KEY PROTECTED CLASSIFICATION FOR GAN ATTACK RESILIENT COLLABORATIVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale publicly available datasets accelerate deep learning studies. However they are not always available for all domains, especially for the ones in which sensitive information of subjects must be kept private. Collaborative learning techniques provide a privacy-preserving solution for the data owners who do not want to directly share their datasets with each other due to privacy concerns. Existing collaborative learning techniques (with the integration of the differential privacy concept) are shown to be resilient against a passive adversary which tries to infer the training data only from the resulting model parameters. However, recently, it has been shown that the existing collaborative learning techniques are vulnerable to an active adversary that runs a GAN attack during the learning phase. In this work, we propose a novel key-based collaborative leaning technique that is resilient against the GAN attack. We propose a collaborative learning technique in which class scores of each participant are protected by class-specific keys. We also introduce fixed neural network components into the proposed model in order to use high dimensional keys (for higher robustness) without increasing the model complexity. Via experiments using two popular datasets MNIST and AT&T Olivetti Faces, we show the robustness of the proposed technique against the GAN attack. To the best of our knowledge, the proposed technique is the first collaborative leaning technique that is resilient against an active adversary.

1 INTRODUCTION

Deep neural networks have shown remarkable performance in numerous domains, including computer vision, speech recognition, language processing, and many more. Most deep learning approaches rely on training over large-scale datasets and computational resources that makes the utilization of such datasets possible.

While large-scale public datasets, such as ImageNet (Deng et al., 2009), Celeb-1M (Guo et al., 2016), and YouTube-8M (Abu-El-Haija et al., 2016), have a fundamental role in deep learning research, it is typically difficult to collect a large-scale dataset for problems that involve processing of sensitive information. Data privacy becomes a significant concern if one considers training over all the voice, iris, finger-print, or face samples collected by the users of a mobile application.

To enable training over large-scale datasets without compromising data privacy, a collaborative learning framework (CLF) is proposed by Shokri & Shmatikov (2015). In CLF, the model is trained in a distributed way, where each *participant* contributes to training without sharing its (sensitive) data with other participants. More specifically, each participant hosts only its own training examples, and a central server, called the *parameter server*, combines local model updates into a shared model. Therefore, the training procedure effectively utilizes the data owned by all participants. At the end, the final model parameters are shared with all participants.

However, there are cases where the original CLF approach fails to preserve data privacy due to knowledge embedded in the final model parameters. In particular, Fredrikson et al. (2015) show that the parameters of a neural network model trained on a dataset can be exploited to partially reconstruct the training examples in that dataset. To mitigate this threat, one may consider partially corrupting the model parameters by adding noise into the final model as formulated by Chaudhuri et al. (2011). However, such prevention mechanisms introduce a difficult trade-off between classifier accuracy versus data privacy for collaborative training.

The study by Shokri & Shmatikov (2015) also shows that differential privacy (Dwork, 2011) can be incorporated into CLF in a way that guarantees the indistinguishability of the participant data. Following this study, several other differential privacy based approaches, which provide noise injection methods (Phan et al., 2016; Abadi et al., 2016) or training frameworks (Papernot et al., 2016), have recently been proposed.

However, it has recently been shown that CLF can be vulnerable to not only passive attacks, but also much more powerful *active attacks*, *i.e.*, training-time attacks, for which the CLF with differential privacy fails to prevent the attack and there is no known prevention technique in general (Hitaj et al., 2017). More specifically, a training participant can construct a generative adversarial network (GAN) (Goodfellow et al., 2014) such that its GAN model learns to reconstruct training examples of one of the other participants over the training iterations. For this purpose, attacker defines a new class for the joint model, which acts as the GAN discriminator, and utilizes the samples generated by its GAN generator when locally updating the model. In this manner, the attacker effectively forces the victim to release more information about its samples, as the victim tries to differentiate its own data from attacker class during its local model updates. Hitaj et al. (2017) also show that reasonable noise levels for differential privacy (that allows local models to train efficiently and accurately) does not prevent this GAN attack.

In this paper, we propose a novel collaborative learning formulation that prevents the GAN attack. First, we observe that GAN attack depends on the classification scores of the targeted classes. Based on this observation, we define a classification model where class scores are protected by class-specific keys, which we call *class keys*. To make the CLF approach secure even in the case of a compromised parameter server, our approach generates class keys independently within each training participant and keeps the keys private throughout the training procedure. In this manner, we prevent the access of the adversary to the target classes, and therefore the GAN attack. We also demonstrate that the dimensionality of the keys directly affect the security of the proposed model, much like the length of the passwords. We observe, however, that naively increasing the key dimensionality can greatly increase the number of model parameters, and therefore, reduce data efficiency and classification accuracy. We demonstrate that this issue can be addressed by introducing fixed neural network components that allow using much higher dimensional keys without increasing the model complexity. We experimentally validate that our approach prevents a GAN attack while providing effective collaborative learning on the MNIST (LeCun et al.) and Olivetti Faces (Samaria & Harter, 1994) datasets, both of which are challenging datasets in the context of privacy attacks due to their relative simplicity, and therefore, the ease of reconstructing the samples in them.

The rest of the paper is organized as follows. In Section 2, we provide an overview of the recent research in privacy-preserving machine learning. In Section 3, we present a detailed and technical summary of collaborative learning, GANs, and the GAN attack technique. In Section 4, we discuss the details of our approach. In Section 5, we provide an experimental validation of our model. Finally, in Section 6, we conclude the paper.

2 Related work

Privacy preserving machine learning methods have become popular over the recent years. In this section we describe the most relevant ones to our work.

Differential privacy (DP) defined in Dwork (2011) provides sample indistinguishability, therefore protects privacy of subject data. This concept is first formalized for machine learning in general in Chaudhuri et al. (2011). Recent studies utilize DP in large-scale learning as (i) structured noise addition as in (Abadi et al., 2016; Phan et al., 2016), or (ii) 2-step training methodology as in Papernot et al. (2016).

Collaborative learning framework (CLF, Shokri & Shmatikov (2015)) enables large-scale learning in scenarios where there is no public dataset available. Besides, authors show that DP can be integrated into CLF in such a way that pooled gradients do not leak much information about participants. This leakage can be controlled by the noise parameter ϵ resulting in a trade-off between utility and accuracy.

Very recently, Hitaj et al. (2017) devised a powerful attack against CLF. In this attack, one of the participants in the CLF is assumed to be the adversary. The adversary tries to exploit one of classes

belonging to other participants (victim) by using a generative adversarial network (GAN) (Goodfellow et al. (2014)). This attack is powerful in a sense that adversary can actively influence the victim to release more details about its samples during the training process. Therefore, it is denoted as an active attack, different from the passive one investigated in Fredrikson et al. (2015). In Hitaj et al. (2017), it is also shown that GAN attack works even when participants adopt DP during gradient sharing. To the best of our knowledge no solution has been proposed against the GAN attack. In this work, we propose a novel collaborative learning model that is resilient against the GAN attack.

3 BACKGROUND

Our work builds on the collaborative learning framework (Shokri & Shmatikov, 2015) and tackles the generative adversarial network attack problem (Hitaj et al., 2017). Therefore, before providing the details of our approach, we first provide brief background on CLF, GAN, and the GAN attack in the following.

3.1 COLLABORATIVE LEARNING OF PARTICIPANTS

The goal of CLF is to collaboratively train a shared model over the private datasets of several participants such that the model generalizes across all the participant datasets. For this purpose, CLF defines a protocol where each participant shares information only about its learning progress, rather than the data directly, with the others over the training iterations. Locally, participants train their model as usual using gradient based optimization, but share with others fractions of changes in model parameters, at predefined intervals. The framework is set up among participants based on the following components and the associated policies:

(i) A mechanism for participants to share parameter updates. This is typically realized by a trusted third-part parameter server (PS). As its name suggests, it is a platform where participants accumulate their model updates by means of uploading or downloading a predefined fraction of parameter changes during training.
(ii) A common objective and model architecture. All participants use the same model architecture and training objective. Typically, participants declare class labels for which they have training data.
(iii) Meta-parameters. The hyper-parameters of the CLF setup, such as the parameter download fraction (θ_d) and the upload fraction (θ_u), gradient clipping threshold (γ), and the order of participants during training (e.g., round robin, random, or asynchronous) are typically pre-determined. See Shokri & Shmatikov (2015) for a full list of meta-parameters.

Once the framework is established, participants start training on their local datasets in a predetermined order. When a participant takes turn, it first downloads θ_d fraction of parameters from the PS and replaces them with its local parameters. After performing one epoch of training on its local dataset, participant uploads θ_u fraction of resulting gradients to the PS. It is also possible to incorporate differential privacy to guarantee sample indistinguishability for enhanced privacy protection.

3.2 GENERATIVE ADVERSARIAL NETWORK

Generative adversarial network (Goodfellow et al., 2014) is an unsupervised learning process for learning a model of the underlying distribution of a sample set. A GAN model consists of two submodels, called generator and discriminator. The generator corresponds to a function $G(z;\theta_G)$ that aims to map each data point z sampled from a prior distribution, *e.g.* uniform distribution $\mathcal{U}(-1,1)$, to a point in the data space, where θ_G represents the generator model parameters. Similarly, the discriminator is a function $D(x;\theta_D)$ that estimates the probability that a given x is a real sample from the data distribution, where θ_D represents the discriminator model parameters.

The generator and discriminator models are trained in turns, by plaing a two-player minimax game. At each turn, the generator is updated towards generating samples that are indistinguishable from the real samples according to the current discriminator's estimation:

$$\min_{\theta_G} \; \mathbf{E}_z\big[\log(1 - D(G(z;\theta_G));\theta_D)\big]. \tag{1}$$

The discriminator, on the other hand, is updated towards distinguishing the samples given by G from the real ones:

$$\max_{\theta_D} \; \mathbf{E}_x\big[\log(D(x;\theta_D)\big] + \mathbf{E}_z\big[\log(1 - D(G(z;\theta_G));\theta_D)\big]. \tag{2}$$

GANs have successfully been utilized in numerous problems, *e.g.* see Luc et al. (2016); Nguyen et al. (2016); Zhang et al. (2016); Yeh et al. (2016).

3.3 GAN Attack in Collaborative Learning Framework

Hitaj et al. (2017) devise a powerful GAN-based active attack for the collaborative learning framework. In this scenario, an adversary participant takes places during training in a CLF setup, (incorrectly) declares that it hosts some class c_{fake} and tries to extract information about some class c_{attack}[1]. Hitaj et al. (2017) show that the adversary can execute an attack by secretly training a generator model for the class c_{attack}, and treating the shared model as a discriminator. Such an approach effectively turns CLF into a GAN training setup where the adversary takes the following steps:

(i) Adversary updates its generator network towards producing samples that are classified as class c_{attack}, by the shared classification model, as in Eq. (1).
(ii) Adversary takes samples from its generator, labels them as c_{fake} and updates the shared classification model towards classifying the synthetic samples as class c_{fake}, as in Eq. (2).

The GAN attack works in two ways. First, throughout the training iterations, the adversary continuously updates its generator, therefore, it can progressively improve its generative model and the reconstructions that it provides. Second, since the adversary defines the class c_{fake} as part of the shared model, the participant that hosts c_{attack} updates the shared model towards minimizing the misclassification of its training examples into class c_{fake}. Over the iterations, this practically forces the victim participant into releasing more detailed information about the class c_{attack} while updating the shared model (Hitaj et al., 2017). This latter step makes the GAN attack particularly powerful as it influences the training of all participants, and, it is also the main reason why the technique is considered as an active attack.

4 Proposed method

In this section, we describe the details of our proposed approach to prevent the GAN attack. Our model relies on the distributed leaning framework proposed by Shokri & Shmatikov (2015) and we assume that there is an active adversary as in Hitaj et al. (2017). We explain the details of our key-protected classification model in Section 4.1. Then, we propose an extension of our approach that enables efficient incorporation of high dimensional keys, which reduces the success likelihood of a GAN attack, in Section 4.2.

4.1 Key Protected Classification Model

Our starting point is the observation that GAN attack relies on the knowledge of classification score of the target class throughout the training iteration, as also discussed in Section 3.3. To prevent a GAN attack in a collaborative learning setup, we aim to mathematically prevent each participant from estimating the classification scores for the classes hosted by the other participants. For this purpose, we introduce class-specific keys for all classes and parameterize the classification function in terms of these keys in a way that makes classification scores estimation unlikely without having access to the keys.

For this purpose, we require each participant to generate a random class key for its classes during initialization, and keep it private until the end of the training procedure. We denote the key for class c by $\psi(c) \in \mathbb{R}^{d_k}$, where d_k is the pre-determined dimensionality of each key. For the sake of simplicity, we assume that there is a single adversary, and participants do not have overlapping classes.

[1]The adversary may additionally have its own real classes and a real dataset, but for the sake of simplicity, we assume that the adversary works only on its privacy attack. Similarly, we assume that each class is hosted by a single participant.

Our goal is to train a deep (convolutional) neural network for classification. However, unlike the traditional neural networks that directly output classification scores, we define the network as an embedding function $\phi(x)$ that maps each given input x (*e.g.* image) from the source domain into a d_k-dimensional vector. Then, we define the classification score for class c by a simple dot product between the embedding output and the class key:

$$\langle \psi(c), \phi(x) \rangle, \tag{3}$$

where $\psi(c), \phi(x) \in \mathbb{R}^{d_k}$. Therefore, the final classification takes form of choosing the class whose class key leads to the maximum classification score, *i.e.* $\arg\max_c \langle \psi(c), \phi(x) \rangle$. We assume that all class keys and also the output of the embedding network are ℓ_2-normalized[2], for two important reasons that we explain below.

Since the classification scores depend on the class keys and each participant has access only to the keys of its own classes, it is arguably not appropriate to try to train over a discriminative loss function. Therefore, we instead formulate a regression-like training objective that aims to maximize the classification score of the true class, without comparing cross-class scores:

$$\max_{\theta} \sum_{i=1}^{N} \sum_{c \in C^i} \sum_{x \in X^c} \langle \psi(c), \phi(x) \rangle + \lambda ||\theta||^2, \tag{4}$$

where N is the number of participants, C^i is the set of classes hosted by the participant i, X^c is the set of samples belonging to class c, θ is set of the trainable model parameters of the embedding function $\phi(x)$ and λ is the regularization weight hyper-parameter. In this manner, we aim to learn $\phi(x)$ such that the resulting embeddings are maximally correlated with the correct class keys.

Here, we emphasize the first reason why it is important to use ℓ_2-normalized class keys and embedding outputs: in this manner, the resulting classification score is by definition restricted to the range $[-1, +1]$, which avoids learning a degenerate embedding function that increases the training objective simply by producing excessively large classification scores for few samples.

In our approach, even if an adversary gets involved during training, it cannot target a particular class without knowing its class key. However, there is still a chance that the adversary may target an arbitrary class by using a randomly generated key as the target key, and, aim to reconstruct the samples belonging to one of the classes without necessarily knowing the id of the targeted class. In principle, such an attack can be successful if the randomly generated attack is sufficiently similar to one of the actual class keys.

To prevent such an attack, it is essential to reduce the probability of approximately replicating class keys by randomly generating keys. Here, one might consider determining and distributing keys in a centralized manner, however, such a prevention technique is not reliable as it is typically not possible to enforce participants to use the assigned keys, *i.e.* the adversary may still attempt to attack with a random key that it generates privately. Therefore, we need to minimize the probability of generating keys that will lead to scores highly correlated with one of the class scores, without relying on the restrictions on the keys used by the participants, such that an adversary will (most likely) not be succesful in training a generative model through a GAN attack.

To address this problem, we rely on high-dimensional, ℓ_2 normalized class keys such that it becomes unlikely that to generate highly correlated class key through random sampling. In this manner, we are also able to let the participants generate their own keys by sampling each key dimension from a fixed distribution, like $\mathcal{N}(0, 1)$ or $\mathcal{U}$(-1,1), and ℓ_2 normalizing the resulting vector. We empirically observed that the particular choice of the distribution does not matter as long as we use sufficiently high dimensional vectors, where the resulting vectors end up benig approximately orthogonal to each other. We also emphasize that a pair of ℓ_2 normalized vectors tend to be progressively less correlated as their dimensionality increases (see Section 5.2 for details). By using these nearly orthonormal class keys, even though adversary can still train its local generator, the generator cannot learn data distribution of samples for any class. Hence the GAN attack is prevented as we show in Section 5.4. We provide the pseudo code for the proposed model in Algorithm 1 in Appendix A.

To further minimize the success probability of the adversary, we devise a slight modification to model architecture as discussed in Section 4.2. With this solution each participant can train its local model on its local dataset by using its class keys generated by itself.

[2]The final layer of the neural network is an ℓ_2 normalization layer.

4.2 LEARNING WITH HIGH DIMENSIONAL KEYS

In Section 4.1 we state that non-correlated key generation mechanism is required to reduce the probability of the adversary exploiting any classes in the CLF. However, ℓ_2 normalized vectors can be quite overlapping as we show in Section 5.2. In order to reduce probability of this overlapping, sizes of the vectors (i.e., sizes of the class keys) can be increased. Increase in key size results in more trainable parameters for participants. Therefore, this increase in the size of key vectors also increases the complexity of model architecture which may cause (i) extra work for participants during training, (ii) poor test performances (due to bias-variance trade-off), (iii) that parameters in the PS overfits to one of participants during training..

To overcome these problems, we propose to add at the end of model architecture a *fixed* dense layer with an activation function. By doing so, parameters of this dense layer are predetermined and kept intact during training. Thus, the layer does not impose any extra trainable parameters to be learned. It is just used for mapping embeddings to higher dimensions. We let this layer to be shared among all participants. By utilizing a fixed layer, we show that key size can be increased without effecting convergence of participants so that the GAN attack is prevented in Sections 5.3 and 5.4.

5 EXPERIMENTS

In this section we demonstrate that our proposed solution prevents GAN attacks while enabling effective collaborative learning of participants. We first explain training details in Section 5.1. We show the relationship between sizes of two ℓ_2 normalized random vectors and their dot products in Section 5.2. We verify that key-based learning works for CLF when there is no adversary inside the framework in Section 5.3. Then, we empirically prove that our proposed model prevents GAN attacks when participants create high dimensional keys in Section 5.4.

5.1 TRAINING DETAILS

We perform experiments on well-known MNIST handwritten digits (LeCun et al.) and AT&T Olivetti Faces (Samaria & Harter (1994)) datasets. We choose these datasets for following reasons:

(i) It is known that GANs are difficult to train (Goodfellow (2017)). Several improvements are proposed for GANs such as Salimans et al. (2016), Tolstikhin et al. (2017), Mirza & Osindero (2014), Che et al. (2016), Warde-Farley & Bengio (2017), Arjovsky et al. (2017), and Arjovsky & Bottou (2017). We want the generator to capture statistics of data rather easily and both these datasets allow this behavior.

(ii) Qualifying reconstructions of the adversary trained on these datasets is relatively easy. Thus, we can assess performance of our model factually.

In all our experiments, we create virtual participants which have datasets that include disjoint class labels. This assumption is plausible since we try to prevent the GAN attack. If the adversary has common classes with the victim, there is no point of attacking to one of these common classes. We observed that collaborative learning with 5 or more participants is challenging as parameters in the PS overfits quickly to the local dataset of any one of these participant. As a work around, we set λ carefully such that (i) it is sufficiently large to make sure that weights can be regularized for all local datasets, and (ii) it is sufficiently small for participants to learn in the presence of a fixed key.

We use $\tanh$ as activation function for the fixed layer. Bounded embeddings containing both positive and negative terms provides smooth training when random class keys are sampled from $\mathcal{U}$(-1,1). We also found out that learning with fixed layer is also compelling especially when the fixed layer transforms embeddings to very high dimensional spaces (e.g., to $\mathbb{R}^{16384}$). We have multiple solutions for this problem: (i) using Adam optimizer (Kingma & Ba (2014)) and sharing gradients of its parameters with the PS as well. (ii) using batch normalization (Ioffe & Szegedy (2015)) (or instance normalization (Ulyanov et al. (2016)) which works better for Olivetti Faces dataset when processing one sample at a time) after the fixed layer. Trainable parameters of the normalization layer provides extra scale and offset for learning embeddings which help participants to converge quickly.

For meta-parameters of CLF, in all experiments we set θ_d and θ_u to 1.0 as Hitaj et al. (2017) shows that the GAN attack also works for smaller values of θ_d and θ_u. We also exclude gradient selection

mechanism, γ and τ. We found out that random and asynchronous orders lead to unstable training of participants, especially when there are 5 or more participants.

5.2 KEY SELECTION

In this section we empirically show how sizes of 2 randomly generated vectors (class keys) effect their dot products, and hence the overlap between them. We generate ℓ_2 normalized 100 random vectors whose entries are sampled from either $\mathcal{N}(0,1)$ or $\mathcal{U}$(-1,1). Then, we find maximum of pair-wise dot products of these vectors. We do this for 1000 times and plot the maximum of maximums of the pair-wise dot products in Figure 1. We see that as size of vectors increase, overlapping of any two of them decreases. In the light of this empirical finding, we claim that it is more robust to generate high dimensional vectors to prevent the GAN attack. However, even if we use a fixed layer to keep the number of trainable parameters constant, we observed that increasing d_k makes training of participants difficult. Therefore, we suggest to consider d_k as an hyper-parameter to be tuned.

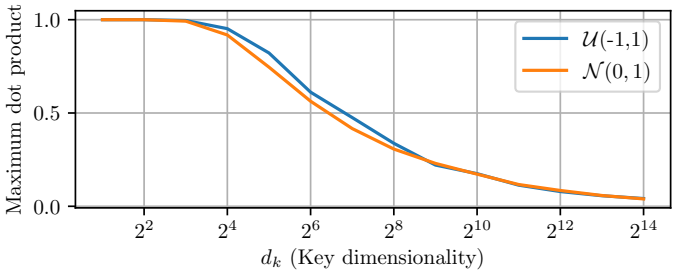


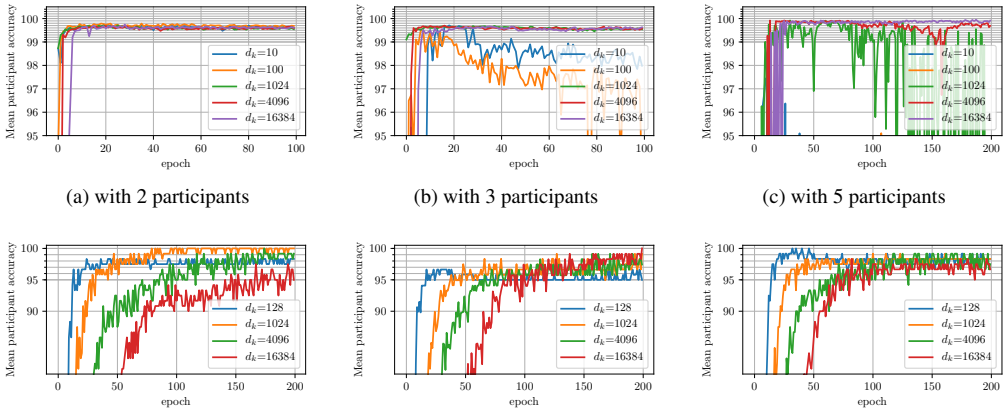
Figure 1: Dots products of two ℓ_2 normalized random vectors, whose entries are sampled from either $\mathcal{N}(0,1)$ or $\mathcal{U}$(-1,1). x-axis represents the size of the vectors and y-axis shows the maximum dot product among 1000 different trials. In each trial, 100 vectors are sampled and the maximum of all pairwise dot products is computed.

5.3 COLLABORATIVE LEARNING EVALUATION

Before showing the robustness of the proposed model against the GAN attack, we evaluate our model in CLF with 2, 3 and 5 participants when no adversary is present. First we investigate how key size d_k effects the learning curves of participants. For MNIST, we experiment with 5 different values of $d_k \in \{10, 100, 1024, 4069, 16384\}$. When there are 2 participants, training with all keys yields almost equal performance. However, shared model is likely to deteriorate when class labels are split among more participants due to overfiting of shared model parameters to local dataset of any one of the participants. However, we show that even for such cases, sufficiently high d_k overcomes this problem. Quite similar observations hold for Olivetti Faces as well. For this dataset we experiment with 4 different key sizes as $d_k \in \{128, 1024, 4069, 16384\}$. As we highly regularize models of participants during training (*Model parameters per image in dataset* statistics are quite different for these datasets. Since we cannot make shallower or narrower networks work for Olivetti Faces, we regularize local models to balance bias-variance), an accurate shared model can be built among participants. Top and bottom rows of Figure 2 shows the results obtained with (from left to right) 2, 3 and 5 participants for MNIST and Olivetti Faces, respectively. We observed that building a shared model that includes a fixed layer is challenging even for MNIST in the absence of 3 or more participants. Whereas, shared model can be regularized to work well in local datasets of all participants for Olivetti Faces.

5.4 PREVENTING GAN ATTACK

To show the robustness of our proposed technique against the GAN attack, we perform 3 sets of experiments: (i) one of the class keys is given to the adversary ($\psi(c_{\text{attack}}) = \psi(c)$ for any c) when d_k is 256 and 128 for MNIST and Olivetti Faces, respectively. (ii) adversary generates random keys that are δ far (measured in Euclidean distance) from any class key ($\|\psi(c_{\text{attack}}) - \psi(c)\| = \delta$ for any



(a) with 2 participants (b) with 3 participants (c) with 5 participants

(d) with 2 participants (e) with 3 participants (f) with 5 participants

Figure 2: From left to right, mean participant accuracies (MPAs) obtained in collaborative learning of 2, 3, and 5 participants on MNIST (top row) and on Olivetti Faces (bottom row) respectively.

c) when d_k is 256 and 1024 for MNIST and Olivetti Faces, respectively, (iii) adversary generates random keys when $d_k \in \{128, 1024, 4096, 16384\}$ for MNIST and Olivetti Faces.

In (i) we demonstrate the extreme case in which adversary guesses the exact same key of any class in CLF. In Figure 3 and Figure 4 we show that adversary achieves its in both datasets. Note however that guessing the exact same key for a specific class of a target participant has a negligible probability, especially for larger class key sizes.

In (ii) we conduct a study to loosely approximate the Euclidean distance, $\widehat{\delta}$, between any class key of any participant and random keys generated by the adversary. We do this study to show the relationship between the success of the adversary and the Euclidian distance between the guessed key (by the adversary) and the actual class keys of the participants (adversary needs to generate keys which satisfy $\|\psi(c_{\text{attack}}) - \psi(c)\| \leq \widehat{\delta}$ for any c). We quantify this study by visually inspecting the reconstructions of the adversary. From Figure 5 we deduce $\widehat{\delta}$ value as 1.2 for MNIST and as 0.1 for Olivetti Faces. We observed that $\widehat{\delta}$ is inversely proportional to d_k. Note that we do not generate the keys of the adversary randomly for this experiment, instead we forced the keys generated by the adversary to be a certain Euclidian distance away from the actual class keys of the target participant.

In (iii) we show that our model is robust against the GAN attack when there is no constraint on key generation (*i.e.* all keys are randomly generated by the participants, including the adversary). It can be seen in Figure 6 that for sufficiently large key sizes (d_k) the GAN attack fails on both MNIST and Olivetti Faces. That is, the adversary is not able to train its local generator when $\psi(\widetilde{c})$ is just ℓ_2 normalized random key.

(a) (b) (c) (d) (e)

Figure 3: We split MNIST among 2 participants, one being the adversary. The victim has samples of digits 0, 1, 2, 3, and 4. When the exact same key of one of these digits is guessed by the adversary, the local generator of the adversary reconstructs samples shown in (a), (b), (c), (d) and (e), respectively.

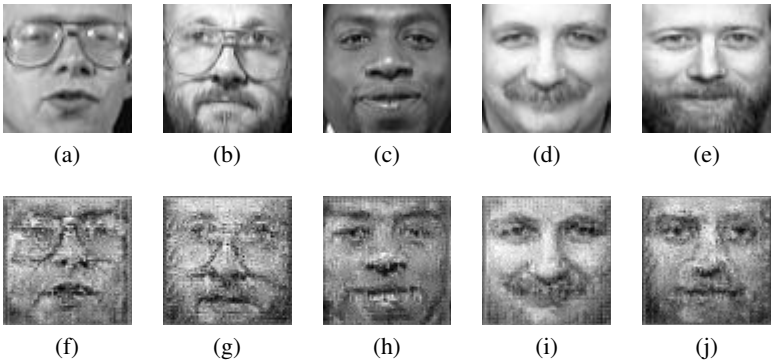


Figure 4: We split Olivetti Faces among 2 participants, one being the adversary. Each participant has face photos of 20 people. We randomly select 5 class labels from the victim for which adversary guesses the exact same key. The local generator of adversary is able to capture details in data distribution of these classes. Samples shown in (f), (g), (h), (i) and (j) are observed from the local generator of the adversary and they are visually close to the original samples shown in (a), (b), (c), (d) and (e), respectively.

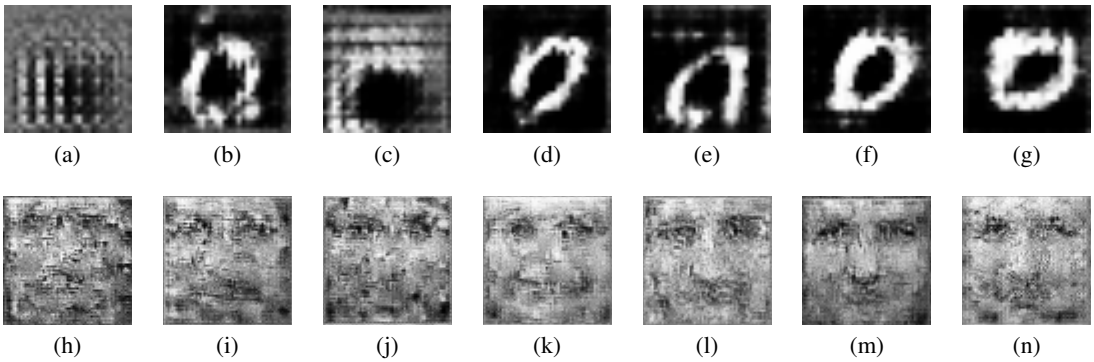


Figure 5: We approximate the maximum Euclidean distance between any class key and $\psi(c_{\text{attack}})$ necessary for the adversary to succeed in attack. From left to right, reconstructions of the adversary when it generates random keys that are $\delta \in \{1.3, 1.2, 1.1, 1.0, 0.5, 0.1, 0.01\}$ far from class key of digit-0 in MNIST (upper row) and person-24 in Olivetti Faces (bottom row), respectively.

6 Conclusions

Traditional collaborative learning frameworks (Shokri & Shmatikov, 2015) makes large-scale machine learning possible when data owners do not want to share their datasets due to privacy concerns. However, very recently, such techniques are shown to be prone to powerful GAN attacks (Hitaj et al., 2017). In this work, we have proposed a novel collaborative learning technique that is resilient to the GAN attack. More specifically, we have reformulated the training objective of participants by introducing random class keys for each class in the framework. This key-based approach provides effective learning of participants. Moreover, by utilizing high dimensional keys, class scores of an input is protected against an active adversary that may run the GAN attack. We have verified our claim by showing that (i) the adversary is no longer able to choose which class to exploit, and (ii) generator trained by the adversary cannot capture data distribution well enough to expose any class in the framework.

As future work, we plan to (i) investigate the performance of key-based approach in supervised classification setting without privacy considerations, (ii) use differential privacy concept to further reduce the success probability of the attack (i.e., to make δ even smaller), (iii) evaluate the case when the label sets of participants overlap, and (iv) evaluate our work in more challenging datasets.

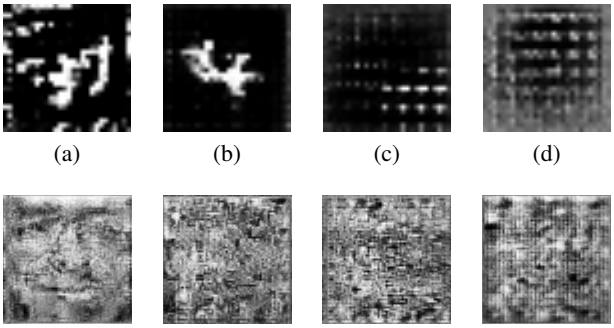


Figure 6: All keys are randomly generated by the participants, including the adversary. From left to right, reconstructions of adversary when size of class keys $d_k \in \{128, 1024, 4096, 16834\}$, respectively (upper row for MNIST and bottom row for Olivetti Faces). For small values of d_k, GAN can learn a mode, which is clearly not representative enough for a realistic digit or a face of person. GAN produces quite noisy images when d_k is much larger.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4139-4. doi: 10.1145/2976749.2978318. URL http://doi.acm.org/10.1145/2976749.2978318.

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

Martín Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *CoRR*, abs/1701.04862, 2017. URL http://arxiv.org/abs/1701.04862.

Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 214–223, 2017. URL http://proceedings.mlr.press/v70/arjovsky17a.html.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *CoRR*, abs/1612.02136, 2016. URL http://arxiv.org/abs/1612.02136.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Cynthia Dwork. Differential privacy. In *Encyclopedia of Cryptography and Security*, pp. 338–340. Springer, 2011.

Matt Fredrikson, Somesh Jha, and Thomas 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. ACM, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ian J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017. URL http://arxiv.org/abs/1701.00160.

Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. MS-Celeb-1M: A dataset and benchmark for large scale face recognition. In *European Conference on Computer Vision*. Springer, 2016.

Briland Hitaj, Giuseppe Ateniese, and Fernando Pérez-Cruz. Deep models under the GAN: information leakage from collaborative deep learning. *CoRR*, abs/1702.07464, 2017. URL http://arxiv.org/abs/1702.07464.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Yann LeCun, Corinna Cortes, and Chris Burges. Mnist handwritten digit database. http://yann.lecun.com/exdb/mnist/. (Accessed on 10/25/2017).

Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. Semantic segmentation using adversarial networks. *CoRR*, abs/1611.08408, 2016. URL http://arxiv.org/abs/1611.08408.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. URL http://arxiv.org/abs/1411.1784.

Anh Nguyen, Jason Yosinski, Yoshua Bengio, Alexey Dosovitskiy, and Jeff Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. *CoRR*, abs/1612.00005, 2016. URL http://arxiv.org/abs/1612.00005.

Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *CoRR*, abs/1610.05755, 2016. URL http://arxiv.org/abs/1610.05755.

NhatHai Phan, Yue Wang, Xintao Wu, and Dejing Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. 2016.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 29*, pp. 2234–2242. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6125-improved-techniques-for-training-gans.pdf.

Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*, pp. 138–142. IEEE, 1994.

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

I. Tolstikhin, S. Gelly, O. Bousquet, C.-J. Simon-Gabriel, and B. Schölkopf. Adagan: Boosting generative models. 2017.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. In *International Conference on Learning Representations 2017 (Conference Track)*, 2017. URL https://openreview.net/forum?id=S1X7nhsxl.

Raymond Yeh, Chen Chen, Teck-Yian Lim, Mark Hasegawa-Johnson, and Minh N. Do. Semantic image inpainting with perceptual and contextual losses. *CoRR*, abs/1607.07539, 2016. URL http://arxiv.org/abs/1607.07539.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016.

A PSEUDO CODE FOR KEY PROTECTED CLASSIFICATION FOR GAN ATTACK RESILIENT COLLABORATIVE LEARNING

Algorithm 1 Key Protected Classification for GAN Attack Resilient Collaborative Learning

Pre-Training Phase:

Participants declare all class labels for which they have samples in their datasets. Adversary declares one more label, c_{fake} for labeling samples generated by its local generator.

Participants agree on the following list:

1. a common neural net architecture and objectives defined for training
2. (θ_u), percentage of parameter uploads to server,
3. (θ_d), percentage of parameter downloads from server,
4. participant order, fixed and sequential
5. d_k, size of class keys
6. learning rate for optimizer
7. λ, regularization coefficient

Participants generate random ℓ_2 normalized class keys, $\psi_i(c)$ for each of their classes

Attacker generates an extra class key, $\psi(c_{\text{attack}})$, for exploiting a random class

Training Phase

```
for epoch = 1 to n_epochs do
    for part_i in Participants do
        download θ_d fraction of parameters from PS
        replace downloaded parameters with local ones
        if (part_i == adversary) then
            run generator on local classifier by targeting c_attack
            generate M samples from generator
            label the generated samples as c_fake
            merge generated samples with local dataset
        end if
        train local model with local dataset
        upload θ_u fraction of differences in parameters to PS
    end for
end for
Participants make their keys publicly available.
```