacy preserving techniques in machine learning along with a comparison and limitations of each. With a focus on federated learning and differential privacy their applications, limitations and future direction.

The rest of the paper is organized as follows, section 2 will give a background knowledge on privacy and current attack on machine learning. section 3 will review the privacy preserving solutions in machine learning. section 4 and 5 will review differential privacy and federated learning in the privacy preserving machine learning context, along with their applications and challenges. section 6 will conclude the paper and summarize the challenges of privacy preserving machine learning with the future research direction.

II. Background

The authors of [15] proposed a PGU triad to systematize the knowledge in privacy preserving ML which refers to Phase, Guarantee, and technical Utility. The phase aspect refers to when the Privacy preserving technique is used in the machine learning phase, which can be in the data aggregation phase or gathering the data, the training phase, the prediction phase, or the inquiry phase. guarantee refers to how much does a system protect the data for various attacks and what are the assumptions assumed for the model, for example is the aggregator of the data trusted, or if the model is hosted on a centralised trusted party. The guarantees are viewed from two aspects according to a pervasive risk model, and trust presumptions: object-oriented and pipeline-oriented. object-oriented solutions aim to protect input privacy by preventing the leakage/exposure of private information from training or inference data samples. The other type of solutions protect model privacy by mitigating privacy disclosure from the learned model. pipeline solutions are concerned with the privacy-preserving functionality associated with an entity or collection of entities in the pipeline of a machine learning solution. These can be viewed as trusting the computational facility with the data and for them to protect the data and the model leakage, or not trusting the computational facility in a particular or all phases of the ML pipeline. Utility; because privacy comes at the expense of accuracy and performance of the ML models, utility specifies if the solution will deteriorate the model and by how much, and if the model is still applicable for the use case after adopting the solution, other aspects include scalability, communication, and computational complexity. The utility of the model is strongly affected by the: data releasing, data analysis, architecture-based, and hybrid approaches of the privacy preserving machine learning approaches as these are the most that affect the performance.

A. Threats on machine learning

In the field of Ai, privacy is nuanced. securing training data, models, model parameters, and protection against inference attacks are all examples of privacy in machine learning. This subsection describes the meaning of privacy breaches in Machine Learning [16].

When a person's anonymity is jeopardized, for example, by membership inference, privacy is said to be violated. Also the model is considered stolen when the adversary's infers the model by firing multiple queries to the model in a MLaaS model. There are three major classifications: The first category is member inference from population: In this category the training dataset can be compromised by using the model itself. The three following attacks allow the adversary to do that. In statistical disclosure an adversary can learn more about the input by using the model and the model output and reveal unintended information. For that reason, Statistical Disclosure Control (SDC) is utilised in that case so no information is revealed about an example of the training data through the use of the model. Unfortunately, it is discovered that no ML model is capable of achieving it [17]. In model inversion is when the attacker is able to reconstruct the raw data that was used in the training, for example a model might classify an input to dogs and cats, the attacker might reconstruct the raw data and generate pictures of dogs and cats. The last type in member inference category is revealing the class representatives. Model inversion generalization can be used by the adversary to construct models that represent the classes in the training dataset. The second category is member inference from the training data, the goal of the attacker is to reveal information about a person's data that was used during training. There are two types in this category first is the membership inference, where the attacker has a person's information and wants to confirm if it was used in the training data. The second one is the property inference which is what was described earlier where specific attributes and features about the dataset is revealed. The third and final category is the model Parameter inference. There are two types in this category first of all is the Model extraction inference which is similar to the concept of reverse engineering, in which an attacker can learn about the hidden architecture of the model and steal it. The second one is the functionality stealing where the attacker can generate a similar workable copy.

B. Attacks on machine learning

This section will review the literature and categorize common Machine Learning attacks and their effect on privacy. Two broad categories exist, explicit and implicit attacks. Explicit attacks are attacks that explicitly breach the training data, either by accessing, exposing, or leaking them. In contrast, implicit attacks are when the attacker can still learn valuable insights about the training data without explicitly stealing it. for explicit attacks and in MLaaS settings, it allows for assaults that expose private data over the cloud service and do not deal transparently with the lifespan of the data collected from customers. These kind of assaults put the privacy of a

person at risk and may have far-reaching ramifications. Since these attacks come primarily within the domain of information security attacks, it is out of this paper's scope. Implicit attacks fall into one of five categories: property inference, model inversion, membership and parameter inference attacks as well as hyper parameter inference. Another dimension of categorization is if the attacker has access to the model, or if the attacker has access to the confidence values of the model. Also some attacks require a white-box approach in which the model is completely known to the attacker, in contrast the black-box approach the attacker does not know the model, parameters or architecture of the model and the attacks are only based on the classification or prediction of the model. [18]

- Training time attacks

Training time attacks are attacks that occur during training the model. The adversary is can either be an active or passive participant in the training. (1) Poison attack In these assaults, the model output is either directly manipulated by the adversary, or the output is damaged. For example, a clean-label data poisoning attack [19] assumes the adversary will not alter the label of any training data, so the poisoning of data samples must be imperceptible; a dirty-label data poisoning attack [20] assumes the adversary will introduce a number of data samples with incorrect labels into the training set; and a model poisoning attack [21] assumes the adversary will introduce data samples with the wrong labels into the training set. Training data poisoning is just the beginning of the assault; algorithm poisoning and model poisoning are also possibilities.

- Inference time attack

After training is complete, these assaults occur while the model is being inferred or tested. As an opponent may query and get results from the model, here is where an adversary can exploit a security and privacy weakness to deduce the model's parameters or membership information of an individual. Model inversion, membership inference, model theft, and property inference are all types of attacks that fall under this category. Fredrikson et al. [22] discovered an attack of model inversion by obtaining API access to the pharmaceutical dosage prediction model and effectively disclosing sensitive information such as the patient's age, height, and genetic metadata, with only black-box access. Again, with only black-box access, Fredrikson et al. [23] employed a similar method on neural network-based learning and demonstrated that they could learn the most delicate attributes in a training dataset by feeding the model with properly constructed input. (1) Membership inference In membership inference, specifically the Membership Inference Attack (MIA) the attacker can learn weather a person's data has been used to train the model. In light of the ML model's increased emphasis on confidentiality, this attack poses an even greater risk to individuals' personal information. MIA was first mentioned in [24], where by querying the target model with black-box access, an attacker is able to get the probability of each attribute for a particular data point. which later is used to infer if the example was in the training dataset. The attack works by generating synthetic, noisy data, and predicting their distribution that they belong to a particular class. After creating the labelled data (i.e. data examples labelled with the probability that they belong to which class), several models are built from it to increase their accuracy. The labelled data generated is called shadow data.The models built from this data is called shadow models. Later on, the attack model is trained using the shadow models, which can tell if a data sample was part of the training or not. For each given data point, an adversary compares the target model to the probability prediction of the attack model (the model built from the shadow models); a high value indicate that membership probabilities are likewise high. If the model hasn't previously seen the example, the probability will be low; otherwise, the probability will be larger, and hence membership. MIA effectiveness is heavily correlated with over-fitting, as under-fitting will give more generalised predictions and the probabilities will be less meaningful. As a result, MIA are reported to be more effective against deep neural network as they typically overfit the data [25]. (2) Model stealing In model stealing ML models are considered an intellectual property due to the enormous investing put into them to collect and aggregate the data, engineering the algorithm and hyper-parameters. therefore stealing the model's parameters and architecture is viewed as a privacy breach. Stealing the model, also means having a white-box access to it, where many attacks can be further done to compromise the information about the training-set. (3) Property inference In Property Inference attacks are attacks that uncover some patterns of the model. For example uncovering how much the model is over-fitting the data, specifically unintended over-fitting as in the case of [26], which can be a step to before conducting an MIA attack.

III. Privacy preserving machine learning

state-of-the-art mechanisms for Privacy-Preserving Machine Learning are reviewed in this section. A typical machine learning process is divided into three major groups, each of which has privacy-preserving methods. Methods for collecting or combining data that preserve privacy fall under the first category. In the second category, stratifies to maintain confidentiality throughout the essential model training phase are devised. The model and training data are protected by these techniques during the training phase of ML. lastly, after the deployment of the model Privacy-Preserving techniques that preserve model users' privacy are included in this third and final category. Finally the Privacy can also be preserved if the environment and execution enhance privacy, though they are not Privacy-Preserving by themselves.

A. Data aggregation

This section discusses privacy concerns associated with data collection and aggregation throughout the Machine Learning process. Maintaining privacy while exchanging data is, perhaps, the most critical strategy. To maintain privacy, the approaches discussed in this section make extensive use of

anonymization, perturbation , and encryption. Additionally, the approaches may be categorized into two groupings depending on their context-awareness while exchanging data. Context-free techniques, like differential privacy, remain unaware of the specific context or purpose for which the data is exchanged. Context-aware approaches, such as information-theoretic privacy, on the other hand, may perform better in achieving privacy since they are aware of the context and purpose of data use [27]. In the process of anonymization, Private details like name, address, and other identifiers are deleted from the dataset before it is shared with the collaboration engine. Nevertheless, sophisticated de-anonymization methods may disrupt anonymization, making this less effective and less useful in privacy protection. Netflix, for instance, released its data of user ratings in 2007 as part of a contest to see whether anybody could surpass its collaborative filtering system. Researchers seem to have been able to retrieve 99 percent of the private details from the data despite the fact that no personally identifiable information was made public. Researchers have been able to recover the material that had been deleted by utilizing the data from the Internet Movie Database (IMDB). In homomorphic encryption approaches, information may be secured before being published for collaborative ML in order to protect privacy. As a method, the strategies leverage semantic security or probabilistic encryption, which, although theoretically breakable, is presently infeasible due to computational limitations. This method makes use of sophisticated encryption and requires a significant amount of computational power. As a result, it is an unfavorable option for use in production because of its high computational requirements.

- A framework to train on encrypted data

In [28], the authors proposed a new framework to train neural networks over encrypted data with homomorphic encryption called CryptoDL. CryptoDL have the threat assumptions of a client-server MLaaS, where the data is provided by the client, and the server builds the machine learning model providing the predictions required by the client. The key distinction here is that the client provides encrypted data to the server, which make it feasible to build a machine learning model without learning anything about the data. As a result the trained model's parameters are encrypted as well with the public key of the client, making the model only use-able by the client. During prediction, the client also predicts encrypted data, and the predictions are encrypted as well, making both the data and predictions inaccessible by the server. The authors in this work do not operate on Fully homomorphic schemes, and utilise leveled homomorphic schemes to have a more efficient and practical solution. To enable such schemes in neural networks the activation functions and complex mathematics of neural networks are approximated by polynomials which only use addition and multiplication. The limitations are that polynomial functions must be restricted to low degree polynomials in order to considered practical. The authors experimented with two orthogonal systems of polynomials, for the sigmoid function within a symmetric interval. The authors experimented with four different degree of polynomials (3,5,7,9)and and four different intervals $-10^i, 10^i$ for i from 1 to 4. Some observations were found, smaller intervals yield better approximations, which means datasets with less variance can fit better in smaller intervals and hence better approximations. The second observation is for higher degree approximations they approximate better however they are not very desirable for performance, but because the coefficients become very small for high degree polynomials, small coefficients can be dropped to zero improving performance. The framework was compared with the state of the are in homomorphic encryption, and SMC based approaches, their work outperforms CryptoNets a popular homorphic appraoch for approximating neural networks function in the training phase in accuracy (97.27 vs 98.15 for approximating the Tanh, and 99.10 vs 99.15 for approximating the sigmoid funciton), the numbers of classifications per hour (163 840 vs 51 739) and training time (570 seconds vs 320). Several limitations come to mind regarding the framework, firstly the work only considers continuous functions for polynomial approximation, which doesn't take into consideration many other more popular activation functions in neural network such as the Relu.

B. Training phase

During the training phase of Machine Learning, privacy-preserving approaches are discussed in this section. The privacy of the data may be protected if the training is done using encrypted data. the methods are either encryption or differential privacy. A review of studies on private training found that the following three types of strategies were most often used: There are three main aspects to this: 1) Secure multi-party computation, 2) Homomorphic encryption, and 3) Differential privacy.

In homomorphic encryption the functions are anonymised and privacy of both the data and functions are anonymised as it allows operations to be done on encrypted data. Encrypted data may be sent to a central server where it is used to train machine learning model as if it were unencrypted and then decrypted by parties who know how to decode it. Non-linear functions can't be handled by homomorphic encryption algorithms, which is a big drawback. Simple classifiers and algorithms with no division should be used instead of more complicated ones like a Neural Network. However, as mentioned above, Hesamifard et al. [28] attempted building a Deep Learning model with homomorphic encryption with the framework called CryptoDL, which can also be considered an approach during training. They came up with approaches for approximating the activation functions of neural networks (Tanh, Sigmoid, and ReLU) using low-degree polynomials to create efficient homomorphic encryption systems. A convolution neural network (CNN) was trained using the approximation polynomial functions, and then evaluated using a character recognition task. Secure Multi-party Computation is a sub-field of cryptography. In which multiple entities (adversary/semi-honest/honest) will work together to tackle privacy concerns using collaborative computing [29]. Participants in this network are unaware of

each other's data since no one can see further than one encoded portion of the entire data while doing joint computations. Insofar as at least one partner inside the collaborative computing network is trust-worthy, SMPC protects the privacy of data owners. For example, knowing the maximum salary between co-workers without any person revealing their own salary. SMPC, however, can't handle complex computing functions well and is very computationally expensive [30], [31]. SMPC in the context of ML, confidentiality is enhanced by separating computing and training on non-colluding servers while diverse parties contribute data for building neural network models, logistic regression, and linear as well as inference. The need that all parties and servers be up at all times is one of SMPC's flaws; which leads to a high communication overhead. Traditionally, SMPC is built on an existing algorithm, however, as suggested in [32] a secure computation protocol and an optimized algorithm can be combined for a more efficient scheme.

- SecureML

One of the most popular work during the training phase is SecureML [33]. The authors assume a two server model, where the user splits the data in half and sends each portion to a server, because no server has full access to the data, nothing can be learnt from it. The two servers then perform two party computation, which more efficient than multi-party computation. The first protocol, is privacy preserving linear regression in which a new protocol is implemented to allow the conventional secret sharing multi party computation protocols to work on decimal numbers. secret sharing works where a message can be divided into several parts and shared in which the original message can not be reconstructed by any number of adversaries in the The reader is referred to [34] for more information on secret sharing in a two party computation. The protocol is summarized in four steps below:

1) The data points (x, y) are secretly shared by the users.
2) The weights of the model is initialized and secretly shared.
3) Stochastic Gradient Decent runs on pre-computed multiplication triplets
4) Truncate the shares after multiplication (as shown in fig 1)

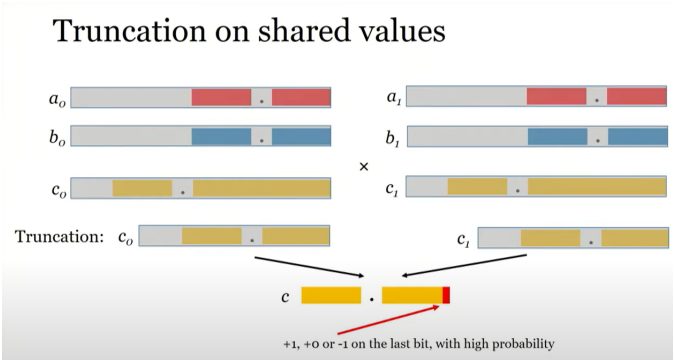

Fig. 1. Adapted decimal multiplication for secret sharing

For logistic regression, the algorithm is similar except for the introduction of the sigmoid function which needs to be addressed to work in a privacy preserving machine learning, the authors approach is to actually come up with another function that have the same properties as the sigmoid function instead of approximating it. The protocol runs as the following

1) Run the linear regression protocol.
2) The protocol switches to garbled circuit for the activation function (where the function is described in a Boolean manner).
3) The protocol switches back to arithmetic secret sharing.

The evaluation was done on 1000 records, 500 features each. The protocol runs on two stages, offline and data independent to generate multiplication triplet and the online phase (or training phase, and they performed order of magnitudes faster than previous work. The client aided-triplets (can not collude with any of the two servers) will significantly decrease the offline phase time. Under this security model the authors had no previous work to compare with.

C. Inference phase

Inference privacy Inference privacy is concerned with maintaining the confidentiality of a system that provides Inference-as-a-Service or MLaaS. according to most solutions differential privacy is considered a de facto solution for training, but homomorphic SMPC and encryption are chosen for inference. A major reason for this is because the computing costs and difficulties associated with encryption technologies make them unsuitable for long-term training programs. On the latter, inference tasks are comparatively faster, since the network has been already trained, implementing encryption is significantly easier. Adding perturbation methods reduces performance and necessitates exact execution; as a result, using them when predicting seems to be disadvantageous. The main idea is that the model is predicting encrypted data, and the results is encrypted as well.

Also, due to the fact that cryptographic schemes are done on the data, they must be in the integer format as cryptographic schemes are in the integer group, this leads to loss of precision on the data, and a compromise has to be made.

D. Privacy enhanced execution environment

This section briefly discusses privacy enhancing techniques such as trusted execution environments, split learning, and federated learning. Nevertheless, such strategies for strengthening confidentiality must be used in conjunction with the previously mentioned privacy-preserving procedures. Federated Learning is a collaborative setting for training machine learning models. According to federated learning's main concept, input data does not leave the client's environment; rather, several users collaborate to fit the classifier utilizing their data under the control of a centralized server. federated learning is a case of distributed learning that is done securely, hence eliminating the privacy problems associated with centralized Machine Learning systems. Split learning is the process of dividing a DNN into numerous portions, every portion is on a distinct set

of clients for training. the training dataset may be stored on a centralized or distributed manner. However, noone engaged in building the DNN is able to ”read” the data of the other clients. In order to train a deep neural network, techniques are done to encode the data into a new space. Because the model's architecture is divided into numerous parts, each of which is trained separately, the training is done by transferring the weights of each section's final layer to the one before it, and the gradients are sent back allowing the network to learn from its mistakes through back propagation. As a result, no confidential data transmitted; instead, only the weights of a single-final layer (i.e. cut layer) of every portion are passed on to the following client.

IV. Differential privacy

Before delving into differential privacy there are two key definitions to understand; linking attacks, and K-anonymity. Linking attacks is fairly simple, suppose we have a dataset that had all personal identifiers removed from it, and another dataset (called auxiliary data) has some personal identifiers in it, and it shares some attributes with the original dataset. Then a simple comparison and merging of the two datasets, would identify the person. For example, I know Alice's birthday, and in the dataset the only two personal information are the birthdays. Then if I only found a single record in the dataset, know other information about Alice. Now suppose after I did the linking attack and I got four matching columns, it becomes more difficult to pinpoint Alice from the four. If any linking attack I did narrowed down the entries to a minimum of four no matter what auxiliary information I have, then the dataset is said to 4-anonymity. Expanding this concept beyond this example, which means if an adversary could not narrow that dataset beyond K entries it is said to have K anonymity. In other words, K-anonymity aims to quantify our instinct that a bit of additional information must not limit down the range of potential entries for a person ”excessively.” k-Anonymity is meant to guarantee that any person could ”fit into the crowd.” The naive approach to this, is to generalize the data, remove outliers, and check every entry groups to make sure it satisfies K-anonymity. However, there are severe computational inhibitions, such that algorithms take up to O(n^2) to check for K-anonymity, and optimally generalizing the dataset is NP-hard. Differential privacy, attempts to solve the issue of linkage attacks, however, it is important to note that differential privacy is a property just like K-anonymity, but differential privacy an algorithm's property unlike k-anonymity which pertains to the data. Define equation The idea is if a single record was removed from the data set x to give me the dataset x' the function is randomized in a way that it will give me the same output, and the data set is considered identical to the function. Back to the example above, an attacker can not limit the data set to a single entry (Alice) after entering her birthday, because the dataset is considered identical for the function, with and without Alice, and therefore the attacker can not be sure that Alice exists in the data set. Now depending on the value of epsilon the function could provide identical results with different inputs or very distinct result with neighboring inputs, the latter of-course does not provide privacy and is useless, while the former provide higher privacy at the expense. It is common practice to respond to individual inquiries by using differential privacy. For example a data query without differential privacy would be all the users that have a specific birthday, which results in number N. Now the idea of differential privacy is to add noise to the query such that it provides privacy without disrupting the underlying distribution. Laplace function or mechanism in the context of differential privacy is a very common method to achieve this as shown below:

The notion of differential privacy provides three fundamental features of deferentially private functions or mechanisms. These characteristics will aid in the creation of usable algorithms that fulfill differential privacy requirements while also ensuring the accuracy of the results they produce.

The first property is sequential composition, in which if we have two mechanisms that satisfy e1, e2 differential privacy respectively, then their composition must at-least satisfy e1+e2 differential privacy. The figure 2 below shows how two deferentially private mechanisms executed sequentially. in other words a mechanism with e1 + e2 will always be as private or less private than two sequentially private mechanisms. Note that the higher the e the lower the privacy as mentioned above. The second property is parallel composition in which

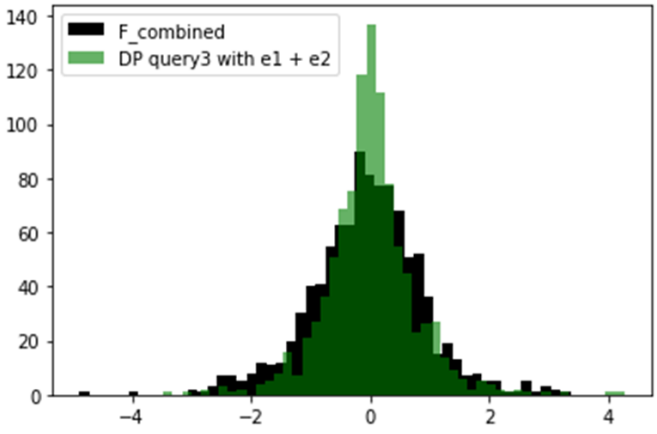

Fig. 2. two deferentially private mechanisms executed sequentially

if the data is split into several disjoint distinct portions then running the mechanism on all of them will still result in the same epsilon, i.e. there will be no loss of privacy. The post-processing property guarantees that no post-processing done on the output of a function that satisfies differential privacy will affect the privacy. In instance, post-processing may be used to minimize the noise or enhance the query output from the mechanism such as removing values that don't make sense. The use of post-processing to minimize noise and increase the accuracy of deferentially private algorithms is really rather common. Additionally, the post-processing attribute gives resilience to assaults on privacy that rely on auxiliary data, for

example a linking attack discussed earlier will be limited in its effectiveness by the privacy parameter epsilon. There are two types of differential privacy: global and localized. Both ensures a single user's differential privacy needs within the privacy parameter epsilon, but they're applied in different situations. In the case of Local differential privacy, a centralized differential privacy server is not utilised instead before sharing, users add deferentially private randomization. The premise that the data collector is untrustworthy and relevant to data aggregation privacy preserving machine learning methods prompted the development of local differential privacy. Solutions based on local differential privacy are commonly used to address privacy concerns..

A. Differential privacy in machine learning

Differential privacy plays a big role in preserving privacy in big data, as it allows institutions to safely publish their datasets for research without compromising the information of individuals. However, in recent years, the power differential privacy as a mechanism to preserve privacy in machine learning came into light. The emerging adoption of differential privacy mechanisms in privacy preserving machine learning focuses on two directions: (i) Centralized differential privacy in which a centralized trusted node with accessibility to every sensitive data injects differential privacy noise, and (ii) local differential privacy, and in a distributed ML settings, each participant perturbs private data or the appropriate ML model before sending it out. In deep learning differential privacy can be applied throughout the process starting from the inputs to the model, the objective learning function, back propagation parameters, the final parameters, and labels. The obvious one is to deploy differential privacy on the training data as a preprocessing technique as in the work of [35], [36]. Functional and label-level outputs differential privacy are useful in preventing data gathering and black-box assaults like inference attack or information theft. A gradient-level method is employed to avoid model theft. gradient level approaches are employed in collaborative machine learning only due, as in local training gradients noise will not improve the privacy. However, it suffers from two drawbacks. First, it assumes that the server is trust worthy. Second, Participants in this approach of cooperative learning share the same server's model. As a result all members know the underlying model architecture. The two most influential papers in applying differential privacy in machine learning are discussed below:

- Differential privacy stochastic gradient descent [37]

The algorithm shown in fig 3 is one of the most influential paper in applying differential privacy on machine learning.

Through post-processing the privacy loss for each gradient update is limited. which is done by:

1) Clipping the gradient: Gradient scaling in order to achieve the maximum L2 norm of C. Clipping the gradient limits the information in a single update, allowing us to calculate the maximum privacy loss.
2) Blurring: To the gradient updates, noise is added proportional to the clipping norm (C). The samples are taken from a normal distribution with a standard deviation of C * sigma. The noise multiplier is referred to as sigma.

Algorithm 1 Differentially private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.
Initialize θ_0 randomly
for $t \in [T]$ **do**
Take a random sample L_t with sampling probability L/N
Compute gradient
For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$
Clip gradient
$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)/\max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$
Add noise
$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$
Descent
$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$
Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method.

Fig. 3. DP-SGD algorithm (adapted from [37])

The C and sigma are hyper-parameters tuned to give different privacy guarantees for each update in the training. However, C doesn't affect epsilon since noise is added in propotion to it, but sigma guarantees a certain privacy for each step. The next step is to ensure privacy for the whole training of the model. Which is done with the help of the moments accountant. Due to what was previously explained in the properties of differential privacy, each step in the training is considered a sequential composition, and will deplete the privacy budget by the amount of steps, which will limit the number of steps and training possible in gradient decent. However, the moments accountant is another contribution of the paper where the privacy budget scales to the square root of the gradient updates (T) not T, which allows for running the training for a longer period of time.

- PATE [38]

PATE is different from DP-SGD in that it applies differential privacy in the inference stage of machine learning not in the training phase. PATE intuition builds on two main ideas

- Building a single model and adding noise to the inference will significantly reduce accuracy and performance.
- Building several models, on disjoint datasets, means that if the models agree about an example its more likely that they agree about the general trend of the data and not on a specific data point.

After building several models, they simply vote, and the class that has been voted the most is the correct prediction. To ensure differential privacy, laplacian noise is applied to the numbers of vote of each class, with a specific gamma (the privacy budget). The trade off is that with a higher number of models, the privacy budget can be very small (the machine learning prediction is highly private), however each model

will have access to less number of data and depreciate the performance. There are two limitations to this:

- Every time the attacker query the model, there is information leaked about the underlying data.
- The models are not private, and the weights infer information about the data.

To solve this issue, another classifier is used, that takes as input the unlabeled data from the original dataset, and the deferentially private labels from the models built above.

B. Applications

When it comes to genome-wide applications, such as precision medicine and discovering fine-grained insights in data gathered from a heterogeneous population, machine learning has significant implications. Also to support statistical analysis and machine learning research there has been an increase generation of various genomics datasets, but linkage attacks using auxiliary information provide one of the most significant threats to user privacy in such a system. Differential privacy can protect against such attacks while still allowing valuable researches to done. Differential privacy can also be applied in the case of geolocation applications such as Uber where they need to predict the average distance travelled without revealing the customers privacy. Because of their size, smaller towns may have fewer travelers, therefore a single journey is likely to have an impact on the model, this may be addressed with differential privacy. In the case of Internet of Things or Wearable technology that is commonly used in healthcare. An example use case would be the collection of fixed-interval streams of health data (e.g. gathering minute-by-minute readings of heart rate) citePrivacy-preserving aggregation of personal health data streams by gadgets like smartwatches. According to the system architecture detailed with in study, Local Differential Privacy is used to introduce noise to the data. The gadget recognizes the most important aspects of the readings that might identify the user and then add noise, then the data is finally sent. Another example, is the possibility of processing enormous amounts of data for biological applications while providing differential privacy protections. DAMSON [39] is a deferentially private system with support of several data analytics tasks while effectively optimizing query superior utility at a minimal privacy cost. Differential privacy can be used as a way to conceal the identity of those whose geographic coordinates are saved in the databases. Partitioning a map into smaller sections is a popular strategy, Following this, each sub-region is perturbed in its placement.

C. Limitations

Differential privacy promises are strong and straightforward, but accuracy may be sacrificed in the process. Also if a user had a sequential amount of queries asked, It is necessary to introduce further noise to the data. And if the user had too many queries sequentially they will eventually exceed epsilon (privacy budget) and they will be terminated. In contrast, to sequential, parallel differential privacy requires K amount of noise added for K partitions of the data and if k is huge, the output's utility is lost. The reason is if a query asks to sum the number of records with a certain attribute, with a huge K, even slight noise will add up and mess with the utility of the query. There are still no standards on selecting the privacy budget, and how the different parameter affect privacy exactly. The trade off between utility and privacy is still obscure.

V. Federated learning

The main feature of federated learning is to ensure the privacy of users, but compared with differential privacy and k-order anonymity, which are often used in the area of large data privacy, are extremely distinct. In order to preserve the privacy of users, federated learning mostly exchanges encrypted processed parameters, while the intruders are unable to access the data's source, as the data is not communicated and stays in the respective parties local devices. All of these measures ensure that the privacy of individuals is protected at the data level while using federated learning. Federated learning builds a single global statistical from data that is stored on a distributed environments with a huge number of entities. The objective function to be minimised is: The following six steps are involved in implementing federated learning: 1. Problem identification. 2. party selection: Involved parties or customers are selected based on their ability to provide the necessary data and training. Server configuration data and the core data from each party's side must typically be maintained by the parties involved. 3. Simulation: deployment engineers may use simulation to try out multiple designs and hyper-parameter settings. 4. Federated model training: Multiple federated training tasks among the parties involved are initiated as part of a formal training job. The untrained model is then sent to individuals, who by using their data train it, and then send the updated model to the centralized server. 5. Model evaluation: after all the individual models are aggregated in the centralized server, a new model is built. For the last trained model, iterations of steps 4 and 5 are necessary. 6. Deployment: Whenever the final design for the model has been determined, it becomes ready for deployment after ensuring it's performance.

A. Five aspects of federated learning

there are five five main aspects for federated learning; data partition, privacy mechanism, suitable ML approaches, communication architecture, and methods for solving heterogeneity [40]. Data partition Data partition is the way data is partitioned in preparation for the federated learning approaches, which are on 2 dimensions, the features and the labels. federated learning approaches can be divided into; horizontal and vertical federated learning and federated transfer learning depending the data partition. Horizontal federated learning is mostly used when two datasets have different users, but they have common features. in which the data is split horizontally (by the users), what that means is that the features are taken to be in common. It's possible to extend the number of data examples by using horizontal federated learning. It's not uncommon to have two separate suppliers of the same

service in two different locations, each with a distinct customer base and minimal overlap. Users features will be identical since their companies are so similar. It then becomes possible to expand the amount of training samples and enhance the model's accuracy. Horizontally federated learning commonly requires all parties to compute and submit local gradients to the central server so that it may aggregate them into a global model. Thus, the concern arises that, the processing and exchange of gradients may expose confidential data. In this step comes the solutions that have been discussed earlier, such as homomorphic encryption and differential privacy. Vertical federated learning is the opposite use case of horizontal federated learning. The idea is to split the data by the examples, i.e. we take similar users regardless of their features. This results in a higher number of features. For example, there are two separate institutions in the same location: one is a bank and the other is an e-commerce corporation. Most of the people in the region are likely to be included in their user groups, resulting in more users intersections. However, their respective features almost have no overlap. Machine learning can also be used here to partition the data. In the event when the two datasets don't really intersect, federated transfer learning may be utilized to circumvent the absence of data or labels by using transfer learning. Transfer learning is a method used to improve the performance in a model with limited data. For instance,if a hospital wanted to train a model to diagnose x-ray scans, they might train the models on a closely related task of image recognition, and then transfer the knowledge to diagnose x-ray scans. In this regard two different institutions with a closely related learning task, but with different features and users, might cooperate with federated learning to build a transfer learning model. The privacy mechanism In federated learning, cooperative clients may maintain their own data locally, but they must exchange model information in order to train the target model, which in turn will expose some private data. common solutions are differential privacy, homomorphic encryption, and model aggregation, a discussion of which may be found above. In model aggregation trains the global model by summarizing the model parameters from all parties, so as to avoid transmitting the original data in the training process. model aggregation avoids transferring the original data by summing all model parameters from all partners in the training process. Applicable ML models At present, federated learning has been widely used in machine learning models, but with the rapid development of machine learning, it is still a challenge to propose practical and efficient federated learning tasks. The three most popular models for federated learning are; linear models, tree models, and neural network models.Currently, federated learning is extensively utilized in machine learning models, however with the fast growth of machine learning, it is still difficult to offer practical and effective federated learning tasks. Neural network, tree, and Linear models are the most often used for federated learning. Communication architecture As part of the architecture of distributed training, all distant devices may connect with a central server and participate in updating the global model. However, the training validity of the entire model is affected by the flexibility of local updates and party interaction in the federal context. which give rise to various issues, such as the unequal distribution of user data, the computational capability of the equipment, and so on. As such developing a communication mechanism that can handle the immense communication workload is necessary. Methods for solving heterogeneity In a federated learning situation, the inefficiency of the whole training process would be affected by the difference in equipment. there are four issues to consider when solving the issue of heterogeneity: model heterogeneity, fault-tolerant mechanism, device sampling, and asynchronous communication. Devices do not have to be included in every iterative training session. they might be selected or gives their own will to engage in training. The different methods that orchestrate this are called device sampling. Fault-tolerant mechanism In any distributed environment, there must be a fault tolerant mechanism to prevent the whole system from collapsing in the event a device failed, which is bound to happen.

B. Applications

Many technical fields might benefit from federated learning. Namely in mobile phones, data sensitive fields such as health care, and industry [41]. A great deal of research has been done on federated learning in mobile phones since Google first suggested utilizing it to predict keystrokes through Gboard on Android devices. with the immense storage and processing power mobile devices have grown in recent years, because of the limited transmission capacity, it is challenging to meet the increasing quality demands of mobile users. Because of this, most service providers choose to provide a service near to the consumer rather than incorporate cloud infrastructure into the main network to prevent network congestion. There is an increased danger of information leaking with this technology, which is referred to as mobile edge computing (MEC). The pairing between federated learning and MEC is a feasible option. In this context, "mobile devices" refers not just to traditional cell phones but also to IOT-enabled gadgets. In light of federated learning's success in protecting data confidentiality, it is only natural that industrial engineering would fall into line. This data isn't readily accessible because of restrictions imposed by laws and regulations. Nevertheless, these scattered datasets can only be used to obtain several advantages when federated learning is applied to these sectors. federated learning enables data owners to expand the range of applications and enhance performance of the model via iterations across diverse entities. In the future, federated learning technology would also support more industries to become more intelligent. The incorporation with federated learning in AI will build a federal ecosystem without data privacy concern. Health care is another area where federated learning has a lot of promise. As one of the most common drawbacks in medical institution is that they don't have enough patient data to build good models, and the patient data can't be shared with other institutions. That way different institutions can collaborate using federated learning, to expand their datasets

and build powerful models for diagnosis purposes. Thus, it is optimistic that federated learning would have great potential in the future development. Nowadays, federated learning is mainly utilised in the horizontal context (i.e. building models on similar features), however, a future direction would be to collaborate with other institutions like insurance companies for pricing. As a result, studying federated learning from a vertical perspective seems to be a worthwhile endeavor. Another issue is that current federated learning focuses on a limited number of companies, and it is not equipped to provide training for a large number of devices or institutions. Therefore, analysis of mobile devices data based on federated learning in an effective way should be progressed to generate more meaningful information.

C. Challenges

There are four main challenges that face federated learning as it currently stands. As federated learning is different from typical private data analysis or distributed learning in data centers. In federated networks, communication is a major constrain and, when when privacy issues about the transmission of raw data are taken into account, data generated on every equipment must be kept local. Also, a huge number of devices may be included in federated networks, and because of restricted power resources, energy, and bandwidth network communication may be several folds of magnitude slower than local processing. This means that if a model is to be trained on federated network data, communication-efficient techniques must be developed that repeatedly transmit short messages or model changes as part of the training process rather than sending the complete data set. The following are two crucial elements to consider lowering in order to further limit communication-overhead in such a setting: 1) the communication rounds and 2) and the messages at each round. communication capabilities, computing, and the storage of each device in federated networks varies because of the heterogeneity in the battery level, network connection (3G, 4G, 5G, and Wi-Fi), and underlying-hardware (memory and CPU). These constraints results on a usually huge fraction of the devices not participating at any given communication rounds, or even drop out during an active communication round. Also such characteristics at the system-level introduce challenges such as fault tolerance and straggler mitigation. as a result the following must be done: 1) a modest level of involvement from the parties involved should be expected, 2) Adapt to a wide range of hardware, and 3) be fault tolerant to any communication failures from devices. independent and identically distributed (i.i.d.) assumptions, are a very common assumptions for machine learning algorithms regarding the distribution of data. Which states that samples all come from the same distribution and each sample is independent of any other sample. Such assumption is violated in the context of Federated learning due to how the data is collected in such a way that samples are not identically distributed. For example each user have a different structure of speaking the same language and therefore the underlying distribution of the same language differs from user to user. As a result modeling the problem, analysing theoretically and empirically evaluating the solutions become increasingly complex in distributed optimization problems (i.e. federated learning). Two solutions are to redefine the objective function, which are; metalearning and multitask. The idea is that they model in a device specific manner or a personalized manner, which solves the issue of having samples coming from diverse statistical distribution. Finally, as discussed above, federated learning only provides a safe environment for learning models, and it must still be implemented with other approaches such as differential learning and homomorphic learning. However, these approaches sacrifice system effectiveness or model performance in order to offer privacy. Such trade-offs must be taken into account, empirically and theoretically, which is very challenging to do and is a limitation.

VI. Challenges and future direction

Privacy preserving machine learning introduces another worry for researchers on top of the usual evaluation of the model's performance. Such worries are what should be threat models and trust assumptions for the model, ideally it should be looked at from the data owner perspective. Another issue is evaluation, as it is not feasible to build a vanilla machine learning model on the same data for comparison and evaluation, as doing so will disrupts the security of the data. As it stands it is difficult to quantify the trade off between privacy and utility, which poses a challenge on to how precisely meet the privacy and utility requirements asked for. Privacy mechanisms impose a significant computational overhead, on top of how much vanilla machine learning is already computationally expensive. Privacy preserving machine learning faces difficulty in solving the class imbalance issue, i.e. underrepresented classes get lost in noise which affects the model's performance when solving such problem. The issue is that usually class imbalance problems occur in areas where the data is sensitive for example medical diagnosis, which exacerbates the issue of compromising utility vs privacy.

With these issues in mind in the future more attention should be paid to the following: a methodical approach to defining, measuring, and assessing privacy, as this will make the comparison between the literature more robust, and compare privacy preserving solutions more robustly.

- Developing efficient communication architecture to lower the communication over-head, which was discussed more thoroughly in the federated learning challenges section.
- Optimizing and designing new machine learning algorithms with privacy preserving in mind to increase the computational inefficiency. As it currently stands, computational overhead in privacy preserving machine learning can be inhibiting.
- Differential privacy challenges and their adverse effect on machine learning models utility is prominent. Especially in the distributed environment (parallel differential privacy) where a huge number of K will severely affect the model's utility.

- Interoperability studies are a good direction to understand the hidden architectures of deep neural network, which will in turn help direct more efficient privacy preserving solutions.

VII. Appendix

The ten main papers used in this article are:

- Deep learning with differential privacy [37]
- Privacy-preserving Machine Learning as a Service [42]
- Differential privacy in Deep learning [14]
- A review of applications in federated learning [41]
- A taxonomy and survey of attacks against machine learning [18]
- A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection [13]
- A Systematic Review of Challenges and Techniques of Privacy-Preserving Machine Learning [16]
- A survey on federated learning [40]
- Membership Inference Attacks Against Machine Learning Models [24]
- semi-supervised knowledge transfer for deep learning from private training data [38]

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

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

[3] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1146–1159, 2019.

[4] N. Vemprala and G. Dietrich, "A social network analysis (sna) study on data breach concerns over social media," in *Proceedings of the 52nd hawaii international conference on system sciences*, 2019.

[5] J. Jia, A. Salem, M. Backes, Y. Zhang, and N. Z. Gong, "Memguard: Defending against black-box membership inference attacks via adversarial examples," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019, pp. 259–274.

[6] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference," in *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, pp. 148–162.

[7] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, 2018, pp. 619–633.

[8] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

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

[10] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, and Z. Lin, "When machine learning meets privacy: A survey and outlook," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–36, 2021.

[11] M. Al-Rubaie and J. M. Chang, "Privacy-preserving machine learning: Threats and solutions," *IEEE Security & Privacy*, vol. 17, no. 2, pp. 49–58, 2019.

[12] J. W. Kim, K. Edemacu, J. S. Kim, Y. D. Chung, and B. Jang, "A survey of differential privacy-based techniques and their applicability to location-based services," *Computers & Security*, vol. 111, p. 102464, 2021.

[13] 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.

[14] T. Ha, T. K. Dang, T. T. Dang, T. A. Truong, and M. T. Nguyen, "Differential privacy in deep learning: an overview," in *2019 International Conference on Advanced Computing and Applications (ACOMP)*. IEEE, 2019, pp. 97–102.

[15] R. Xu, N. Baracaldo, and J. Joshi, "Privacy-preserving machine learning: Methods, challenges and directions," *arXiv preprint arXiv:2108.04417*, 2021.

[16] K. Tiwari, S. Shukla, and J. P. George, "A systematic review of challenges and techniques of privacy-preserving machine learning," *Data Science and Security*, pp. 19–41, 2021.

[17] W. Du, Y. S. Han, and S. Chen, "Privacy-preserving multivariate statistical analysis: Linear regression and classification," in *Proceedings of the 2004 SIAM international conference on data mining*. SIAM, 2004, pp. 222–233.

[18] N. Pitropakis, E. Panaousis, T. Giannetsos, E. Anastasiadis, and G. Loukas, "A taxonomy and survey of attacks against machine learning," *Computer Science Review*, vol. 34, p. 100199, 2019.

[19] 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," *Advances in neural information processing systems*, vol. 31, 2018.

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

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

[22] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing," in *23rd USENIX Security Symposium (USENIX Security 14)*, 2014, pp. 17–32.

[23] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1322–1333.

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

[25] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 233–242.

[26] N. Carlini, C. Liu, J. Kos, Ú. Erlingsson, and D. Song, "The secret sharer: Measuring unintended neural network memorization & extracting secrets," *arXiv preprint arXiv:1802.08232*, vol. 5, 2018.

[27] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *Journal of Network and Computer Applications*, vol. 116, pp. 1–8, 2018.

[28] E. Hesamifard, H. Takabi, and M. Ghasemi, "Cryptodl: Deep neural networks over encrypted data," *arXiv preprint arXiv:1711.05189*, 2017.

[29] S. Shukla and G. Sadashivappa, "Secure multi-party computation protocol using asymmetric encryption," in *2014 International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2014, pp. 780–785.

[30] E. Makri, D. Rotaru, N. P. Smart, and F. Vercauteren, "Epic: efficient private image classification (or: Learning from the masters)," in *Cryptographers' Track at the RSA Conference*. Springer, 2019, pp. 473–492.

[31] S. Shukla and G. Sadashivappa, "A distributed randomization framework for privacy preservation in big data," in *2014 Conference on IT in Business, Industry and Government (CSIBIG)*. IEEE, 2014, pp. 1–5.

[32] N. Agrawal, A. Shahin Shamsabadi, M. J. Kusner, and A. Gascón, "Quotient: two-party secure neural network training and prediction," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1231–1247.

[33] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 19–38.

[34] D. Demmler, T. Schneider, and M. Zohner, "Aby-a framework for efficient mixed-protocol secure two-party computation." in *NDSS*, 2015.

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

[36] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy." *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3-4, pp. 211–407, 2014.

[37] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[38] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *arXiv preprint arXiv:1610.05755*, 2016.

[39] M. Winslett, Y. Yang, and Z. Zhang, "Demonstration of damson: Differential privacy for analysis of large data," in *2012 IEEE 18th International Conference on Parallel and Distributed Systems*. IEEE, 2012, pp. 840–844.

[40] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106775, 2021.

[41] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020.

[42] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service." *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, 2018.