TRANSACTIONS ON DATA PRIVACY 11 (2018) 61–79

Membership Inference Attack against Differentially Private Deep Learning Model

Md Atiqur Rahman[*], **Tanzila Rahman**[**], **Robert Laganière**[*], **Noman Mohammed**[**], **Yang Wang**[**]

[*]School of Electrical Engineering and Computer Science, University of Ottawa, ON, K1N 6N5, Canada.

[**]Department of Computer Science, University of Manitoba, MB, R3T2N2, Canada.

E-mail: mrahm021@uottawa.ca, rahmant4@myumanitoba.ca, laganier@eecs.uottawa.ca, noman@cs.umanitoba.ca, ywang@cs.umanitoba.ca

Received 7 August 2017; received in revised form 16 January 2018; accepted 7 February 2018

Abstract. The unprecedented success of deep learning is largely dependent on the availability of massive amount of training data. In many cases, these data are crowd-sourced and may contain sensitive and confidential information, therefore, pose privacy concerns. As a result, privacy-preserving deep learning has been gaining increasing focus nowadays. One of the promising approaches for privacy-preserving deep learning is to employ differential privacy during model training which aims to prevent the leakage of sensitive information about the training data via the trained model. While these models are considered to be immune to privacy attacks, with the advent of recent and sophisticated attack models, it is not clear how well these models trade-off utility for privacy. In this paper, we systematically study the impact of a sophisticated machine learning based privacy attack called the membership inference attack against a state-of-the-art differentially private deep model. More specifically, given a differentially private deep model with its associated utility, we investigate how much we can infer about the model's training data. Our experimental results show that differentially private deep models may keep their promise to provide privacy protection against strong adversaries by only offering poor model utility, while exhibit moderate vulnerability to the membership inference attack when they offer an acceptable utility. For evaluating our experiments, we use the CIFAR-10 and MNIST datasets and the corresponding classification tasks.

Keywords. differential privacy, membership inference attack, deep learning, privacy-preserving deep learning, differentially private deep learning

1 Introduction

Deep learning based on deep neural networks (DNNs) has led to impressive success in a wide variety of applications like image classification [1], face recognition [2], medical diagnoses [3] and so on. These advances have been made possible due to the availability of large amount of data used to train the deep learning models. Moreover, the accuracy and utility of deep learning models are undeniable. As a result, giant companies like Google, Facebook, Amazon, Apple etc. are taking advantage of their large user base in order to gather data from the users to form massive amount of training data and deploy deep learning on a large scale. This huge amount of data often contain sensitive and confidential user information which may be vulnerable to serious privacy risks. This is because, companies usually store these data indefinitely and users from whom the data are collected have no control on the data [4]. Moreover, recording of images and voices may accidentally capture sensitive user information such as, faces, license plates, computer screens etc. [5]. Therefore, as more and more companies engage in incorporating these "big data" to train large scale deep models, it raises more and more concerns about the privacy of the users the data are collected from.

For many domains, machine learning applications can bring great benefits only through the analysis of sensitive user data. For example, in healthcare industry, medical records are used by hospitals or insurance providers to learn machine learning models estimating individual risk towards certain diseases [6]. Personalized recommendation systems are other good examples where efficiency is driven by the use of sensitive user data. Deep learning algorithms usually employ different regularization techniques (e.g., l_2-regularization, dropout etc.) to prevent overfitting of the models to the training examples. As a side effect, these regularization techniques help the deep models protect the privacy of the training data. But, the enormous learning capacity of the deep models make such attempts insufficient to prevent the trained models from implicitly memorizing details of at least a few specific training examples, if not all. This was aptly demonstrated by Fredrikson et al [7] who were successful in launching a model-inversion attack against a deep model to reproduce recognizable faces belonging to the training data of a face recognition system.

To address the privacy concerns in deep learning, privacy preserving deep learning (PPDL) has been gaining popularity recent days. The core idea of PPDL is to make it ideally impossible for an adversary to gain access to the training data via the trained model. While most PPDL approaches, for the sake of privacy, cannot afford to share the trained model publicly, differential privacy (DP) [8] based approaches that employ differential privacy during model training come with the promise of providing protection against a strong adversary who has full knowledge of and access to the model parameters. As a result, differentially private deep models (DPDMs) are assumed to be privacy protected even when the models are shared publicly. A recent work by Abadi et al. [9] introduces one such DPDM which is state-of-the-art and offers modest privacy guarantee while still providing moderate utility.

While machine learning has brought countless benefits, its adversarial use has also seen notable success. Recent sophisticated attack models has been successful in turning machine learning against itself with a view to leaking sensitive information contained in the target model's training data. One such sophisticated attack is the membership inference attack proposed by Shokri et al. [10] that trains an attack model to recognize the differences in the behavior of a target model on inputs that come from the target model's training data versus inputs that the target model did not encounter during training. Simply put, given an input record and a target model, membership inference attack tries to determine if the record was used during training of the target model or not. This kind of attack can directly lead to privacy breach. For example, knowing that a patient's record was used to train a certain machine learning model that is used to determine the appropriate medicine dosage for a certain disease or discover the genetic basis of the disease can be sufficient to conclude that the patient has that disease [10].

1.1 Our Contribution

In the wake of such sophisticated attack models, it becomes a consequential query to ask, "how well DPDM's promise withstand the membership inference attack?". In this study, we try to answer this question by launching the membership inference attack against a state-of-the-art DPDM proposed by Abadi et al. [9]. Although membership inference attack has been launched against several deep models in a black-box setting [10], to the best of our knowledge, this is the first attempt to study its effect on DPDM in a white-box setting. In a black-box setting, an adversary can only query a model without having any access to the model parameters, whereas, adversaries have access to the model parameters in a white-box setting. Our experimental results show that DPDMs offering higher utility levels are vulnerable to the membership inference attack. This vulnerability, in certain cases, can reach as close to that as exhibited by the totally non-private models. We also quantify the vulnerability of the DPDMs using attack accuracy and F_1-score as the performance metrics. Moreover, since DP has always been criticized for not being able to provide an intuitive explanation of the privacy parameter ε, this study also aims at providing an explanation of how much protection the privacy parameter ε really provides with respect to the membership inference attack and also suggests optimal values for ε that may offer a good trade-off between utility and privacy for deep models.

The rest of the paper is organized as follows. In Section 2, we review the literature related to the study. We describe the membership inference attack and the differentially private deep learning technique in Section 3. In Section 4, we give details about the experimental setup, datasets used as well as the different experiments conducted, whereas, experimental results are discussed in Section 5. Finally, we conclude by mentioning possible future scope of this work in Section 6.

2 Related Works

Our work is mostly related to research in two directions, one involves designing differentially private deep learning algorithms, while the other direction is about attacks on machine learning models. In the following subsections, we review recent literature from each of these areas.

2.1 Differentially Private Deep Learning

Deep learning itself being a relatively new technique, little focus has been given to its privacy concerns. In response to recent attacks against deep models, privacy preserving deep learning has drawn considerable attention from the deep learning as well as data privacy research communities, and DP is in the forefront of research in this direction. DP has mostly been applied to shallow models, for example, collaborative recommender systems by McSherry et al. [11], logistic regression by Chaudhuri et al. [12], or empirical risk minimization by Bassily et al. [13]. Very recently, there have been some approaches proposed that employ DP in deep learning. For example, Phan et al. [14] focuses on learning differentially private deep auto-encoders by perturbing the objective functions of the deep network. While the authors successfully used this approach for human behavior prediction, it lacks the ability to generalize for other types of deep networks and also do not provide meaningful privacy guarantee.

Recent works on privacy preserving deep learning has explored possibilities to incorporate DP in Stochastic Gradient Descent (SGD), a very effective optimization technique for deep learning algorithms. One such approach called Distributed Selective SGD (DSSGD) was proposed by Shokri and Shmatikov [4]. They designed a distributed training mechanism for deep networks among multiple private data owners. Sharing of the gradients during SGD is made differentially private by using additive-noise mechanism. Although

Methods	Model Type			Privacy Guarantee		
	Shallow	Deep	Logistic Regression	Limited	Modest	Full
Collaborative Recommender Systems [11]	✓					✓
Linear Classifier [12]			✓			✓
Empirical Risk Minimization [13]	✓					✓
Differentially Private Deep Auto-encoders [14]		✓		✓		
Distributed Selective SGD (DSSGD) [4]		✓		✓		
Differentially Private SGD (DPSGD) [9]		✓			✓	
PATE [15]		✓				✓

Table 1: Comparison among different differentially private machine learning models based on the privacy guarantee they offer. Here, “Limited”, “Modest” and “Full” represent an ε guarantee of >8, 2–8, and <2, respectively.

the utility of DSSGD is impressive, it can provide privacy guarantee of $\varepsilon > 2$ for each of the 300,000 parameters of the deep model, thereby naively making the total $\varepsilon >$600,000, and thus, fails to provide any meaningful privacy guarantee.

Recently, Abadi et al. [9] proposed a differentially private version of SGD called DPSGD that is able to offer a substantially stricter privacy bound for the whole deep model while still delivering comparable utility. To achieve a tighter bound, they introduced a state-of-the-art privacy accountant method called moments accountant which keeps track of the privacy budget spending during training. This work is particularly important for its flexibility that makes it applicable for any type of machine learning algorithm that employs SGD as the optimization technique.

Very recently, there has been another approach proposed to provide an even more tighter privacy bound with better utility [15]. Based on the original idea of knowledge aggregation and transfer [16], this approach basically learns an ensemble of teacher models each based on a disjoint set of private and sensitive data. These teachers are then used to train a student model in a way so that it can accurately mimic the ensemble. But the major limitation of this approach is that it requires non-private unlabeled data that have similar distribution as the private data which may not be always available. Table 1 shows comparison among different differentially private machine learning techniques with respect to the privacy guarantees they offer.

In this paper, to study the impact of membership inference attack on differentially private deep models, we choose the approach proposed in [9] as the target DPDM model mainly because, it is versatile, requires no extra data and provides state-of-the-art privacy and utility trade-off.

2.2 Attacks on Machine Learning Models

Several studies show that machine learning models can leak various types of sensitive information contained in the training data. In [17], comparison of published statistics about minor allele frequencies with the distribution of these statistics in the general population were used to infer the presence of a particular genome in the training dataset. Calandrino et al. [18] introduced an attack against a collaborative recommender system, where changes in the outputs of the system are exploited by an adversary in order to infer inputs that caused these changes. Ateniese et al. [19] reported that knowledge about the parameters of machine learning models such as SVM and HMM can be used to infer general statistical information about the training dataset.

Recent trends for attack models are inspired by the adversarial use of machine learning.

A number of sophisticated attack models have been proposed that actually turn machine learning against itself. A well-known attack of this kind is the model-inversion attack. Given a model and a known output, the model inversion attack tries to infer certain hidden feature(s) (usually sensitive) of the input corresponding to the output. To this end, the attacker actually trains a machine learning model that tries to learn the hidden feature(s) by minimizing the error between the predicted output and the known output. This is done simply by following the gradients of the error with respect to the hidden input feature(s) with a view to adjusting the hidden feature(s) so as to minimize the error. As a result of this reverse-engineering, the attacker can recover prototypical examples from the training set. In fact, it has been shown that these prototypical examples simply correspond to average of the input features that characterize the corresponding output and do not represent any real member of the training set [10].

One example of model inversion attack was presented by Fredrikson et al. in [7] that showed a case study on pharmacogenetics analysis where the correlation between a patient's genotype and the dosage of a certain medicine were captured by training an adversarial model with a view to leaking disease information about the patient. In a more aggressive attack along the line, Fredrikson et al. [7] were successful in reconstructing the face of an individual belonging to the training data of a face recognition system, provided that the particular face label is given.

Among other attacks that make adversarial use of machine learning include the model extraction attack [21]. The aim of this attack is to extract important parameters of a model often trained on private and sensitive data and use these parameters to train an adversarial model to mimic the performance of the original model. Another recent work by Sharif et al. [22] is able to mislead a biometric facial recognition system. It trains a state-of-the-art deep face recognition model that allows the adversary to evade being recognized or to impersonate another individual.

While most of the machine learning based attacks discussed above are either limited by the application domain (e.g., model inversion was reported not to work for tasks other than face recognition, model extraction is only suitable for specific types of models), or require substantial auxiliary information, a very novel and sophisticated attack proposed by Shokri et al. [10] applies ideally to any type of machine learning models including deep models and can even work in a black-box setting. The authors call it membership inference attack where the goal of an adversary, given an input record and a target model, is to determine if the input record comes from the training data of the target model or not. The work presented in this paper is totally based on this attack model details about which is discussed in the next section.

3 Methodology

In the following subsections, we give details about the membership inference attack [10] as well as the differentially private deep learning technique [9].

3.1 Membership Inference Attack

The membership inference attack capitalizes on the fact that machine learning models often behave very differently on the training data than test data (i.e., data that were never seen before). Among others, over-fitting is considered to be the main reason behind this

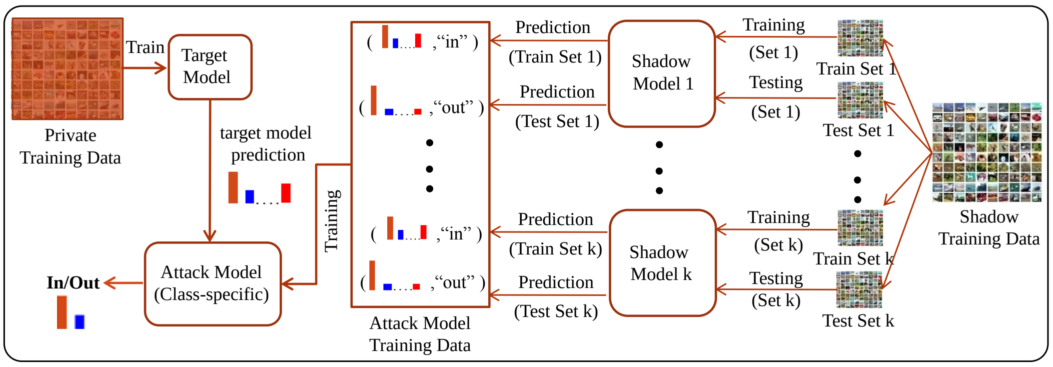


Figure 1: Overview of the membership inference attack. For each shadow model, the output prediction vectors on its training dataset are labeled as “in”, whereas prediction vectors on the test set are labeled as “out”. These prediction vectors along with their associated class labels (“in”/“out”) are used to train n class-specific attack models, where n is the total number of output classes in the target model. Finally, the adversary queries the target model with a target record and passes the prediction vector to the corresponding attack model which emits the membership probabilities (“in” and “out” probabilities) for the target record in the target model’s training dataset.

phenomenon which almost all machine learning models fall prey to. Therefore, an adversary can train a machine learning model to recognize such differences in the target model’s behavior and then use that model to determine if an input record was used to train the target model or not.

For the purpose of the attack, we assume that the target model is a classification model which, given an input record, emits prediction vector of class probabilities, one for each output class. Since our target model is the DPDM proposed in [9] which is meant to be public, the adversary has full access to the target model’s architecture as well as input and output formats. We also assume that the adversary may have some background knowledge about the population the target models’ training dataset was drawn from. This should not be a limiting factor, as the adversary has access to the target model’s input and output format, and therefore, can independently draw samples from the same population.

As stated in the original paper [10], the attack model is basically a collection of several models, one for each output class. Doing so increases accuracy as the target model’s distribution over the output classes depends heavily on the input’s true class [10]. To begin with, the adversary first draws samples independently from the same population where the target model’s training data are drawn from. These sampled data are then used to train a set of models which are called shadow models. The shadow models are expected to mimic the behavior of the target model as the target model and the shadow models are all trained on data coming from the same population. Finally, the output prediction vectors of the shadow models on both training and test sets are combined to form a large training dataset which is used to train a set of attack models, one for each output class. These class-specific attack models are learned in a way so that they can distinguish shadow models’ outputs corresponding to inputs that are part of models’ training datasets from outputs corresponding to inputs that are not part of the training datasets (e.g., inputs coming from models’ test datasets). The overview of the membership inference attack is illustrated in

Fig. 1.

More formally, let $f_{target}()$ denote the target model, and D^{train}_{target} denote its private training dataset which contains labeled data records denoted as $(x^i_{target}, y^i_{target})$. Let the total number of classes be denoted by c_{target}. Therefore, the true class label y^i_{target} takes on a value from a set of classes of size c_{target} and the prediction probability vector $\mathbf{y}$ as output by the target model is of size c_{target}. Let $f_{attack}()$ denote the attack model. Since the objective of the attack is to determine the membership of an input record in the target model's training dataset, $f_{attack}()$ turns out to be a binary classifier. Therefore, given an input record (x, y), $f_{attack}()$ outputs probabilities $Pr\{(x, y) \in D^{train}_{target}\}$ for two classes, namely "in" and "out" representing the membership and non-membership of the input record (x, y) in D^{target}_{train}, respectively. In the following subsections, we give details about the shadow models and the attack models.

3.1.1 Shadow Models

As mentioned in the previous subsection, the adversary trains k shadow models, each on a dataset that is similar in format and distribution as the target model's private training set. While the individual shadow datasets may overlap, there is of course, no assumption that the target model's training and test datasets and the shadow datasets overlap. An adversary can generate shadow data using any of the three techniques as described in Section VI-C in the original paper [10]. One such technique is for the adversary to independently draw samples from the same population where the target model's training data are drawn from. For example, if the target model is a digit classifier, the adversary knows that the target model's training data come from the population of digit recognition data. He may therefore, draw samples independently from digit recognition datasets which may not necessarily overlap with the target model's training data. For the purpose of this study, we use this technique to generate shadow data. One requirement for training the shadow models is that they must be trained in a similar fashion as the target model. In our case, this requirement can be easily satisfied as the target model is a DPDM which is meant to be shared publicly. As a result, the adversary knows the target model's type and architecture as well as the training algorithm used. The larger the number of shadow models, the better the accuracy of the attack model [10].

3.1.2 Attack Models

The adversary queries each shadow model with its own disjoint training and test datasets which are of the same size. The outputs on the training set are labeled as "in" (i.e., the record is in shadow model's training set), whereas, those on the test set are labeled as "out" (i.e., the record is not in the shadow model's training set). As illustrated in Fig. 1, for each shadow model i, first the prediction vector $\mathbf{y}$ for each training example $(x, y) \in D^{train}_{shadow^i}$ is computed and then the corresponding record $(y, \mathbf{y}, in)$ is collected to form a dataset D_{attack} for training the attack models. Similarly, for each test example $(x, y) \in D^{test}_{shadow^i}$, the record $(y, \mathbf{y}, out)$ is added to D_{attack}. After that, the dataset D_{attack} is randomly shuffled and then split into two equally sized sets D^{train}_{attack} and D^{test}_{attack} for training and testing the attack models, respectively. Since the attack model is a collection of c_{target} models one for each output class, both of the datasets D^{train}_{attack} and D^{test}_{attack} are split into c_{target} partitions, each associated with a different class label $y \in c_{target}$. Finally, for each class label y, a separate attack model is trained that can predict the "in" and "out" membership for x, given the target model's prediction vector $\mathbf{y}$ on x.

3.2 Differentially Private Deep Learning

Differential privacy, as developed in [23, 24, 25], offers a stronger notion of privacy guarantee for computations involving aggregate datasets. It is based on the principle that seeing the output of a computation, one should not be able to infer the presence or absence of a particular record in the input dataset. To be specific, given two adjacent datasets D and $D^{'}$ that differ only in one record, the outcome of a differentially private mechanism on D and $D^{'}$ should be ideally indistinguishable. The formal definition of differential privacy is given below.

Definition 1: A randomized computation A is said to satisfy (ϵ, δ)-differential privacy if, for any two adjacent datasets D and $D^{'}$ and for any subset O of possible outcomes belonging to $Range(A)$, we have:

$$Pr[A(D) \in O] = e^{\epsilon} Pr[A(D^{'}) \in O] + \delta. \tag{1}$$

The parameter ϵ acts as a nob to control the trade-off between the accuracy of a differentially private mechanism and the amount of information it leaks. This is often referred to as the privacy budget of the differentially private mechanism. The lower the privacy budget ϵ, the lower the information leakage and the higher the privacy guarantee. The additive term δ, as introduced by Dwork et. al. [26], allows for the possibility that the plain ϵ-differential privacy can be broken with probability δ, the value of which is typically chosen to be smaller than $1/|D|$.

A common approach to achieve differential privacy for a function f is to add noise to the output of f calibrated to f's global sensitivity $S_f = max(f(D), f(D^{'}))$, where D and $D^{'}$ are two adjacent datasets differing only in one record. Deep learning, on the other hand, is based on deep neural networks (DNNs) composed of multiple layers of computations that perform many different nonlinear transformations of the inputs. DNNs basically try to learn a composite function f from inputs to outputs as a composition of different operations carried out in the different layers. Therefore, one approach to achieve differential privacy with deep learning is to bound the sensitivity of the component functions at each layer and then add noise to these bounded-sensitivity functions. Finally, the privacy guarantee for the whole composite function f is analyzed. One such approach was proposed in [9] which is able to offer a state-of-the-art DPDM with moderate utility.

To study the impact of membership inference attack on DPDM, we choose to use the above mentioned approach to train a target DPDM model, reasons for which are given in Section 2.1. This particular DPDM is based on two major components – a differentially private version of SGD algorithm called DPSGD, and a tighter bound on privacy budget spending called moments accountant. We give details about these components below.

3.2.1 Differentially Private Stochastic Gradient Descent (DPSGD)

To ensure differential privacy during model training, [9] introduces a differentially private version of the Stochastic Gradient Descent (SGD) algorithm known as Differentially Private Stochastic Gradient Descent (DPSGD) which incorporates Gaussian additive-noise with the gradient updates in SGD. In conventional SGD, first the gradients of the objective function L with respect to the current parameters θ_t are computed. Then the current parameters are updated to produce a new set of parameters θ_{t+1} by subtracting the gradients multiplied by a factor called learning rate (denoted by η) from the current parameter values. This process is repeated until an optimum value for the objective function L is reached. But

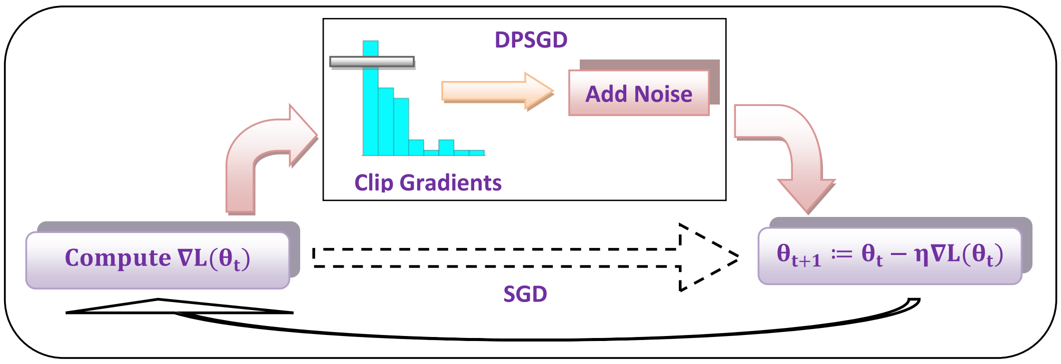

DPSGD
Add Noise
Clip Gradients
Compute $\nabla L(\theta_t)$
SGD
$\theta_{t+1} := \theta_t - \eta\nabla L(\theta_t)$

Figure 2: Differentially Private Stochastic Gradient Descent (DPSGD) algorithm. The conventional Stochastic Gradient Descent (SGD) algorithm, shown at the lower part of the figure, updates the current model parameters denoted by θ_t by directly applying the gradients w.r.t. the model parameters as denoted by $\nabla L(\theta_t)$. On the other hand, DPSGD first clips the l_2-norm of the gradients and then adds noise to the clipped gradients to ensure differential privacy. These differentially private gradients are used for parameter updates just like conventional SGD. Here, η denotes learning rate of the SGD algorithm.

in DPSGD, before updating the parameters, the gradients are clipped to have a maximum predefined l_2-norm bound so that no data point in the training data dominates the learned parameters. This is done in an attempt to avoid overfitting of the model parameters to the training examples. After that, in order to ensure differential privacy, Gaussian noise is added to the clipped gradients before they can be finally used for parameter updates just like conventional SGD. Figure 2 gives an overview of the DPSGD algorithm. For more details, please refer to Algorithm 1 in [9].

3.2.2 Moments Accountant

As with normal SGD, DPSGD usually needs to iterate over the training data and apply the gradient updates multiple times. However, each access to the training dataset causes some leakage of information about the training data and therefore, incurs some privacy loss from the overall privacy budget ϵ. Therefore, an important issue for DPGSD is to be able to keep track of the privacy loss incurred over the whole training process so that the overall privacy budget ϵ is not exceeded. The task of keeping track of the accumulated privacy loss was introduced by McSherry [27] and is known as privacy accounting.

DPSGD needs to access the training dataset multiple times via differentially private mechanisms where each access has its own privacy guarantee. This follows from the basic composition theorem for differential privacy [28, 29] as well as their advanced refinements [30, 31]. The composition theorem basically addresses the problem of how the privacy of a differentially private mechanism degrades under composition of interactive queries, where each query individually provides certain privacy guarantee. These techniques can provide an overall privacy guarantee on the union of all the interactive queries.

However, the existing composition theorems for DP including the strong composition theorem [32] can only provide a loose bound on privacy spending [9]. As a result, they exhaust a moderate privacy budget very quickly allowing only a few iterations over the training dataset, thereby resulting in deep models offering poor model utility. To address this problem, the authors of [9] proposed an state-of-the-art privacy accounting method called moments accountant. Moments accountant can provide a much tighter privacy bound for appropriately chosen settings of the noise scale and the clipping threshold for the gradients in DPSGD. This enables moments accountant to save a factor of $\sqrt{\log(1/\delta)}$ in the ε part and Tq in the δ part over the previously known best bound as provided by the strong composition theorem. Here, T is the total number of iterations over the training dataset and q is the fraction of training examples selected randomly from the training dataset to perform one batch of gradient updates in DPSGD. Because of this tighter bound on privacy spending, DPSGD can iterate over the training dataset sufficient number of times before exhausting a moderate privacy budget. This is why DPSGD is able to train deep models that offer good model utility. Please refer to the original paper [9] for details about the moments accountant technique.

4 Experiments

For evaluating our experiments, we use two well-known image classification tasks - the CIFAR-10 [33] image classification and the MNIST [34] digit recognition. We chose the CIFAR-10 and MNIST classification tasks as they have long been considered as benchmarks for image classification in machine learning. Though we focus on images, in practice, the membership inference attack is not biased towards any particular dataset or model type as demonstrated in [10].

4.1 Datasets

The details of the two datasets are given in the following subsections.

4.1.1 CIFAR-10

CIFAR-10 is a standard classification dataset consisting of 32×32 color images belonging to 10 different object classes, such as, “Aeroplane”, “Automobile”, “Bird”, “Cat”, “Deer”, “Dog”, “Frog”, “Horse”, “Ship” and “Truck”. Given an input image, the goal of the CIFAR-10 classification task is to find out which of the 10 classes the input belongs to. It contains a total of 50,000 training images and 10,000 test images. For the purpose of this study, we first mix-up the training and test images to form a single dataset of 60,000 images which is then randomly split into two datasets each containing 30,000 images. One of these two datasets is used for training and testing the target model with 15,000 images for training and the rest 15,000 for testing. The other dataset is used for training and testing the shadow models. For this dataset, we trained a total of 34 shadow models where each model is trained on a randomly selected 15,000 images from the shadow dataset and tested on the remaining 15,000 images.

4.1.2 MNIST

MNIST is a handwritten digit recognition dataset that contains 28×28 gray-level images of handwritten digits (0 to 9). Given an input image of a digit, the goal of the MNIST classification task is to recognize the digit present in the input image. The MNIST dataset contains a total of 60,000 training images and 10,000 test images. Like CIFAR-10, we also

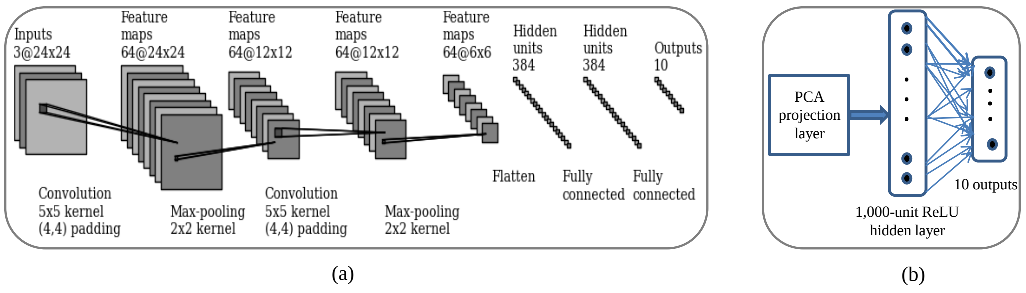


Figure 3: (a) Architecture of the Convolutional Neural Network (CNN) [1] used to train the target model on the CIFAR-10 dataset. (b) Architecture of the deep network used to train the target model on the MNIST dataset.

mix-up the training and test images for MNIST and then follow a similar procedure as CIFAR-10 to generate the train and test datasets for target model as well as the shadow models. For MNIST, we trained a total of 50 shadow models.

4.2 Training Target Models

To train the target DPDM models, we follow the approach of [9] and use two different deep networks for the two datasets, respectively. We describe the network architecture of the target models below.

4.2.1 Target Model for CIFAR-10

For the CIFAR-10 dataset, following the original paper [9], we used a simple Convolutional Neural Network (CNN) architecture having two convolution layers followed by two fully connected layers as shown in Fig. 3(a). Both of the convolution layers contain 64 filters having a kernel size of 5×5 and stride of 1 with input padding of (4, 4). The output from each of the convolution layers is fed into a 2×2 max pooling layer which is then passed through rectified linear unit (ReLU) nonlinearity. The output from the second ReLU unit is then flattened into a vector and passed through two consecutive fully connected layers each having 384 units. Finally, the outputs of the second fully connected layer is fed into a SoftMax classification layer having 10 units for the 10 different classes in CIFAR-10, respectively.

Before feeding the images to the CNN, following the standard practice, we distort each image by first randomly cropping a 24×24 patch from the image and then randomly flipping the image to the left or right. These distorted images are then normalized to have 0-mean and 1-standard deviation by subtracting off the mean and dividing by the variance of the pixels, respectively. Moreover, just like [9], we initialize the weights of the convolution layers from a similar CNN pre-trained on CIFAR-100 dataset [33]. This is because, the approach proposed in [9] requires to compute the gradients of each example for each layer in the deep network, and doing so for the convolution layers has a greater computational overhead [9]. Please note that, there exists no overlap in the images and the classes between CIFAR-10 and CIFAR-100 datasets.

[1]Figure generated by adapting the code from `https://github.com/gwding/draw_convnet`

4.2.2 Target Model for MNIST

For the MNIST dataset, we used a deep network as shown in Fig. 3(b). Following the original work [9], we use a 60-dimensional Principle Component Analysis (PCA) layer at the beginning of the network just to reduce the number of input features from 28x28 to 60, while still capturing the important ones. This is reported to reduce the training time by a factor of about 10 and also increases model accuracy [9]. To enforce differential privacy, Gaussian noise is added to these PCA features. The differentially private PCA features are then fed into a 1,000-unit hidden layer with ReLU nonlinearity, the output of which is then passed to a 10-output SoftMax classification layer for the 10 different digit classes in the MNIST dataset, respectively.

4.3 Training Shadow Models

Since membership inference attack requires the shadow models to be trained in a similar fashion as the target model, we follow the same training procedures for the shadow models as described above. This requirement should not be a limiting factor for launching the membership inference attack against the target DPDMs, as a DPDM is meant to be public, thereby providing white-box access to the model architecture and parameters.

4.4 Training Attack Models

As mentioned earlier, we train n class-specific binary attack models, where n is the total number of output classes in the target model (e.g., 10 for CIFAR-10 as well as MNIST). The reason behind training class-specific attack models is because the target model produces different distributions over its output classes depending on the true class of the input [10]. These binary attack models output probabilities over two output classes namely, "in" and "out", denoting the membership and non-membership of an input record into target model's training dataset, respectively. As illustrated in Fig. 1, the training data for the attack models are generated by collecting the output prediction vectors of the shadow models on their training and test datasets into a single database and then splitting it into n (here, 10) different partitions according to their class labels. For the binary classifiers, we use a simple fully connected neural network with one hidden layer having 64 units with ReLU nonlinearity and a 2-output SoftMax layer to generate predictions for the "in" and "out" membership.

4.5 Varying the Privacy Parameter ε

In order to study the impact of membership inference attack against different private models, we trained 4 different ε-DPDMs providing an ε guarantee of 1, 2, 4, and 8, respectively with a fixed δ value of 10^{-5}. Moreover, for each dataset, we also trained a totally non-private deep model having the same network architecture as the corresponding DPDM in an attempt to gain further insight. Since the output sensitivity of the different layers in a deep network do not change over iterations over the training data, we trained the different ε-DPDMs by allowing training to continue until the target privacy budget ε (i.e., 1, 2, 4, and 8) is exhausted. Table 2 shows the total number of iterations over the training data required to train the corresponding ε-DPDMs as well as the non-private model for both datasets.

Datasets	ε=1	ε=2	ε=4	ε=8	non-private
CIFAR-10	4	15	55	72	3000
MNIST	2	15	63	232	627

Table 2: Number of iterations over the training data required to train the different ε-DPDMs (achieved by varying the privacy parameter ε) as well as a non-private model for CIFAR-10 and MNIST datasets.

4.6 Experimental Setup

Following the best reported values in [9], we use a noise scale σ of 6, and a gradient clipping threshold of 3 for the CIFAR-10 dataset, whereas, for the MNIST dataset, both of these parameters are set to 4. Moreover, the PCA projection layer for the MNIST dataset uses a noise scale σ of 7. The learning rate for the CIFAR-10 dataset is fixed at 0.01, while that for the MNIST dataset is initialized at 0.1 and then linearly decreased to 0.05 over the first 10 epochs after which it is fixed at 0.05. The batch size used for the CIFAR-10 and MNIST datasets are 2,000 and 600, respectively. For training the attack models, the batch size used is 10,000 and the learning rate is initially set at 0.01 and then linearly downscaled to 0.001 over the first 10 epochs after which it is fixed at that value. For all the experiments, we used the TensorFlow [35] deep learning framework.

5 Results

5.1 Performance Metrics

The goal of the membership inference attack is to determine whether a given input record belongs to the training dataset of the target model or not. Therefore, to evaluate the performance of the attack, we use two metrics – accuracy and F_1-score of the attack. The accuracy measure simply reports the percentage of examples that are correctly predicted to be members of the target model's training dataset. Since the target model's training and test datasets are of the same size (15,000 for CIFAR-10 and 17,500 for MNIST), the attack model's test dataset have equal number of examples for members and non-members of the target model. Therefore, attack accuracy for random guess would be 0.5 which we use as a baseline for this work.

The F_1-score, as defined in Eq. 2, is a metric that combines the precision and recall measures into a single value. In our case, precision measures the fraction of the images predicted to be members of the target model's training dataset that are truly members, whereas, recall is the fraction of the images belonging to the target model's training dataset that are correctly predicted to be members. As a strong baseline, we consider a naive classifier that predicts every input record to be the member of the target model's training dataset. Since we have equal number of members and non-members in the attack model's test dataset, the recall for such a naive classifier would be 1.0 and precision would be 0.5, thereby resulting in a baseline F_1-score of 0.67. Since the attack models are trained class-wise, we report all the measurements per class.

$$F_1\text{-score} = \frac{2 * (Precision * Recall)}{Precision + Recall} \tag{2}$$

Datasets	ε=1	ε=2	ε=4	ε=8	non-private
CIFAR-10 (train)	0.247	0.450	0.608	0.686	0.944
CIFAR-10 (test)	0.253	0.450	0.607	0.681	0.737
MNIST (train)	0.762	0.874	0.909	0.937	0.999
MNIST (test)	0.757	0.870	0.906	0.932	0.970

Table 3: Train and test accuracies on CIFAR-10 and MNIST datasets for different ε-DPDMs (achieved by varying the privacy parameter ε) as well as a non-private model.

5.2 Accuracy of Membership Inference Attack

5.2.1 Accuracy on CIFAR-10

Figure 4(a) shows per class accuracy of the membership inference attack against different private models (achieved by varying the privacy parameter ε) as well as a non-private model on the CIFAR-10 dataset. As expected, the non-private model leaks considerable amount of information about its training dataset resulting in an attack accuracy as high as 0.80 (for the "Aeroplane" class) and as low as 0.60 (for the "Truck" class), both above the baseline of 0.5. As reported in the original paper for membership inference attack [10], this is mainly attributed to the fact that the non-private model is heavily over-fitted to its training dataset which is obvious from Table 3 reporting the train and test accuracies of the different models. The higher the gap between train and test accuracies, the more overfitted the model is to its training dataset, thereby causing more information leakage [10].

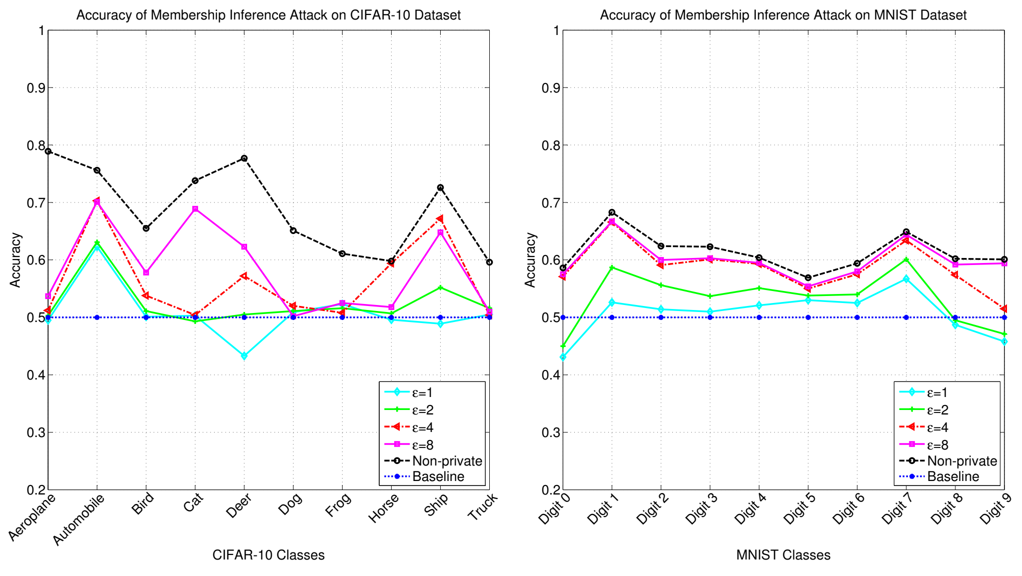
The plots for ε=1 and ε=2 show that the attack accuracy against the corresponding target DPDM models is not significant and almost similar to the baseline for most of the classes. This result, therefore, serves as an evidence that the DPDM's promise to offer protection against a strong adversary having full knowledge of the training mechanism and the model parameters holds for these particular values of ε for this particular dataset.

While the accuracy of the attack for ε=4 jumps on and off around the baseline, the plot for the DPDM model with ε=8 shows that considerable amount of information about the training dataset is leaked through the model for quite a good number of classes. This particular result demonstrates that for deep networks, such as the CIFAR-10 CNN, an ε value of 8 may not be sufficient to protect the privacy of the training data against sophisticated attack models such as the membership inference attack.

On the other hand, Table 3 suggests that a privacy budget of ε=8 is a minimum in order for a DPDM to achieve comparable accuracy on the CIFAR-10 dataset, as the other ε values reported in the table result in poor model accuracies. This is also in-line with what is reported in the literature [9]. This observation combined with the attack accuracy on the CIFAR-10 dataset, therefore, indicates that a DPDM for CIFAR-10 may withstand strong adversaries by only sacrificing model utility (e.g., more privacy for reduced accuracy at ε=4 and below). But, it exhibits moderate vulnerability to the membership inference attack when offering an acceptable utility level (e.g., competitive model accuracy for ε=8 and possibly, above).

5.2.2 Accuracy on MNIST

Figure 4 (b) shows per class accuracy of the membership inference attack against different private models (achieved by varying the the privacy parameter ε) as well as a non-private model on the MNIST dataset. Compared to the CIFAR-10 dataset, the membership infer-



(a) (b)

Figure 4: Per class accuracy of the membership inference attack against different ε-DPDMs (achieved by varying the privacy parameter ε) as well as a non-private model on (a) CIFAR-10 and (b) MNIST datasets.

ence attack on the MNIST dataset is more successful for the different ε values with the exception of the non-private model. This is obvious from Table 4 reporting the attack accuracy and F_1-score averaged over all classes on CIFAR-10 and MNIST datasets for different ϵ-DPDMs. This may be attributed to the fact that the train-test accuracy gap on the MNIST dataset for each ε-DPDM is comparatively higher than that on the CIFAR-10 dataset as evidenced from Table 3, with the scenario reversed for the non-private model. This higher train-test accuracy gap is one of the main reasons for the success behind the membership inference attack as mentioned in the original paper on membership inference attack [10].

Coming back to Fig. 4, of particular interest are the cases for ε=4 and ε=8, for these models exhibit an attack accuracy as close to that of the totally non-private model, thus resulting in substantial leakage of information about the training data. But, referring back to Table 3, we see that a minimum privacy budget of ε=4 is required for a DPDM to offer a comparable

	ε=1 (averaged)	ε=2 (averaged)	ε=4 (averaged)	ε=8 (averaged)	Overall Average
CIFAR-10 (Accuracy)	**0.508**	0.524	0.563	0.583	0.544
MNIST (Accuracy)	0.507	**0.533**	**0.587**	**0.600**	**0.557**
CIFAR-10 (F_1-score)	**0.622**	0.621	0.690	0.721	0.664
MNIST (F_1-score)	0.581	**0.674**	**0.797**	**0.832**	**0.721**

Table 4: Attack accuracy and F_1-score on CIFAR-10 and MNIST datasets averaged over all classes for different ε-DPDMs (achieved by varying the privacy parameter ε).

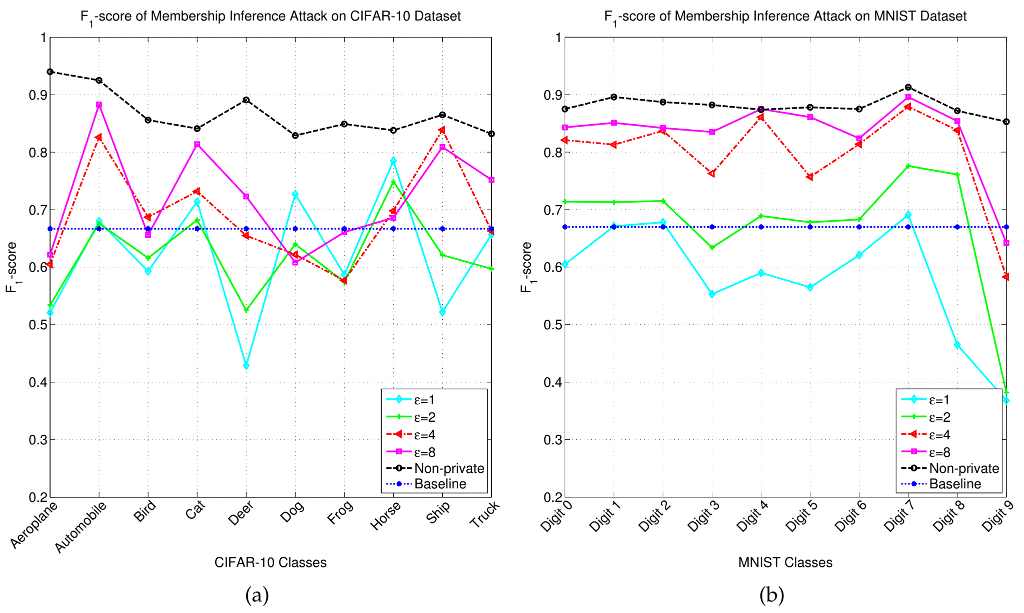


Figure 5: Per class F_1-score of the membership inference attack against different ε-DPDMs (achieved by varying the privacy parameter ε) as well as a non-private model on (a) CIFAR-10 and (b) MNIST datasets.

accuracy on the MNIST dataset. Therefore, just in the case of CIFAR-10 dataset, it follows that a DPDM for MNIST, when offering an acceptable utility level (e.g., ε=4 or above), becomes vulnerable to membership inference attack, but may provide protection against such attacks by sacrificing the model utility by a moderate margin (e.g., ε=2 or below).

5.3 F_1-score of Membership Inference Attack

5.3.1 F_1-score on CIFAR-10

Figure 5 (a) shows per class F_1-score of the membership inference attack against different private models (achieved by varying the privacy parameter ε) as well as a non-private model on the CIFAR-10 dataset. The plots for F_1-score follow a similar trend as accuracy with the F_1-score for ε=1 and ε=2 being well below the baseline for most of the classes. The DPDM model with ε=4 is just below the baseline with a few exceptions (e.g., "Automobile" and "Ship" classes). On the other hand, the F_1-score for a privacy budget of ε=8 clearly reveals concerning level of vulnerabilities to membership inference attack for majority of the classes, with the vulnerability, in some cases, reaching as close as that of the totally non-private model (e.g., "Automobile", "Cat" and "Ship" classes). This is in line with the results we achieved for attack accuracy on this dataset, and therefore, reiterates the observation that a DPDM with ε=8 on CIFAR-10 cannot survive the membership inference attack, but may provide privacy protection against such attacks for ε values that cause poor model utility.

5.3.2 F_1-score on MNIST

The F_1-score of the attack on the MNIST dataset for different private models as achieved by varying the value of ε as well as a non-private model are depicted class-wise in Fig. 5(b). As before, the DPDMs with a privacy budget of ε=1 and ε=2, respectively are able to keep their promise to withstand privacy attacks as the F_1-score of the attack for these models are mostly below the baseline. But, the DPDMs with an ε value of 4 and 8 exhibit vulnerability to membership inference attack making them almost as bad as the totally non-private model. Moreover, as is the case with attack accuracy, the membership inference attack achieves a higher F_1-score on MNIST than CIFAR-10 as demonstrated in Table 4.

5.4 Discussion

5.4.1 Utility vs. Privacy

As evidenced from experimental results above, DPDMs with ε values of 2 and below may keep their promise to withstand strong adversarial attacks like the membership inference attack. But, such ε values result in deep models that are unable to offer a competitive level of utility as reported in Table 3. On the other hand, though an ε value of 8 and above for CIFAR-10, or 4 and above for MNIST could result in models offering moderate utility, such models may not withstand the membership inference attack as obvious from the experimental results. It is, therefore, a choice of the application designer to appropriately trade-off between model utility and expected privacy and is largely dependent on the application at hand.

5.4.2 Over-fitting and Attack Accuracy

One of the main factors behind the success of the membership inference attack is over-fitting [10]. Over-fitting is a scenario when a trained model tries to fit too closely to the training data, possibly fitting to every single example including the random noise usually present in the data. A commonly used metric to measure how much a model is over-fitted to the training data is the gap between the accuracy on the training data and test data. The larger the gap, the more over-fitted a model is and the more information it leaks about its training data.

Referring back to Table 3, we see that the largest train-test accuracy gap is exhibited by the non-private model, followed by the DPDMs with $\epsilon = 8$ and $\epsilon = 4$. Therefore, the non-private model is more over-fitted to the training data than the ϵ-DPDMs, with $\epsilon = 8$ showing higher over-fitting than $\epsilon = 4$. Looking at the accuracy of the membership attack as demonstrated in Table 4, we see that the non-private model is more vulnerable to the attack than the ϵ-DPDMs, with $\epsilon = 8$ showing greater vulnerability than $\epsilon = 4$. These results directly follow our discussion at the beginning of the subsection.

6 Conclusion

In this paper, we systematically study the impact of a recent and sophisticated machine learning-based privacy attack called membership inference attack against a state-of-the art differentially private deep model. Our experimental results indicate that differentially private deep models may provide privacy protection against strong adversaries only by sacrificing model utility by a considerable margin. As a result, such models may be vulnerable to modern and sophisticated privacy attacks such as the membership inference attack, when providing a competitive utility level. We, therefore, advocate for an empirical value of ε that best trade-offs the utility and privacy for the current application at hand.

As a future extension to this work, it would be interesting to see how the attack's performance varies as the shadow data are generated synthetically or sampled based on a noisy-real distribution, as demonstrated for non-private models in the original paper on membership inference attack [10]. Moreover, varying the size of the training dataset would be another interesting option to explore. Apart from image datasets, we also would like to experiment with some non-image datasets. Another interesting avenue to extend the work would be to study the impact of different neural network architectures on membership inference attack.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[2] W. AbdAlmageed, Y. Wu, S. Rawls, S. Harel, T. Hassner, I. Masi, J. Choi, J. Lekust, J. Kim, P. Natarajan *et al.*. Face recognition using deep multi-pose representations. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, IEEE, pp. 1–9, 2016.

[3] I. W. P. Consortium *et al.*. Estimation of the warfarin dose with clinical and pharmacogenetic data. *N Engl J Med*, vol. 2009, no. 360, pp. 753–764, 2009.

[4] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1310–1321, 2015.

[5] A. D. Sarwate and K. Chaudhuri. Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data. *IEEE signal processing magazine*, vol. 30, no. 5, pp. 86–94, 2013.

[6] S. Gade and N. H. Vaidya. Private learning on networks. *arXiv preprint arXiv:1612.05236*, 2016.

[7] 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*, pp. 1322–1333, 2015.

[8] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, Springer, pp. 265–284, 2006.

[9] M. Abadi, A. Chu, I. Goodfellow, B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *23rd ACM Conference on Computer and Communications Security (ACM CCS)*, pp. 308–318, 2016.

[10] R. Shokri, M. Stronati, and V. Shmatikov. Membership inference attacks against machine learning models. To appear in *Security and Privacy (SP), 2017 IEEE Symposium on*, IEEE, 2017.

[11] F. McSherry and I. Mironov. Differentially private recommender systems: Building privacy into the net. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09, pp. 627–636, 2009.

[12] K. Chaudhuri and C. Monteleoni. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems 21*, pp. 289–296, 2009.

[13] R. Bassily, A. D. Smith, and A. Thakurta. Private empirical risk minimization, revisited. *CoRR*, vol. abs/1405.7085, 2014.

[14] N. Phan, Y. Wang, X. Wu, and D. Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In *AAAI*, pp. 1309–1316, 2016.

[15] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *Proceedings of the 5th International Conference on Learning Representations, Toulon, France*, 2016.

[16] L. Breiman. Bagging predictors. *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[17] 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 Genet*, vol. 4, no. 8, p. e1000167, 2008.

[18] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov. "you might also like:" privacy risks of collaborative filtering. In *Security and Privacy (SP), 2011 IEEE Symposium on*, IEEE, pp. 231–246, 2011.

[19] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, vol. 10, no. 3, pp. 137–150, 2015.

[20] 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 *USENIX Security*, pp. 17–32, 2014.

[21] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart. Stealing machine learning models via prediction apis. In *USENIX Security*, 2016.

[22] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1528–1540, 2016.

[23] C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[24] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pp. 265–284, 2006.

[25] C. Dwork, and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. (3–4), pp. 211–407, 2014.

[26] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pp. 486–503, 2006.

[27] F. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *SIGMOD*, pp. 19–30, ACM, 2009.

[28] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pp. 486–503. 2006.

[29] C. Dwork, and J. Lei. Differential privacy and robust statistics. In *STOC*, pp. 371–380. ACM, 2009.

[30] C. Dwork, and G. Rothblum. Concentrated differential privacy. CoRR, abs/1603.01887, 2016.

[31] P. Kairouz, S. Oh, and P. Viswanath. The composition theorem for differential privacy. In *ICML*, pp. 1376–1385. ACM, 2015.

[32] C. Dwork, G. N. Rothblum, and S. P. Vadhan. Boosting and differential privacy. In *FOCS*, pp. 51–60, 2010.

[33] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[35] M. Abadi *et al.*. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.