Privacy in Deep Learning: A Survey

Fatemehsadat Mireshghallah [1] **Mohammadkazem Taram** [1]
Praneeth Vepakomma [2] **Abhishek Singh** [2] **Ramesh Raskar** [2] **Hadi Esmaeilzadeh** [1]

Abstract

The ever-growing advances of deep learning in many areas including vision, recommendation systems, natural language processing, etc., have led to the adoption of Deep Neural Networks (DNNs) in production systems. The availability of large datasets and high computational power are the main contributors to these advances. The datasets are usually crowdsourced and may contain sensitive information. This poses serious privacy concerns as this data can be misused or leaked through various vulnerabilities. Even if the cloud provider and the communication link is trusted, there are still threats of inference attacks where an attacker could speculate properties of the data used for training, or find the underlying model architecture and parameters. In this survey, we review the privacy concerns brought by deep learning, and the mitigating techniques introduced to tackle these issues. We also show that there is a gap in the literature regarding test-time inference privacy, and propose possible future research directions.

1. Introduction

The success of Deep Neural Networks (DNNs) in various fields including vision, medicine, recommendation systems, natural language processing, etc., has resulted in their deployment in numerous production systems [1, 2]. In the world of medicine, learning is used to find patterns in patient histories and to recognize abnormalities in medical imaging which help with disease diagnosis and prognosis. The use of machine learning in healthcare can compromise patient privacy, for instance by exposing the patient's genetic markers, as shown by Fredrikson et al. [3]. Deep learning is also widely used in finance for predicting prices or creating portfolios, among many other applications. In these cases, usually, an entity trains its own model and the model parameters are considered confidential. Being able to find or infer them is considered a breach of privacy [4]. Ease of access to large datasets and high computational power (GPUs and TPUs) have paved the way for the aforementioned advances. These datasets are usually crowdsourced and might contain sensitive information. This poses serious privacy concerns, as neural networks are used in different aspects of our lives [5, 3, 6, 7, 8, 9, 10].

Figure 1 shows a classification of possible threats to deep learning. One threat is the direct intentional or unintentional exposure of sensitive information, through untrusted data curator, communication link, or cloud [11, 12]. This information can be the training data, inference queries or model parameters or hyperparameters. If we assume that information cannot be attained directly, there is still the threat of information exposure through inference, indirectly. Membership inference attacks [13] can infer whether a given data instance was part of the training process of a model. Model inversion and attribute inference attacks can infer sensitive features about a data instance, from observed predictions of a trained model, and other non-sensitive features of that data instance [14, 15]. Some attacks are targeted towards stealing information about a deployed model, such as its architecture [16], trained parameters [17] or a general property of the data it was trained on, for instance, if the images used for training were all taken outdoor [18].

There is a myriad of methods proposed to tackle these threats. The majority of these methods focus on the data aggregation/dataset publishing and training stages of deep learning. We classify these methods into three classes. The first class of methods focuses on sanitizing the data and trying to remove sensitive information from it while maintaining the statistical trends [19, 20]. The second class focuses on making the DNN training phase private and protecting the data used for training [21, 22, 23, 24, 25]. The last class, of which there is only a handful of works, attempts to protect the privacy of the test-time inference phase by protecting the input data (request) that the user sends to a deployed DNN [26, 27, 28].

In this paper, we first briefly discuss existing attacks and privacy threats against deep learning. Then, we focus on the existing privacy-preserving methods for deep learning and demonstrate that there is a gap in the literature regarding test-time inference privacy.

2. Existing Threats

In this section, we map the space of existing threats against privacy in deep learning and machine learning in general. While the focus of this survey is privacy-preserving

[1]University of California San Diego [2]Massachusetts Institute of Technology. Correspondence to: Fatemehsadat Mireshghallah <fmireshg@eng.ucsd.edu>.

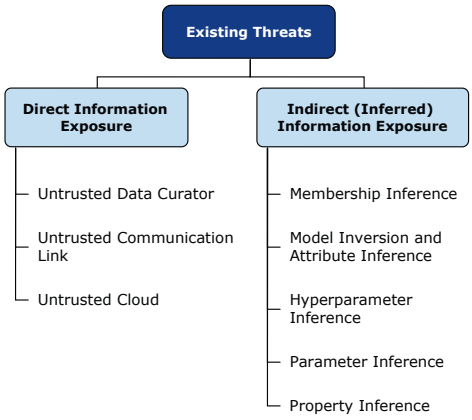


Figure 1. Categorization of existing threats against deep learning

techniques, we provide a brief summary of attacks to better situate the need for privacy protection. Figure 1 shows the landscape of these threats, which we have divided into two main categories of direct and indirect information exposure hazards. Direct threats are those where the attacker can gain access to the actual information. In indirect attacks, however, the attacker tries to infer or guess the information and does not have access to the actual information.

2.1. Direct Information Exposure

Direct intentional or unintentional data breaches can occur in many different settings and are not limited to machine learning. Dataset breaches through data curators or entities housing the data can be caused unintentionally by hackers, malware, virus, or social engineering, by tricking individuals into handing over sensitive data to adversaries [11]. A study by Intel Security [29] demonstrated that employees are responsible for 43% of data leakage, half of which is believed to be unintentional. A malicious party can exploit a system's backdoor to bypass a server's authentication mechanism and gain direct access to sensitive datasets, or sensitive parameters and models [30, 31, 32]. The recent hacking of Equifax, for instance, exploited a vulnerability in the Apache Struts software, which was used by Equifax [30].

Data sharing by transmitting confidential data without proper encryption is an example of data exposure through communication link [33]. Kaspersky Labs reported in 2018 that they found four million Android apps that were sending unencrypted user profile data to advertisers' servers [34]. Private data can also be exposed through the cloud service that receives it to run a process on it, for instance, Machine Learning as a Service (MLaaS). Some of these services do not clarify what happens to the data once the process is finished, nor do they even mention that they are sending the user's data to the cloud, and not processing it locally [6].

2.2. Indirect (Inferred) Information Exposure

As shown in figure 1, we categorize indirect attacks into 5 main groups of membership inference, model inversion, hyperparameter inference, parameter inference, and property inference attacks. Table 1 shows a summary of different attacks and their properties. The "Access to Model" column determines whether the attack needs white-box or black-box access to model to successfully mount. White-box access assumes access to the full target model, whereas black-box assumes only query access to the model, without knowledge on the architecture or parameters of the target model. The last column shows whether the attacker needs access to the output confidence values of the model (the probabilities, logits), or whether only the predicted labels suffice.

2.2.1. Membership Inference

Given a data instance and (black-box or white-box) access to a pre-trained target model, a membership inference attack speculates whether or not the given data instance has contributed to the training step of the target model. Shokri et al. [13] propose the first membership inference attack on machine learning where they consider an attacker who has black-box query access to the target model and can obtain confidence scores (probability vector) for the queried input. The attacker uses this confidence score to deduce the participation of given data in training. They first train shadow models on a labeled dataset that can be generated using three methods: model inversion attack (we will see next), statistics-based synthesis (through assumptions about the underlying distribution of training set), or noisy real data. Using these shadow models, the attacker trains an "attack model" that distinguishes the participation of a data instance in the training set of the shadow models. Lastly, for the main inference attack, the attacker makes queries to the target deployed model to receive confidence scores for each given input data instance and infers whether or not the input was part of the target training data. This attack is built on the assumption that if a record was used in the training of a model, it would yield a higher confidence score, than a record which was not seen before by the model.

Some studies [44, 45, 15] attribute membership inference attacks to the generalization gap, the over-fitting of the model, and data memorization capabilities of neural networks. Deep neural networks have been shown to memorize the training data [46], rather than learning the latent properties of the data, which means they often tend to over-fit to the training data. Long et al. [35] propose an approach which more accurately tests the membership of a given instance. They train the shadow models with and without this given instance, and then at inference time the attacker tests to see if the instance was used for training the target model, similar to Shokri et al.'s approach. More recently, Salem et al. [36] propose a more generic attack that could relax the main requirements

Table 1. Properties of some notable attacks against machine learning privacy

Attack	Membership Inference	Model Inversion	Hyperparam. Inference	Parameter Inference	Property Inference	Access to Model	Access to Model Output
Membership Inference [13]	●	○	○	○	○	Blackbox	Logits
Measuring Membership Privacy[35]	●	○	○	○	○	Blackbox	Logits
ML-Leaks[36]	●	○	○	○	○	Blackbox	Logits
The Natural Auditor [37]	●	○	○	○	○	Blackbox	Label
LOGAN [38]	●	○	○	○	○	Both	Logits
Privacy Risk in ML [15]	●	●	○	○	○	Whitebox	Logits+Auxilary
Fredrikson et al. [3]	○	●	○	○	○	Blackbox	Logits
MIA w/ Confidence Values [14]	○	●	○	○	○	Both	Logits
Adversarial NN Inversion [39]	○	●	○	○	○	Blackbox	Logits
Updates-Leak [40]	○	●	○	○	○	Blackbox	Logits
The Secret Sharer [41]	○	○	○	○	●	Blackbox	Logits
Property Inference on FCNNs [18]	○	○	○	○	●	Whitebox	Logits
Hacking Smart Machines w [42]	○	○	○	○	●	Whitebox	Logits
Cache Telepathy [16]	○	○	○	●	○	Blackbox	Logits
Stealing Hyperparameters [43]	○	○	○	●	○	Blackbox	Logits
Stealing ML Models [17]	○	○	●	●	○	Blackbox	Label

Figure 2. The image on the left was recovered using the model inversion attack of Fredrikson et al. [14]. The image on the right shows an image from the training set. The attacker is given only the person's name and access to a facial recognition system that returns a class confidence score [14].

in previous attacks (such as using multiple shadow models, knowledge of the target model structure, and having a dataset from the same distribution as the target model's training data), and show that such attacks are also applicable at a lower cost, without significantly degrading their effectiveness.

Membership inference attacks do not always need access to the confidence values (logits) of the target model, as shown by Song & Shmatikov in a recent attack [37], which can detect with very few queries to a model if a particular user's texts were used to train it.

Yeom et al [15] suggest a membership inference attack for cases where the attacker can have white-box access to the target model and know the average training loss of the model. In this attack, for an input record, the attacker evaluates the loss of the model and if the loss is smaller than a threshold (the average loss on the training set), the input record is deemed part of the training set. Membership inference attacks can also be applied to Generative Adversarial Networks (GANs), as shown by Hayes et al. [38].

2.2.2. Model Inversion and Attribute Inference

Model inversion and attribute inference attacks are against attribute privacy, where an adversary tries to infer sensitive attributes of given data instances from a released model and the instance's non-sensitive attributes [47]. The most prominent of these attacks is against a publicly-released linear regression model, where Fredrikson et al. [3] invert the model of a medicine (Warfarin) dosage prediction task. They recover genomic information about the patient, based on the model output and several other non-sensitive attributes (e.g., height, age, weight). This attack can be applied with only black-box API access to the target model. Fredrikson et al. formalize this attack as maximizing the posterior probability estimate of the sensitive attribute. In other words, the attacker assumes that features f_1 to f_{d-1}, of the f_d features of each data instance are non-sensitive. the attacker then tries to maximize the posterior probability of feature f_d, given the nonsensitive features of f_1 to f_{d-1}, and the model output.

In another work, given white-box access to a neural network, Fredrikson et al. [14] show that they could extract instances of training data, from observed model predictions. Figure 2 shows a recovered face image that is similar to the input image and was reconstructed by utilizing the confidence score of the target model. Yeom et al. [15] also propose an attribute inference attack, built upon the same principle used for their membership inference attack, mentioned in Section 2.2.1. The attacker evaluates the model's loss on the input instance for different values of the sensitive attribute and infers the value that yields a loss value similar to that outputted by the original data, as the sensitive value. Salem et al. [40] suggest a model inversion attack on online-learning, using a generative adversarial network and based on the difference between a model, before and after a gradient update.

2.2.3. Model Stealing: Hyperparameter and Parameter Inference

Trained models are considered intellectual properties of their owners and can be considered confidential in many cases [4], therefore extracting the model can be considered a privacy breach. Apart from this, as discussed earlier, DNNs are shown to memorize information about their training data, therefore exposing the model parameters could lead to exposure of training data. A model stealing attack is meant to recover the model parameters via black-box access to the target model. Tramer et al. [17] devise an attack that finds parameters of a model given the observation of its predictions (confidence values). Their attack tries to find parameters of the model through equation solving, based on pairs of input-outputs. This attack cannot be mounted in a setting where the confidence values are not provided.

Hyperparameter stealing attacks try to find the hyperparameters used during the model training, such as the regularization coefficient [43] or model architecture [16].

2.2.4. Property Inference

This class of attacks tries to infer specific patterns of information from the target model. An example of these attacks is the memorization attack that aims to find sensitive patterns in the training data of a target model [41]. These attacks have been mounted on Hidden Markov Models (HMM) and Support Vector Machines (SVM) [42] and neural networks [18].

3. Privacy-Preserving Mechanisms

In this section, we review the literature of privacy-preserving mechanisms for deep learning and machine learning in general. Figure 3 shows our classification of the landscape of this field. We divide the literature into three main groups. The first is private data aggregation methods, which aim at collecting data and forming datasets, while preserving the privacy of the contributors [19, 20]. The second group, which is comprised of a large body of work focuses on devising mechanisms that make the training process of models private so that sensitive information about the participants of the training dataset would not be exposed. Finally, the last group aims at the test-time inference phase of deep learning. It tries to protect the privacy of users of deployed models, who send their data to a trained model for having a given inference service carried out.

3.1. Data Aggregation

Here, we introduce the most prominent data privacy-preserving mechanisms. Not all these methods are applied to deep learning, but we briefly discuss them for the sake of comprehensiveness. These methods can be broadly divided into two groups of context-free privacy and context-aware. Context-free privacy solutions, such as differential privacy, are unaware of the specific context or the purpose that the data will be used for. Whereas context-aware privacy solutions, such as information-theoretic privacy, are aware of the context where the data is going to be used, and can achieve an improved privacy-utility tradeoff [48].

3.1.1. Naive Data Anonymization

What we mean by naive anonymization in this survey is the removal of identifiers from data, such as the names, addresses, and full postcodes of the participants, to protect privacy. This method was used for protecting patients while processing medical data and has been shown to fail on many occasions [49, 19, 50]. Perhaps the most prominent failure is the Netflix prize case, where Narayanan & Shmatikov apply their de-anonymization technique to the Netflix Prize dataset. This dataset contains anonymous movie ratings of 500,000 subscribers of Netflix. They showed that an adversary with auxiliary knowledge (from the publicly available Internet Movie Database records) about individual subscribers can easily identify the user and uncover potentially sensitive information [49].

3.1.2. K-Anonymity

A dataset has k-anonymity property if each participant's information cannot be distinguished from at least $k-1$ other participants whose information is in the dataset [19]. K-anonymity means that for any given combination of attributes that are available to the adversary (these attributes are called quasi-identifiers), there are at least k rows with the exact same set of attributes. K-anonymity has the objective of impeding re-identification. However, k-anonymization has been shown to perform poorly on the anonymization of high-dimensional datasets [51]. This has led to privacy notions such as l-diversity [52] and t-closeness [53], which are out of the scope of this survey.

3.1.3. Differential Privacy

Definition 3.1. ϵ-Differential Privacy (ϵ-DP). For $\epsilon \geq 0$, an algorithm A satisfies ϵ-DP [54, 20] if and only if for any pair of datasets D and D' that differ in only one element:

$$\mathcal{P}[A(D)=t] \leq e^{\epsilon}\mathcal{P}[A(D')=t] \ \forall t \tag{1}$$

where, $\mathcal{P}[A(D)=t]$ denotes the probability that the algorithm A outputs t. In this setup, the quantity $ln\frac{\mathcal{P}[A(D)=t]}{\mathcal{P}[A(D')=t]}$ is named the privacy loss. DP tries to approximate the effect of an individual opting out of contributing to the dataset, by ensuring that any effect due to the inclusion of one's data is small. One of the widely used DP mechanisms when dealing with numerical data is the Laplace mechanism.

Definition 3.2. Laplace Mechanism. [20] Given a target function f and a fixed $\epsilon \geq 0$, the randomizing algorithm $A_f(D) = f(D) + x$ where x is a perturbation random variable drawn from a Laplace distribution $Lap(\mu, \frac{\Delta_f}{\epsilon})$, is called the Laplace Mechanism and is ϵ-DP. Here, Δ_f

is the global **sensitivity** of function f, and is defined as $\Delta_f = \sup|f(D) - f(D')|$ over all the dataset pairs (D, D') that differ in only one element. Finding this sensitivity is not always trivial, specifically if the function f is a deep neural network, or even a number of layers of it [55].

Differential privacy satisfies a composition property that states when two mechanisms with privacy budgets ϵ_1 and ϵ_2 are applied to the same datasets, together they use a privacy budget of $\epsilon_1 + \epsilon_2$. As such, composing multiple differentially private mechanisms consumes a linearly increasing privacy budget. It has been shown that tighter privacy bound for composition can be reached, so that the privacy budget decreases sub-linearly [56, 57]. There are multiple variants of the conventional ϵ-differential privacy which have been proposed to provide a tighter analysis of the privacy budget under composition. One of them is differential privacy with Advanced Composition (AC) [58], which allows an additive leakage probability parameter δ to the right-hand side of Equation 1.

Differential privacy can also be achieved without the need to trust a centralized server by having each participant apply a differentially private randomization to their data themselves, before sharing it. This model is named the local model of differential privacy, and the method "randomized response" is shown to be locally differentially private [59]. Local differential privacy has been deployed on many systems for gathering statistics privately. For instance, Google uses a technique named RAPPOR [60] to allow web browser developers to privately collect usage statistics. A large body of Differentially private mechanisms has been proposed for various applications. Triastcyn & Faltings present a technique that generates synthetic datasets that still have statistical properties of the real data while providing differential privacy guarantees with respect to this data [61]. A generalized version of differential privacy called Pufferfish was proposed by [62]. The Pufferfish framework can be used to create new privacy definitions tailored for specific applications [63], such as Census data release.

3.1.4. Semantic Security and Encryption

Semantic security [64] (computationally secure) is a standard privacy requirement of encryption schemes which states that the advantage (a measure of how successfully an adversary can attack a cryptographic algorithm) of an adversary with background information should be cryptographically small. Semantic security is theoretically possible to break but it is infeasible to do so by any known practical means [65]. Secure Multiparty Computation (SMC), which we discuss in Section 3.2, is based on semantic security definition [66].

3.1.5. Information-Theoretic Privacy

Information-theoretic privacy is a context-aware privacy solution. Context-aware solutions explicitly model the dataset statistics, unlike context-free solutions that assume worst-case dataset statistics and adversaries. There is a body of work studying information-theoretic based methods for both privacy and fairness, where privacy and fairness are provided through information degradation, through obfuscation or adversarial learning and demonstrated by mutual information reduction [67, 68, 48, 69, 70, 71, 72, 73, 74, 75, 76]. Huang et al. introduce a context-aware privacy framework called generative adversarial privacy (GAP), which leverages generative adversarial networks (GANs) to generate privatized datasets. Their scheme comprises of a sanitizer that tries to remove private attributes, and an adversary that tries to infer them [48]. They show that the privacy mechanisms learned from data (in a generative adversarial fashion) match the theoretically optimal ones.

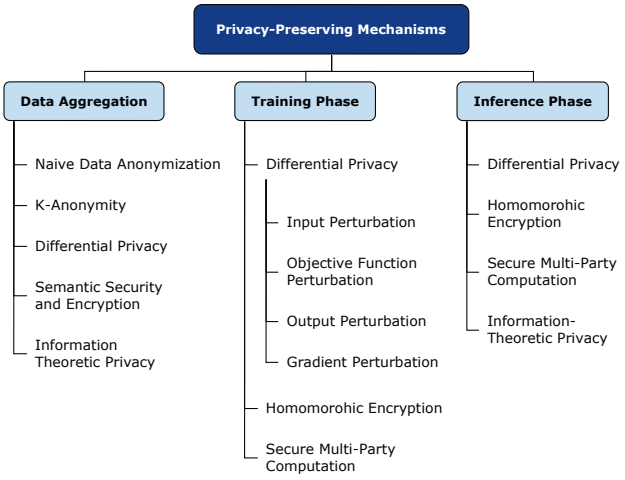


Figure 3. Categorization of privacy-preserving schemes for deep learning.

3.2. Training Phase

The literature surrounding private training of deep learning, and machine learning can be categorized based on the guarantee that these methods provide, which is most commonly either based on differential privacy (Section 3.1.3) or semantic security and encryption (Section 3.1.4). Privacy using encryption is achieved by doing computation over encrypted data. The two most common methods for this are Homomorphic Encryption (HE) and Secure Multi-Party Computation (SMC).

Homomorphic Encryption (HE). HE [101] allows computation over encrypted data. A client can send their data, in an encrypted format, to a server and the server can compute over this data without decrypting it, and then send a ciphertext (encrypted result) to the client for decryption. HE is extremely compute-intensive and is therefore not yet deployed in many production systems [102, 103].

Secure Multi-Party Computation (SMC). SMC attempts at designing a network of computing parties (not all of

Table 2. Categorization of some notable privacy-preserving mechanisms for training. In the table, the following abbreviations have been used: ERM for Empirical Risk Minimization, GM for Generative Model, AE for Auto Encoder, LIR for Linear Regression, LOR for Logistic Regression, LM for Linear Means, FLD for Fisher's Linear Discriminant, NB for Naive Bayes and RF for Random Forest.

Method	DP	SMC	HE	Dataset(s)	Task
DPSGD [22]	●	○	○	MNIST, CIFAR-10	Image Classification w/ DNN
DP LSTM [77]	●	○	○	Reddit Posts	Language Model w/ LSTMs
DP LOR [78]	●	○	○	Artificial Data	Logistic Regression
DP ERM [79]	●	○	○	Adult, KDD-99	Classification w/ ERM
DP GAN [80]	●	○	○	MNIST, MIMIC-III	Data Generation w/ GAN
DP GM [81]	●	○	○	MNIST, CDR, TRANSIT	Data Generation w/ GM
DP AE [82]	●	○	○	Health Social Network Data	Behaviour Prediction w/ AE
DP Belief Network[83]	●	○	○	YesiWell, MNIST	Classification w/ DNN
Adaptive Laplace Mechanism[84]	●	○	○	MNIST, CIFAR-10	Image Classification w/ DNN
PATE [23]	●	○	○	MNIST, SVHN	Image Classification w/ DNN
Scalable Learning w/ PATE [85]	●	○	○	MNIST, SVHN, Adult, Glyph	Image Classification w/ DNN
DP Ensemble [24]	●	○	○	KDD-99, UCI-HAR, URLs	Classification w/ ERM
SecProbe [86]	●	○	○	US, MNIST, SVHN	Regress. & Class. w/ DNN
Distributed DP [87]	●	○	○	eICU, TCGA	Classification w/ DNN
DP model publishing [88]	●	○	○	MNIST, CIFAR	Image Classification w/ DNN
DP federated learning [89]	●	○	○	MNIST	Image Classification w/ DNN
ScalarDP, PrivUnit [90]	●	○	○	MNIST, CIFAR	Image Classification w/ DNN
DSSGD [21]	●	○	○	MNIST, SVHN	Image Classification w/ DNN
Private Collaborative NN [91]	●	●	○	MNIST	Image Classification w/ DNN
Secure Aggregation for ML [92]	○	●	○	-	Federated Learning
QUOTIENT [93]	○	●	○	MNIST, Thyroid, Credit	Classification w/ DNN
SecureNN [94]	○	●	○	MNIST	Image Classification w/ DNN
ABY3 [95]	○	●	○	MNIST	LIR, LOR, NN
SecureML [96]	○	●	●	MNIST, Gisette, Arcene	LIR, LOR, NN
Deep Learning w/ AHE [97]	○	○	●	MNIST	Image Classification w/ DNN
ML Confidential [98]	○	○	●	Wisconsin Breast Cancer	LM, FLD
Encrypted Statistical ML [99]	○	○	●	20 datasets from UCI ML	LOR, NB, RF
CryptoDL [25]	○	○	●	MNIST, CIFAR-10	Image Classification w/ DNN
DPHE [100]	○	○	●	Caltech101/256, CelebA	Image Classification w/ SVM

which the user necessarily has to trust) that carry out a given computation and makes sure no data leaks. Each party in this network has access to only an encrypted part of the data. SMC ensures that as long as the owner of the data trusts at least one of the computing systems in the network, their input data remain secret. Simple functions can easily be computed using this scheme. Arbitrarily complex function computations can also be supported, but with an often prohibitive computational cost [103].

In this survey, we divided the literature of private training into three groups of methods that employ: 1) Differential Privacy (DP), 2) Homomorphic Encryption (HE) and 3) Secure Multi-Party Computation (SMC). Table 2 shows this categorization for the literature we discuss in this section.

3.2.1. Differential Privacy

This section briefly discusses methods for modifying deep learning algorithms to satisfy differential privacy. Figure 4 shows an overview of a deep learning framework. As can be seen, the randomization required for differential privacy (or the privacy-preserving noise) can be inserted in five places: to the input, to the loss/objective function, to the gradient updates, to the output (the optimized parameters of the trained model) and to the labels [23].

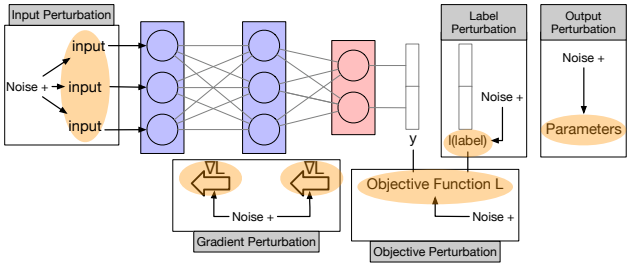


Figure 4. Overview of how a deep learning framework works and how differential privacy can be applied to different parts of the pipeline.

Input perturbations can be considered equivalent to using a sanitized dataset (discussed in Section 3.1) for training. **objective function perturbation** and **output perturbation** are explored for machine learning tasks with convex objective functions. For instance in the case of logistic regression, Chaudhuri et al. prove that objective perturbation requires sampling noise in the scale of $\frac{2}{n\epsilon}$, and output perturbation requires sampling noise in the scale of $\frac{2}{n\lambda\epsilon}$, where n is the num-

ber of samples and λ is the regularization coefficient [79]. In deep learning tasks, due to the non-convexity of the objective function, calculating the sensitivity of the function (which is needed to determine the intensity of the added noise) becomes non-trivial. One solution is replacing the non-convex function with an approximate convex polynomial function [82, 83, 84] and then using objective function perturbation. This approximation limits the capabilities and the utility that a conventional DNN would have. Given discussed limitations, **gradient perturbation** is the approach that is widely used for private training in deep learning. Applying perturbations on the gradients requires the gradient norms to be bounded, and since in deep learning tasks the gradient could be unbounded, clipping is usually used to alleviate this issue.

Shokri et al. showed that deep neural networks can be trained in a distributed manner and with perturbed parameters to achieve privacy [21], but their implementation requires ϵ proportional to the size of the target model, which can be in the order of couple millions. Abadi et al. [22] propose a mechanism dubbed the "moments accountant (MA)", for bounding the cumulative privacy budget of sequentially applied differentially private algorithms, over deep neural networks. The moments accountant uses the moment generating function of the privacy loss random variable to keep track of a bound on the privacy loss during composition. MA operates in three steps: first, it calculates the moment generating functions for the algorithms A_1, A_2,.., which are the randomizing algorithms. It then composes the moments together through a composition theorem, and finally, finds the best leakage parameter (δ) for a given privacy budget of ϵ. The moments accountant is widely used in different DP mechanisms for private deep learning. Papernot et al. use MA to aid bounding the privacy budget for their teacher ensemble method that uses noisy voting and **label perturbation** [23, 85]. MA is also employed by the works [87, 88, 89, 80, 81, 90] all of which use perturbed gradients.

3.2.2. Homomorphic Encryption

There are only a handful of works that exploit solely homomorphic encryption for private training of machine learning models [98, 99, 25]. Graepel et al. use a Somewhat HE (SHE) scheme to train Linear Means (LM) and Fisher's Linear Discriminate (FLD) classifiers [98]. HE algorithms have some limitations in terms of the functions they can compute (for instance they cannot implement non-linearities). For that reason, Graepel et al. propose division-free algorithms and focus on simple classifiers and not complex algorithms such as neural networks.

Hesamifard et al. [25] try to exploit HE for deep learning tasks. They introduce methods for approximating the most commonly used neural network activation functions (ReLU, Sigmoid, and Tanh) with low degree polynomials. This is a crucial step for designing efficient homomorphic encryption schemes. They then train convolutional neural networks with those approximate polynomial functions and finally, implement convolutional neural networks over encrypted data and measure the performance of the models.

3.2.3. Secure Multi-Party Computation (SMC)

A trend in research on private and secure computation consists of designing custom protocols for applications such as linear and logistic regression [96] and neural network training and inference [96, 93, 104]. These methods usually target settings where different datasets from different places are set to train a model together, or where computation is off-loaded to a group of computing servers that do not collude with each other. SMC requires that all participants be online at all times, which requires a significant amount of communication [105]. Mohassel & Zhang proposed SecureML which is a privacy-preserving stochastic gradient descent-based method to privately train machine learning algorithms such as linear regression, logistic regression and neural networks in multi-party computation settings. SecureML uses secret sharing to achieve privacy during training. In a more recent work [95], Mohassel et al design protocols for secure three-party training of DNNs with a majority of honest parties. Agrawal et al. propose QUOTIENT [93] where their goal is to design an optimization algorithm alongside a secure computation protocol customized for it, instead of a conventional approach which is using encryption on top of existing optimization algorithms.

3.3. Inference Phase

As shown in Table 3 there are fewer works in the field of inference privacy, compared to training. Inference privacy targets systems that are deployed to offer Inference-as-a-Service. In these cases, the deployed system is assumed to be trained and is not to learn anything new from the data provided by the user. It is only supposed to carry out its designated inference task. The categorization of literature for inference privacy is similar to training, except that there is one extra group here, named Information-Theoretic (IT) privacy. The works in this group usually offer information-theoretic mathematical or empirical evidence of how their methods operate and help privacy. These works are based on the context-aware privacy definition of Section 3.1.5, and they aim at decreasing the information content in the data sent to the service provider for inference so that there is only as much information in the input as needed for the service and not more.

One notable difference between training and inference privacy is the difference in the amount of literature on different categories. There seems to be a trend of using differential privacy for training, and encryption methods (HE and SMC) for inference. One underlying reason could be computational complexity and implementation. Encryption methods, specifically homomorphic encryption, are shown to be at least two orders of magnitude slower

Table 3. Categorization of some notable privacy-preserving mechanisms for inference. In this table, NB is short for Naive Bayes, and DT is short for Decision Tree.

Method	DP	SMC	HE	IT	Dataset(s)	Task
ARDEN [106]	●	○	○	○	MNIST, CIFAR-10, SVHN	Image Classification w/ DNN
Cloak [107]	●	○	○	●	CIFAR-100, CelebA, UTKFace	Image Classification w/ DNN
Cryptonets [108]	○	○	●	○	MNIST	Image Classification w/ DNN
Private Classification [109]	○	○	●	○	MNIST	Image Classification w/ DNN
TAPAS [110]	○	○	●	○	MNIST, Faces, Cancer, Diabetes	Image Classification w/ DNN
FHE–DiNN [111]	○	○	●	○	MNIST	Image Classification w/ DNN
Face Match [112]	○	○	●	○	LFW, IJB-A, IJB-B, CASIA	Face recognition with CNNs
EPIC [103]	○	●	○	○	CIFAR-10, MIT, Caltech	Image Classification w/ DNN
DeepSecure [113]	○	●	○	○	MNIST, UCI-HAR	Classification w/ DNN
XONN [102]	○	●	○	○	MNIST, CIFAR-10	Image Classification w/ DNN
Chameleon [114]	○	●	○	○	MNIST, Credit Approval	Classification w/ DNN and SVM
Classification over Encrypted Data[115]	○	●	●	○	Wisconsin Breast Cancer	Classification w/ NB, DT
MiniONN [116]	○	●	●	○	MNIST, CIFAR-10	Image Classification w/ DNN
GAZELLE [117]	○	●	●	○	MNIST, CIFAR-10	Image Classification w/ DNN
DELPHI [118]	○	●	●	○	CIFAR-10, CIFAR-100	Image Classification w/ DNN
Shredder [28]	○	○	○	●	SVHN, VGG-Face, ImageNet	Classification w/ DNN
DPFE [26]	○	○	○	●	CelebA	Image Classification w/ DNN

than conventional execution [117]. That's why adopting them for training will increase training time significantly. Also, as mentioned in Section 3.2.2, due to approximating non-linear functions, the capabilities of neural networks in terms of performance become limited during training on encrypted data. For inference, however, adopting encryption is more trivial, since the model is already trained. Employing differential privacy, and noise addition, however, is less trivial for inference, since it could damage the accuracy of the trained model, if not done meticulously. Below we delve deeper into the literature of each category.

3.3.1. Differential Privacy

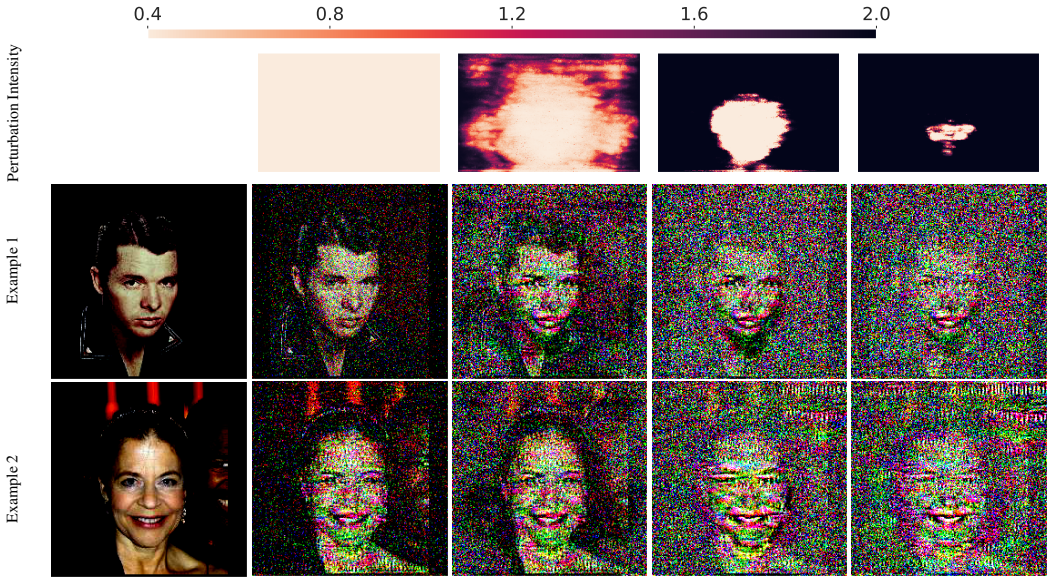
There are very few works using differential privacy for inference. The main reason is that differential privacy offers a worst-case guarantee which requires high-intensity noise (noise with high standard deviation) to be applied to all the segments of the input. This inherently causes performance degradation on pre-trained networks. Wang et al. [106] propose Arden, a data nullification and differentially private noise injection mechanism for inference. Arden partitions the DNN across edge device and the cloud. A simple data transformation is performed on the mobile device, while the computation heavy and complex inference relies on the cloud data center. Arden uses data nullification, and noise injection to make different queries indistinguishable so that the privacy of the clients is preserved. The proposed scheme requires noisy retraining of the entire network, with noise injected at different layers. Since it is complicated to calculate the global sensitivity at each layer of the neural network, the input to the noise injection layer is clipped to the largest possible value created by a member of the training set, on the trained network.

Mireshghallah et al. propose a non-intrusive approach dubbed Cloak, in which there is no need to change/retrain the network parameters, nor partition it. In short, Cloak suggests a principled approach to learning Laplace noise distributions that when added to the input, create perturbed representations. These representations obfuscate the sensitive information in the input while attempting to keep the nonsensitive information intact, to be used for the inference task at hand. An example of representations produced by Cloak, for a smile detection task can be seen in Figure 5. In other words, Cloak combines the context-aware privacy of information-theoretic methods, with context-free differential privacy to detect and lose excessive information in the inputs, while providing a worst-case bound on the additive noise.

3.3.2. Homomorphic Encryption

CryptoNets is one of the first works in HE inference [108]. Dowlin et al. present a method for converting a trained neural network into an encrypted one, named a CryptoNet. This allows the clients of an inference service to send their data in an encrypted format and receive the result, without their data being decrypted. CryptoNets allows the use of SIMD (Single Instruction Multiple Data) operations, which increase the throughput of the deployed system. However, for single queries, the latency of this scheme is still high.

Chabanne et al. [109] approximate the ReLu non-linear activation function using low-degree polynomials and provide a normalization layer before the activation function, which offers high accuracy. However, they do not show results on the latency of their method. More recently, Juvekar et al. propose GAZELLE [117], a system with lower latency (compared to prior work) for secure and private neural network inference. GAZELLE combines homomorphic encryption with traditional two-party computation techniques (such as garbled circuits). With the help of its homomorphic linear algebra kernels, which map neural network operations to optimized



(a) Original Image (b) $\bar{\mathsf{B}} = 0.44$ (c) $\bar{\mathsf{B}} = 1.01$ (d) $\bar{\mathsf{B}} = 1.70$ (e) $\bar{\mathsf{B}} = 1.95$

Figure 5. Visualizing the effect of Cloak on the input images as we increase the average scale of perturbations $\bar{\mathsf{B}}$ that is proportional to the standard deviation. The non-sensitive classification task is smile detection. The first row shows the heat map of perturbation scales for each pixel. At higher scales, Cloak obfuscates the features that are non-conducive to the smile detection task e.g. the background, the hair, etc., while only the lips and some minor facial attributes are recognizable.

homomorphic matrix-vector multiplication and convolutions, GAZELLE is shown to be three orders of magnitude faster than CryptoNets. Sanyal et al. leverage binarized neural networks to speed-up their HE inference method. They claim that unlike CryptoNets which only protects the data, their proposed scheme can protect the privacy of the model as well.

3.3.3. Secure Multi-Party Computation (SMC)

Liu et al. propose MiniONN [116], which uses additively homomorphic encryption (AHE) in a preprocessing step, unlike GAZELLE which uses AHE to speed up linear algebra directly. MiniONN demonstrates a significant performance improvement compared to CryptoNets, without loss of accuracy. However, it is only a two-party computation scheme and does not support computation over multiple parties. Riazi et al. introduce Chameleon, a two-party computation framework whose vector dot product of signed fixed-point numbers improves the efficiency of prediction in classification methods based upon heavy matrix multiplications. Chameleon achieves a 4.2× latency improvement over MiniONN. Most of the efforts in the field of SMC for deep learning are focused on speeding up the computation, as demonstrated above, and also by [102], [103], [113]. The accuracy loss of the aforementioned methods, compared to their pre-trained models is negligible (less than 1%).

3.3.4. Information Theoretic Privacy

Privacy-preserving schemes that rely on information-theoretic approaches usually assume a non-sensitive task, the task that the service is supposed to execute and try to degrade any excessive information in the input data that is not needed for the main inference task. Osia et al. propose Deep Private Feature Extraction (DPFE) [26] which aims at obfuscating input images to hinder the classification of given sensitive (private) labels, by modifying the network topology and re-training all the model parameters. DPFE partitions the network in two partitions, first partition to be deployed on the edge and the second on the cloud. It also modifies the network architecture by adding an auto-encoder in the middle and then re-training the entire network with its loss function. The encoder part of the auto-encoder is deployed on the edge device, and the decoder is deployed on the server. The auto-encoder aims to reduce the dimensions of the sent data which decreases the communication cost, alongside decreasing the amount of information that is sent, which helps the privacy.

DPFE's loss function can be seen in Equation 2. It is composed of three terms, first, the cross-entropy loss for a classification problem consisting of M classes ($y_{o,c}$ indicates whether the observation o belongs to class c and $p_{o,c}$ is the probability given by the network for the observation to belong to class c). This term aims at maintaining accuracy. Second, a term that tries to decrease the distance between intermediate activations of inputs with different private labels, and a final term which tries to increase the distance between intermediate activations of inputs with the same private label. γ is a constant which depends on the number of dimensions and the training data, it is used as a normalization factor. k is also a constant which depends on the training data. i and j are iterators over the main batch and a random batch,

respectively and Y is the private label for that batch member.

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c}) + \gamma(\sum_{(i,j):Y_i \neq Y_j} ||a'_i - a'_j||_2 + \sum_{(i,j):Y_i = Y_j} (k - ||a'_i - a'_j||_2)) \quad (2)$$

DPFE retrains the given neural network and the auto-encoder with this loss function. The training can be seen as an attempt to create clustered representations of data, where the inputs with the same private labels go in different clusters, and inputs with different labels are pushed to the same cluster, to mislead any adversary who tries to infer the private labels. Given its loss function, DPFE cannot operate without the private labels. Therefore, if there is a setting in which no sensitive labels are provided, DPFE cannot be used. After training, for each inference request, a randomly generated noise is added to the intermediate results on the fly. This noise is not there to achieve differential privacy.

More recently, Mireshghallah et al. suggested Shredder [28], a framework that without altering the topology or the weights of a pre-trained network, heuristically learns additive noise distributions that reduce the information content of communicated data while incurring minimal loss to the inference accuracy. Shredder's approach also consists of cutting the neural network and executing a part of it on the edge device, similar to DPFE. This approach has been shown to decrease the overall execution time in some cases [28], compared to running the entire neural network on the cloud, since the communication takes the bulk of time and sensing intermediate representations can sometimes save on the communication since there are fewer dimensions.

Shredder initializes a noise tensor, with the same dimension as the intermediate activation, by sampling from a Laplace distribution with location of 0, and scale of b, which is a hyperparameter. Then, using the loss function shown in Equation 3, it tries to maintain the accuracy of the model (first term), while increasing the amount of additive noise (second term). λ is a knob that provides an accuracy-privacy trade-off.

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c}) - \lambda \sum_{i=1}^{N} |n_i| \quad (3)$$

Once the training is terminated, a Laplace distribution is fit to the trained tensor, and the parameters of that distribution, alongside the order of the elements, are saved. A collection of these distributions are gathered. During inference, noise is sampled from one of the saved distributions and re-ordered to match the saved order. This noise tensor is then added to the intermediate representation, before being sent to the cloud.

Both DPFE and Shredder empirically demonstrate a reduction in the number of mutual information bits between the original data and the sent intermediate representation. DPFE can only be effective if the user knows what s/he wants to protect against, whereas Shredder offers a more general approach that tries to obliterate any information that is irrelevant to the primary task. Empirical evaluations showed that Shredder can in average loose more mutual information, compared to DPFE. However, in the task of inferring private labels, DPFE performs slightly better by causing a higher misclassification rate for the adversary since it has access to the private labels during training time.

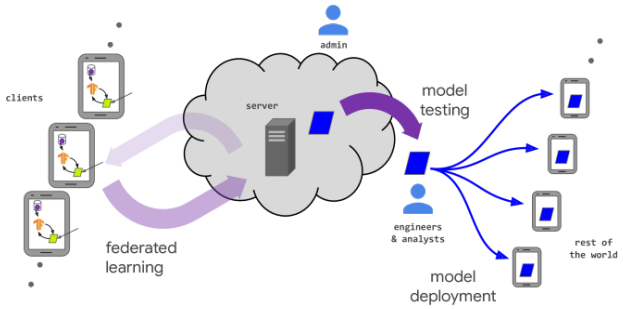


Figure 6. The workflow of federated learning model [105].

4. Privacy-Enhancing Execution Models and Environments

Apart from privacy-preserving schemes which are methods that directly optimize for a given definition of privacy, there are given execution models and environments that help enhance privacy and are not by themselves privacy-preserving. In this section, we will briefly discuss federated learning, split learning and trusted execution environments, which have been used to enhance privacy. These methods are usually accompanied by privacy-preserving schemes from the previous section.

4.1. Federated Learning

Federated learning (FL) is a machine learning setting where many clients collaboratively train a model under the administration of a central server while keeping the training data local. Federated learning is built on the principles of focused collection and data minimization which can alleviate the privacy risks of centralized machine learning [105].

The workflow of federated learning can be seen in Figure 6. This workflow is broken into six stages [105]:

1. Problem identification: The problem that is to be solved using federated learning should first be defined.

2. Client instrumentation: The clients can be instructed to save the data needed for training. For example, the applications running on the edge devices might need to locally save some metadata (e.g. user interaction data) alongside the main data (for instance text messages).

3. Simulation prototyping (optional): The engineer who is deploying the system might need to prototype different architectures and try different hyperparameters in a federated learning simulation.

4. Federated model training: Multiple federated training tasks are initiated which train different variations of the model or use different optimization hyperparameters.

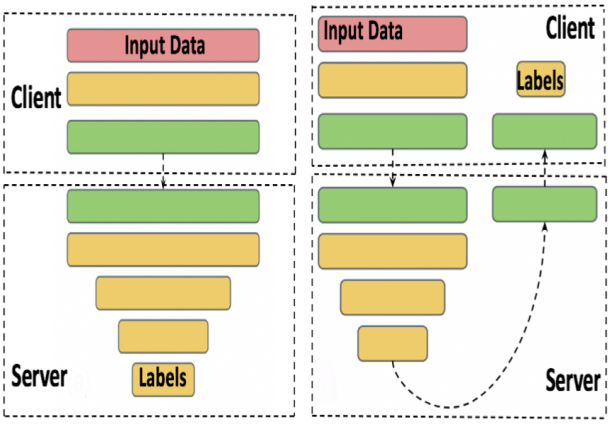
5. Model evaluation: When the tasks are done with the training phase (usually after a few days), the models are analyzed and evaluated, either on standard centralized datasets or on local client data.

6. Deployment: When the analysis is finished and a model is selected, the launch process is initiated. This process consists of live A/B testing, manual quality assurance, and a staged roll-out.

Federated learning is being widely used with SMC and differential Privacy [105, 92, 119].Bonawitz et al. apply **Secure aggregation** to privately combine the outputs of local machine learning on user devices in the federated learning setup, to update a global model. Secure aggregation refers to the computation of a sum in a multiparty setting, where no party reveals its update in the clear, even to the aggregator. When Secure Aggregation is added to Federated Learning, the aggregation of model updates is performed by a virtual incorruptible third party induced by secure multiparty communication. With this setup, the cloud provider learns only the aggregated model update. There are also bodies of work that consider shuffling of user data, so as to hide the origin of each data item.The works of Cheu et al., and Balle et al. have proposed secure aggregation protocols that satisfy differential privacy guarantees in the shuffle model [120, 121]. More recent work [122] mitigates the incurred error and communication overheads in shuffle model. More in-depth details of federated learning workflow and integration is out of the scope of this survey.

4.2. Split Learning

Split-learning is an execution model where the neural network is split, between the client and the server [124]. This is very similar to the neural network partitioning described in Shredder [28] and DPFE [26]. Vanilla split learning is formed by each client computing the forward pass through a deep neural network up to a specific layer, called the cut layer. The outputs of the cut layer, referred to as smashed data, are sent from the edge device to another entity (either the server or another client), which completes the rest of the computation. With this execution scheme, a round of forward pass is computed without sharing raw data. The gradients can then be backpropagated from the server to the cut layer in a similar fashion. The gradients at the cut layer are transferred back to the clients, where the rest of the backpropagation is completed. In this fashion, the training or inference is done without having clients directly access each other's raw data. An instantiation of this setup where labels are also not shared along with raw data is shown in Figure 7.



(a) Vanilla split learning (b) Boomerang split learning

Figure 7. The vanilla configuration of split learning where raw data is not shared between client and server, and boomerang (U-shaped) configuration where neither raw data nor the labels are shared between client and server [123].

4.3. Trusted Execution Environments (TEEs)

Trusted execution environments, also referred to as secure enclaves, provide opportunities to move parts of decentralized learning or inference processes into a trusted environment in the cloud, whose code can be attested and verified. Recently, Mo et al. have suggested a framework that uses an edge device's Trusted Execution Environment (TEE) in conjunction with model partitioning to limit the attack surface against DNNs [125]. TEEs can provide integrity and confidentiality during execution. TEEs have been deployed in many forms, including Intel's SGX-enabled CPUs [126], Arm's TrustZone [127]. This execution model, however, requires the users to send their data to an enclave running on remote servers which allows the remote server to have access to the raw data and as the new breaches in hardware [32, 31, 128, 129, 130, 131] show, the access can lead to comprised privacy.

5. Conclusion

The surge in the use of machine learning is due to the growth in data and compute. The data mostly comes from people [5] and includes an abundance of sensitive information. This work tries to provide a comprehensive and systematic summary of the efforts made to protect privacy of users in deep learning settings. We find an apparent disparity in the number of efforts between data aggregation, training, and inference phases. In particular, little attention has been made to privacy of the users during inference phase.

Acknowledgements

We thank Gary Marcus, Kamalika Chaudhuri, Lawrence Saul, Alex Snoeren and Dean Tullsen for insightful discussions and comments.This work was in part supported by National Science Foundation (NSF) awards CN#1703812, ECCS#1609823, CCF#1553192, Air Force Office of Scientific Research (AFOSR) Young Investigator Program(YIP) award #FA9550-17-1-0274, National Institute of Health (NIH) award #R01EB028350, and Air Force Research Laboratory (AFRL) and Defense Advanced Research Project Agency (DARPA) under agreement number #FA8650-20-2-7009 and #HR0011-18-C-0020. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, AFSOR, NIH, AFRL, DARPA or the U.S. Government.

References

[1] S. Laine, T. Karras, T. Aila, A. Herva, S. Saito, R. Yu, H. Li, and J. Lehtinen, "Production-level facial performance capture using deep convolutional neural networks," in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, (New York, NY, USA), Association for Computing Machinery, 2017.

[2] V. Këpuska and G. Bohouta, "Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home)," *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 99–103, 2018.

[3] 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 *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC'14, (USA), p. 17–32, USENIX Association, 2014.

[4] R. J. Bolton and D. J. Hand, "Statistical fraud detection: A review," *Statist. Sci.*, vol. 17, pp. 235–255, 08 2002.

[5] S. A. Thompson and C. Warzel, "The privacy project: Twelve million phones, one dataset, zero privacy," 2019. online accessed February 2020 https://www.nytimes.com/interactive/2019/12/19/opinion/location-tracking-cell-phone.html.

[6] C. Warzel, "The privacy project: Faceapp shows we care about privacy but don't understand it," 2019. online accessed February 2020 https://www.nytimes.com/2019/07/18/opinion/faceapp-privacy.html.

[7] A. Newcomb, "Facebook data harvesting scandal widens to 87 million people," 2018. online accessed February 2020 https://www.nbcnews.com/tech/tech-news/facebook-data-harvesting-scandal-widens-87-million-people-n862771.

[8] A. Frome, G. Cheung, A. Abdulkader, M. Zennaro, B. Wu, A. Bissacco, H. Adam, H. Neven, and L. Vincent, "Large-scale privacy protection in google street view," *2009 IEEE 12th International Conference on Computer Vision*, pp. 2373–2380, 2009.

[9] J. Schiff, M. Meingast, D. K. Mulligan, S. S. Sastry, and K. Goldberg, "Respectful cameras: Detecting visual markers in real-time to address privacy concerns," in *Protecting Privacy in Video Surveillance*, 2009.

[10] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Ying-Li Tian, A. Ekin, J. Connell, Chiao Fe Shu, and M. Lu, "Enabling video privacy through computer vision," *IEEE Security Privacy*, vol. 3, no. 3, pp. 50–57, 2005.

[11] L. Cheng, F. Liu, and D. D. Yao, "Enterprise data breach: causes, challenges, prevention, and future directions," *WIREs Data Mining and Knowledge Discovery*, vol. 7, no. 5, p. e1211, 2017.

[12] T. Armerding, "The 18 biggest data breaches of the 21st century," 2018. online accessed February 2020 https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html.

[13] R. Shokri, M. Stronati, and V. Shmatikov, "Membership inference attacks against machine learning models," *CoRR*, vol. abs/1610.05820, 2016.

[14] 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*, CCS '15, (New York, NY, USA), p. 1322–1333, Association for Computing Machinery, 2015.

[15] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282, July 2018.

[16] M. Yan, C. W. Fletcher, and J. Torrellas, "Cache telepathy: Leveraging shared resource attacks to learn dnn architectures," *ArXiv*, vol. abs/1808.04761, 2018.

[17] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," *CoRR*, vol. abs/1609.02943, 2016.

[18] 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*, CCS '18, (New York, NY, USA), p. 619–633, Association for Computing Machinery, 2018.

[19] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[20] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, (Berlin, Heidelberg), pp. 265–284, Springer-Verlag, 2006.

[21] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, (New York, NY, USA), p. 1310–1321, Association for Computing Machinery, 2015.

[22] 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*, CCS '16, (New York, NY, USA), p. 308–318, Association for Computing Machinery, 2016.

[23] N. Papernot, M. Abadi, Úlfar Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," 2016.

[24] J. Hamm, P. Cao, and M. Belkin, "Learning privately from multiparty data," *CoRR*, vol. abs/1602.03552, 2016.

[25] E. Hesamifard, H. Takabi, and M. Ghasemi, "Cryptodl: Deep neural networks over encrypted data," *CoRR*, vol. abs/1711.05189, 2017.

[26] S. A. Ossia, A. Taheri, A. S. Shamsabadi, K. Katevas, H. Haddadi, and H. R. Rabiee, "Deep private-feature extraction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, pp. 54–66, 2018.

[27] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 201–210, JMLR.org, 2016.

[28] F. Mireshghallah, M. Taram, A. Jalali, D. Tullsen, and H. Esmaeilzadeh, "Shredder: Learning noise distributions to protect inference privacy," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '20, (New York, NY, USA), Association for Computing Machinery, March 2020.

[29] I. Security, "Data exfiltration study: actors, tactics, and detection (2015)," 2015.

[30] C. Warzel, "Chinese hacking is alarming. so are data brokers.," 2020. online accessed February 2020 https://www.nytimes.com/2020/02/10/opinion/equifax-breach-china-hacking.html.

[31] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown: Reading kernel memory from user space," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018.

[32] P. Kocher, J. Horn, A. Fogh, , D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in *40th IEEE Symposium on Security and Privacy (S&P'19)*, 2019.

[33] A. Whitten and J. D. Tygar, "Why johnny can't encrypt: A usability evaluation of pgp 5.0," in *Proceedings of the 8th Conference on USENIX Security Symposium - Volume 8*, SSYM'99, (USA), p. 14, USENIX Association, 1999.

[34] R. Unuchek, "Leaking ads – is user data truly secure?," April 2018. online accessed February 2020 https://published-prd.lanyonevents.com/published/rsaus18/sessionsFiles/8161/ASEC-T08-Leaking-Ads-Is-User-Data-Truly-Secure.pdf.

[35] Y. Long, V. Bindschaedler, and C. A. Gunter, "Towards measuring membership privacy," *ArXiv*, vol. abs/1712.09136, 2017.

[36] A. Salem, Y. Zhang, M. Humbert, M. Fritz, and M. Backes, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," *ArXiv*, vol. abs/1806.01246, 2018.

[37] C. Song and V. Shmatikov, "The natural auditor: How to tell if someone used your words to train their model," *ArXiv*, vol. abs/1811.00513, 2018.

[38] J. Hayes, L. Melis, G. Danezis, and E. D. Cristofaro, "LOGAN: evaluating privacy leakage of generative models using generative adversarial networks," *CoRR*, vol. abs/1705.07663, 2017.

[39] Z. Yang, E.-C. Chang, and Z. Liang, "Adversarial neural network inversion via auxiliary knowledge alignment," *ArXiv*, vol. abs/1902.08552, 2019.

[40] A. Salem, A. Bhattacharyya, M. Backes, M. Fritz, and Y. Zhang, "Updates-leak: Data set inference and reconstruction attacks in online learning," *CoRR*, vol. abs/1904.01067, 2019.

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

[42] G. Ateniese, G. Felici, L. V. Mancini, A. Spognardi, A. Villani, and D. Vitali, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *CoRR*, vol. abs/1306.4447, 2013.

[43] B. Wang and N. Z. Gong, "Stealing hyperparameters in machine learning," *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 36–52, 2018.

[44] A. Sablayrolles, M. Douze, Y. Ollivier, C. Schmid, and H. Jégou, "White-box vs black-box: Bayes optimal strategies for membership inference," 2019.

[45] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei, "Towards demystifying membership inference attacks," *ArXiv*, vol. abs/1807.09173, 2018.

[46] D. Arpit, S. Jastrzundefinedbski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and et al., "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, p. 233–242, JMLR.org, 2017.

[47] X. Wu, M. Fredrikson, S. Jha, and J. F. Naughton, "A methodology for formalizing model-inversion attacks," in *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, pp. 355–370, June 2016.

[48] C. Huang, P. Kairouz, X. Chen, L. Sankar, and R. Rajagopal, "Context-aware generative adversarial privacy," *CoRR*, vol. abs/1710.09549, 2017.

[49] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pp. 111–125, May 2008.

[50] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays," *PLoS genetics*, vol. 4, no. 8, 2008.

[51] C. C. Aggarwal, "On k-anonymity and the curse of dimensionality," in *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, p. 901–909, VLDB Endowment, 2005.

[52] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," *ACM Trans. Knowl. Discov. Data*, vol. 1, p. 3–es, Mar. 2007.

[53] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115, IEEE, 2007.

[54] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques*, EUROCRYPT'06, (Berlin, Heidelberg), pp. 486–503, Springer-Verlag, 2006.

[55] M. Lécuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 656–672, 2018.

[56] C. Dwork, G. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS '10)*, (Las Vegas, NV), p. 51–60, IEEE, IEEE, 23–26 October 2010.

[57] S. Oh and P. Viswanath, "The composition theorem for differential privacy," *CoRR*, vol. abs/1311.0776, 2013.

[58] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, pp. 211–407, Aug. 2014.

[59] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," *CoRR*, vol. abs/1407.1338, 2014.

[60] Ú. Erlingsson, A. Korolova, and V. Pihur, "RAPPOR: randomized aggregatable privacy-preserving ordinal response," *CoRR*, vol. abs/1407.6981, 2014.

[61] A. Triastcyn and B. Faltings, "Generating differentially private datasets using gans," *CoRR*, vol. abs/1803.03148, 2018.

[62] D. Kifer and A. Machanavajjhala, "Pufferfish: A framework for mathematical privacy definitions," *ACM Trans. Database Syst.*, vol. 39, Jan. 2014.

[63] Y. Wang, S. Song, and K. Chaudhuri, "Privacy-preserving analysis of correlated data," *CoRR*, vol. abs/1603.03977, 2016.

[64] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of computer and system sciences*, vol. 28, no. 2, pp. 270–299, 1984.

[65] K. Nissim and A. Wood, "Is privacy privacy?," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 376, no. 2128, p. 20170358, 2018.

[66] J. Liu, L. Xiong, and J. Luo, "Semantic security: Privacy definitions revisited.," *Trans. Data Privacy*, vol. 6, no. 3, pp. 185–198, 2013.

[67] M. Diaz, H. Wang, F. P. Calmon, and L. Sankar, "On the robustness of information-theoretic privacy measures and mechanisms," *CoRR*, vol. abs/1811.06057, 2018.

[68] A. Pinceti, O. Kosut, and L. Sankar, "Data-driven generation of synthetic load datasets preserving spatio-temporal features," in *2019 IEEE Power Energy Society General Meeting (PESGM)*, pp. 1–5, Aug 2019.

[69] J. Liao, P. Kairouz, C. Huang, and L. Sankar, "Learning generative adversarial representations under fairness and censoring constraints," 2019.

[70] D. P. Varodayan and A. Khisti, "Smart meter privacy using a rechargeable battery: Minimizing the rate of information leakage," *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1932–1935, 2011.

[71] H. L. Hsu, S. Asoodeh, and F. du Pin Calmon, "Obfuscation via information density estimation," *ArXiv*, vol. abs/1910.08109, 2019.

[72] A. Li, J. Guo, H. Yang, and Y. Chen, "Deepobfuscator: Adversarial training framework for privacy-preserving image classification," *ArXiv*, vol. abs/1909.04126, 2019.

[73] V. Mirjalili, S. Raschka, and A. Ross, "Flowsan: Privacy-enhancing semi-adversarial networks to confound arbitrary face-based gender classifiers," *IEEE Access*, vol. 7, pp. 99735–99745, 2019.

[74] P. C. Roy and V. N. Boddeti, "Mitigating information leakage in image representations: A maximum entropy approach," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2581–2589, 2019.

[75] Z. Wu, Z. Wang, Z. Wang, and H. Jin, "Towards privacy-preserving visual recognition via adversarial training: A pilot study," *ArXiv*, vol. abs/1807.08379, 2018.

[76] C. Xu, J. Ren, D. Zhang, Y. Zhang, Z. Qin, and K. Ren, "Ganobfuscator: Mitigating information leakage under gan via differential privacy," *IEEE Transactions on Information Forensics and Security*, vol. 14, pp. 2358–2371, 2019.

[77] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private language models without losing accuracy," *CoRR*, vol. abs/1710.06963, 2017.

[78] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *Advances in Neural Information Processing Systems 21* (D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, eds.), pp. 289–296, Curran Associates, Inc., 2009.

[79] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, p. 1069–1109, July 2011.

[80] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," *CoRR*, vol. abs/1802.06739, 2018.

[81] G. Ács, L. Melis, C. Castelluccia, and E. D. Cristofaro, "Differentially private mixture of generative neural networks," *CoRR*, vol. abs/1709.04514, 2017.

[82] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: An application of human behavior prediction," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, p. 1309–1316, AAAI Press, 2016.

[83] N. Phan, X. Wu, and D. Dou, "Preserving differential privacy in convolutional deep belief networks," *Mach. Learn.*, vol. 106, p. 1681–1704, Oct. 2017.

[84] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," *CoRR*, vol. abs/1709.05750, 2017.

[85] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Úlfar Erlingsson, "Scalable private learning with pate," 2018.

[86] L. Zhao, Y. Zhang, Q. Wang, Y. Chen, C. Wang, and Q. Zou, "Privacy-preserving collaborative deep learning with irregular participants," *CoRR*, vol. abs/1812.10113, 2018.

[87] B. K. Beaulieu-Jones, W. Yuan, S. G. Finlayson, and Z. S. Wu, "Privacy-preserving distributed deep learning for clinical data," *CoRR*, vol. abs/1812.01484, 2018.

[88] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," *CoRR*, vol. abs/1904.02200, 2019.

[89] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *CoRR*, vol. abs/1712.07557, 2017.

[90] A. Bhowmick, J. C. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," *ArXiv*, vol. abs/1812.00984, 2018.

[91] M. Chase, R. Gilad-Bachrach, K. Laine, K. E. Lauter, and P. Rindal, "Private collaborative neural network learning," *IACR Cryptology ePrint Archive*, vol. 2017, p. 762, 2017.

[92] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, (New York, NY, USA), p. 1175–1191, Association for Computing Machinery, 2017.

[93] N. Agrawal, A. S. Shamsabadi, M. J. Kusner, and A. Gascón, "QUOTIENT: two-party secure neural network training and prediction," *CoRR*, vol. abs/1907.03372, 2019.

[94] S. Wagh, D. Gupta, and N. Chandran, "Securenn: 3-party secure computation for neural network training," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, pp. 26 – 49, 2019.

[95] P. Mohassel and P. Rindal, "Aby3: A mixed protocol framework for machine learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, (New York, NY, USA), p. 35–52, Association for Computing Machinery, 2018.

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

[97] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, pp. 1333–1345, May 2018.

[98] T. Graepel, K. Lauter, and M. Naehrig, "Ml confidential: Machine learning on encrypted data," in *Information Security and Cryptology – ICISC 2012* (T. Kwon, M.-K. Lee, and D. Kwon, eds.), (Berlin, Heidelberg), pp. 1–21, Springer Berlin Heidelberg, 2013.

[99] L. J. M. Aslett, P. M. Esperança, and C. C. Holmes, "Encrypted statistical machine learning: new privacy preserving methods," 2015.

[100] R. Yonetani, V. N. Boddeti, K. M. Kitani, and Y. Sato, "Privacy-preserving visual learning using doubly permuted homomorphic encryption," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2059–2069, 2017.

[101] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *In Proc. STOC*, pp. 169–178, 2009.

[102] M. S. Riazi, M. Samragh, H. Chen, K. Laine, K. Lauter, and F. Koushanfar, "Xonn: Xnor-based oblivious deep neural network inference," in *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, (USA), p. 1501–1518, USENIX Association, 2019.

[103] E. Makri, D. Rotaru, N. P. Smart, and F. Vercauteren, "Epic: Efficient private image classification (or: Learning from the masters)," in *CT-RSA*, 2019.

[104] A. S. Shamsabadi, A. Gascón, H. Haddadi, and A. Cavallaro, "Privedge: From local to distributed private training and prediction," *ArXiv*, vol. abs/2004.05574, 2020.

[105] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, S. E. Rouayheb, D. Evans, J. Gardner, Z. A. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, O. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. X. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *ArXiv*, vol. abs/1912.04977, 2019.

[106] J. Wang, J. Zhang, W. Bao, X. Zhu, B. Cao, and P. S. Yu, "Not just privacy," *Proceedings of the 24th ACM*

SIGKDD International Conference on Knowledge Discovery and Data Mining, Jul 2018.

[107] F. Mireshghallah, M. Taram, A. Jalali, A. T. Elthakeb, D. M. Tullsen, and H. Esmaeilzadeh, "A principled approach to learning stochastic representations for privacy in deep neural inference," *ArXiv*, vol. abs/2003.12154, 2020.

[108] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," Tech. Rep. MSR-TR-2016-3, February 2016.

[109] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," *IACR Cryptology ePrint Archive*, vol. 2017, p. 35, 2017.

[110] A. Sanyal, M. J. Kusner, A. Gascón, and V. Kanade, "TAPAS: tricks to accelerate (encrypted) prediction as a service," *CoRR*, vol. abs/1806.03461, 2018.

[111] F. Bourse, M. Minelli, M. Minihold, and P. Paillier, "Fast homomorphic evaluation of deep discretized neural networks," in *CRYPTO*, 2017.

[112] V. N. Boddeti, "Secure face matching using fully homomorphic encryption," *ArXiv*, vol. abs/1805.00577, 2018.

[113] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," *CoRR*, vol. abs/1705.08963, 2017.

[114] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," *CoRR*, vol. abs/1801.03239, 2018.

[115] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, "Machine learning classification over encrypted data," *IACR Cryptology ePrint Archive*, vol. 2014, p. 331, 2014.

[116] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via minionn transformations," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, (New York, NY, USA), p. 619–631, Association for Computing Machinery, 2017.

[117] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: A low latency framework for secure neural network inference," in *Proceedings of the 27th USENIX Conference on Security Symposium*, SEC'18, (USA), p. 1651–1668, USENIX Association, 2018.

[118] P. Mishra, R. T. Lehmkuhl, A. Srinivasan, W. Zheng, and R. A. Popa, "Delphi: A cryptographic inference service for neural networks," *IACR Cryptology ePrint Archive*, vol. 2020, p. 50, 2020.

[119] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," in *CCS '17*, 2017.

[120] A. Cheu, A. D. Smith, J. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," *IACR Cryptology ePrint Archive*, vol. 2019, p. 245, 2019.

[121] B. Balle, J. Bell, A. Gascón, and K. Nissim, "The privacy blanket of the shuffle model," in *CRYPTO*, 2019.

[122] B. Ghazi, R. Pagh, and A. Velingker, "Scalable and differentially private distributed aggregation in the shuffled model," *ArXiv*, vol. abs/1906.08320, 2019.

[123] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," *ArXiv*, vol. abs/1812.00564, 2018.

[124] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *J. Netw. Comput. Appl.*, vol. 116, pp. 1–8, 2018.

[125] F. Mo, A. S. Shamsabadi, K. Katevas, S. Demetriou, I. Leontiadis, A. Cavallaro, and H. Haddadi, "Darknetz: Towards model privacy at the edge using trusted execution environments," *ArXiv*, vol. abs/2004.05703, 2020.

[126] V. Costan and S. Devadas, "Intel sgx explained," *IACR Cryptology ePrint Archive*, vol. 2016, p. 86, 2016.

[127] A. inc., "Arm trustzone technology." online accessed February 2020 https://developer.arm.com/ip-products/security-ip/trustzone.

[128] O. Weisse, J. Van Bulck, M. Minkin, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, R. Strackx, T. F. Wenisch, and Y. Yarom, "Foreshadow-NG: Breaking the virtual memory abstraction with transient out-of-order execution," *Technical report*, 2018.

[129] C. Canella, D. Genkin, L. Giner, D. Gruss, M. Lipp, M. Minkin, D. Moghimi, F. Piessens, M. Schwarz, B. Sunar, J. Van Bulck, and Y. Yarom, "Fallout: Leaking data on meltdown-resistant cpus," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, ACM, 2019.

[130] M. Taram, A. Venkat, and D. M. Tullsen, "Packet chasing: Spying on network packets over a cache side-channel," *ArXiv*, vol. abs/1909.04841, 2019.

[131] M. Taram, A. Venkat, and D. Tullsen, "Context-sensitive fencing: Securing speculative execution via microcode customization," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19, (New York, NY, USA), p. 395–410, Association for Computing Machinery, 2019.