Machine Learning (2021) 110:651–674
https://doi.org/10.1007/s10994-021-05951-6

Protect privacy of deep classification networks by exploiting their generative power

Jiyu Chen[1] · **Yiwen Guo**[2] · **Qianjun Zheng**[3] · **Hao Chen**[1]

Received: 20 September 2020 / Revised: 15 December 2020 / Accepted: 23 January 2021 /
Published online: 13 April 2021

Abstract
Research showed that deep learning models are vulnerable to membership inference attacks, which aim to determine if an example is in the training set of the model. We propose a new framework to defend against this sort of attack. Our key insight is that if we retrain the original classifier with a new dataset that is independent of the original training set while their elements are sampled from the same distribution, the retrained classifier will leak no information that cannot be inferred from the distribution about the original training set. Our framework consists of three phases. First, we transferred the original classifier to a Joint Energy-based Model (JEM) to exploit the model's implicit generative power. Then, we sampled from the JEM to create a new dataset. Finally, we used the new dataset to retrain or fine-tune the original classifier. We empirically studied different transfer learning schemes for the JEM and fine-tuning/retraining strategies for the classifier against shadow-model attacks. Our evaluation shows that our framework can suppress the attacker's membership advantage to a negligible level while keeping the classifier's accuracy acceptable. We compared it with other state-of-the-art defenses considering adaptive attackers and showed our defense is effective even under the worst-case scenario. Besides, we also found that combining other defenses with our framework often achieves better robustness. Our code will be made available at https://github.com/ChenJiyu/meminf-defense.git.

Keywords Data privacy · Membership inference attack · Generative modeling · Deep neural networks

Editors: Annalisa Appice, Sergio Escalera, Jose A. Gamez, Heike Trautmann.

✉ Jiyu Chen
jiych@ucdavis.edu

Yiwen Guo
guoyiwen.ai@bytedance.com

Qianjun Zheng
bachjin@mail.ustc.edu.cn

Hao Chen
chen@ucdavis.edu

[1] University of California, Davis, USA

[2] ByteDance AI Lab, Beijing, China

[3] University of Science and Technology of China, Hefei, China

1 Introduction

Deep learning models are widely deployed, and many of them are continuously trained by data collected from users. As such, privacy becomes a major concern.

Researchers have proposed several privacy attacks on deep learning models, such as inference attacks (Shokri et al., 2017; Hayes et al., 2019; Nasr et al., 2019), inversion attacks (Fredrikson et al., 2015), and model stealing attacks (Tramèr et al., 2016; Wang & Gong, 2018). We focus on defending against membership inference attacks, which aims to infer whether an example is among the target model's training data. The ability of resisting membership inference attacks is crucial in practical scenarios where the privacy of training data is of importance. For example, while a patient consents to use her medical record to train a sensitive disease classifier, she does not want the model to reveal that her record is part of the training set (and hence she has the disease).

Researchers considered overfitting as a major contributing factor to inference attacks (Shokri et al., 2017; Yeom et al., 2018). Therefore, many defenses aimed to reduce overfitting by forcing the model to learn the distribution instead of memorizing the data by applying different regularization strategies, such as weight decay and dropout (Srivastava et al., 2014). However, recently researchers also showed that overfitting is a sufficient but not necessary condition for performing privacy attacks (Yeom et al., 2018; Jain et al., 2015; Salem et al., 2018), which means that merely reducing overfitting may be inadequate. Others proposed certifiable training algorithms (Chaudhuri et al., 2011; Abadi et al., 2016; Wang et al., 2018) to theoretically guarantee that the trained model is differentially private. Section 6 will discuss more defenses. Among these defenses, the original training data are always exposed to model training, which makes it challenging to defend against privacy attacks. Moreover, some defenses bear conditional restrictions, require additional architectures and data, or need large time budgets.

In this paper, we propose a novel framework for defending a pre-trained deep learning classifier against inference attacks without requiring additional architectures and datasets, or special training algorithms. Our key insight is that if we retrain the model with a generated dataset that (1) is from the same distribution as the original training set but (2) is independent of the original training set, then no information about the original training set that cannot be inferred from the distribution itself will be available to the retrained model. Therefore, the retrained model achieves information-theoretic security against inference attacks. However, we do not know the true distribution of the underlying data in practice, so we instead turn to approximate it using the empirical distribution via generative models based on our original datasets.

Our framework consists of three phases. First, we train a generative model using the original training data. Second, we sample from the generative model to create a new dataset. Finally, we use the new dataset to retrain or fine-tune the original classifier.

Applying generative models creates two potential problems. First, the divergence between the original dataset's empirical distributions and our generated dataset may reduce the retrained classifier's accuracy. Second, the generated dataset may not be completely independent of the original dataset. To address these problems, we need to carefully select generative models and evaluate our retrained models on classification accuracy and robustness against inference attacks.

Several generative models can produce high-quality examples, including generative adversarial networks (GANs) (Goodfellow et al., 2014), variational auto-encoders (VAEs) (Kingma and Welling 2013), and energy-based models (EBMs) (Ackley et al.,

1985). Recently, Grathwohl et al. introduced joint energy-based models (JEM). A JEM consists of a discriminative model and a generative model that share the same deep network architecture (Grathwohl et al., 2019). We selected JEM to provide the required generative ability for several reasons. First, since JEM's generative model can reuse the same network architecture of the pre-trained classifiers, we need no architectural engineering. Second, sampling from EBMs (including JEMs) is theoretically guaranteed to be asymptotically in the true data distribution. Finally, training JEMs requires less computation, especially when it is possible to transfer the pre-trained classifier to JEM. This transfer learning is efficient because it can inherit the useful features that the pre-trained classifier has already learned.

We have performed extensive evaluations for two main objectives. One is to provide insight into how different transferring to JEM and model fine-tuning strategies would result in our final results. The other is to demonstrate the performance of the model produced by our framework under popular membership inference attacks.

For the first objective, we compared different amounts of network transfer from the original classifier to the JEM and found that JEM training failed to converge when we transferred all the layers from the original classifier to the JEM. By contrast, transferring only the early convolutional layers resulted in the fastest convergence. We also compared retraining the original classifier with fine-tuning it and found that fine-tuning increased both classification accuracy and the attacker's membership advantage moderately. Meanwhile, for the second objective, our evaluation shows that our framework reduced the attacker's *membership advantage* from 32.91% on the original model to 2.66% on the retrained model on CIFAR-10, and from 25.28% on the original model to 4.55% on the retrained model on SVHN,[1] while our retrained model suffers a moderate decrease in classification accuracy. More importantly, by comparing our defense with other state-of-the-art defenses under adaptive attacks, we show that our defense is effective even under the worst-case scenario and can provide better accuracy-robustness tradeoff when combined with other defenses.

2 Background

2.1 Membership inference attacks

One main category of privacy attacks consists of *inference attacks*, which contains *membership inference attacks* and *attribute inference attacks*. Membership inference attacks aim to infer whether an example was in the target model's training dataset, e.g., inferring whether a patient's record was used in medical research. Shokri et al. (2017) designed a black-box membership inference attack against machine learning models. Subsequently, researchers introduced several variants of the attack, such as attacks on GANs (Hayes et al., 2019), VAEs (Hilprecht & Härterich, 2019), model explanations (Shokri et al., 2019), and collaborative learning models (Nasr et al., 2019). We focus on mitigating membership inference attacks on DNN classifiers in this paper.

A well-known membership inference attack is the shadow-model attack (Shokri et al., 2017). It requires the attacker to train several shadow models and attack models. To attack

[1] Membership advantage = 2 * attack accuracy − 1

a victim model, first, the attacker collects or synthesizes data from the same domain as the victim classifier's training and test data, and divides the data into several private training and test sets. Then, the attacker uses each of the private training set to train one *shadow model* f^i_{shadow} to mimic the behavior of the victim classifier. Next, the attacker sends all the examples in each private training and test set to its corresponding shadow model to create a dataset D that contains the tuple $(y, f^i_{\text{shadow}}(x), I(x))$ for each example x, where y is the class label of x, $f^i_{\text{shadow}}(x)$ is a vector containing the outputs of the i^{th} shadow model, and $I(x)$ indicates whether x is used for training the shadow models. Finally, the attacker partitions D based on the examples' class labels and uses them to train one attack model per class to distinguish the training data from the others.

2.2 Energy-based models

Energy-based models (EBMs) was proposed long ago (Ackley et al., 1985) and has been largely improved during the years (LeCun & Huang, 2005; Hinton et al., 2006). Currently, energy-based models can achieve state-of-the-art generative power compared with other generative models (Du & Mordatch, 2019), such as GANs and VAEs, but have more flexible architectures because they directly model reasonable energy representations.

An energy-based model represents the probability density function $p(x), x \in \mathbb{R}^D$ as

$$p_\theta(x) = \frac{\exp(-E_\theta(x))}{Z(\theta)}, \tag{1}$$

where $E_\theta(x) : \mathbb{R}^D \to \mathbb{R}^D$ is the *energy function* parameterized by a set of learnable parameters θ (e.g., a neural network), and $Z(\theta) = \sum_{x' \in X} \exp(-E_\theta(x'))$ is a normalizing constant known as the *partition function*.

Computing $Z(\theta)$ directly is usually intractable. Instead, we can train the energy-based model by computing the gradient of $p_\theta(x)$ w.r.t. θ:

$$\nabla_\theta \log(p_\theta(x)) = \mathbb{E}_{p_\theta(x')} \nabla_\theta E_\theta(x') - \nabla_\theta E_\theta(x). \tag{2}$$

The detailed derivation of Eq. (2) can be found in "Appendix 1". At each training step, we approximate the expectation in Eq. (2) by sampling from the current model, using Stochastic Gradient Langevin Dynamics (SGLD) (Welling & Teh, 2011) which is an iterative sampling method:

$$x_{t+1} = x_t - \frac{\lambda}{2} \nabla_{x_t} E_\theta(x_t) + \epsilon, \tag{3}$$

where x_0 is uniformly drawn from the valid input domain of the model, the step size λ decays polynomially, and the noise ϵ is drawn from the normal distribution $N(0, \lambda)$. SGLD is efficient and is provable to asymptotically produce samples from the target distribution.

3 Design

3.1 Threat model

A DNN classifier takes an example as input and outputs a vector describing the probability that the input belongs to each class (or label). In the concerned *membership inference*

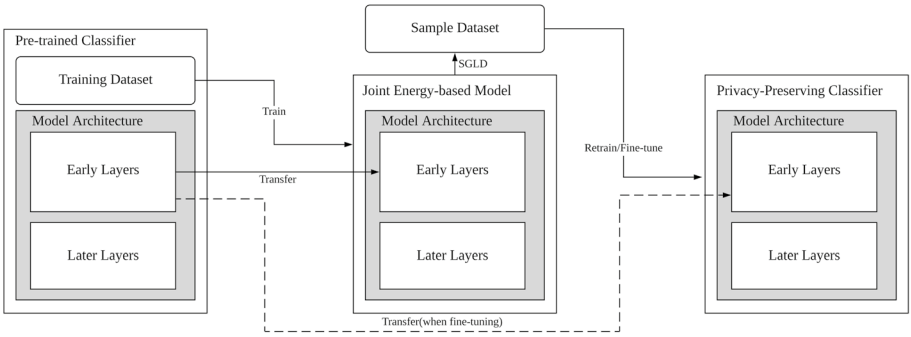


Fig. 1 The three-phase framework of our defense. Transferring the pre-trained classifier to a JEM, sampling a new dataset by SGLD, retraining/fine-tuning a privacy-preserving classifier. All phases share a single model architecture

attacks, the attacker aims to determine if a given example was used to train the classifier. As defenders, our goal of this paper is to protect a pre-trained classifier from such attacks.

We assume the following threat model in this paper:

- The attacker has gray-box access to the classifier. She knows the architecture of the classifier and can query it using arbitrary examples. However, she knows neither the weights nor the activation of nodes inside the classifier.
- Even though the attacker does not know the training data of the original classifier, she knows the distribution of the training data and can access other datasets from the same distribution.

This threat model applies to many real-life scenarios. For example, a company provides face recognition service via an API. The attacker can query the API. She may even guess the architecture of the classifier. However, she can access neither the weights nor the activation of nodes inside the classifier. She has no access to the training data, but she can collect her own face datasets and use them to launch an attack, e.g., by training a shadow model (Shokri et al., 2017).

3.2 Defense framework

The goal of a classifier is to learn the conditional probability distribution of data. Unfortunately, DNN classifiers tend to overfit the training data and thus enabling membership inference attacks. Our insight is that if we replace the training dataset with a new dataset that (1) is independent of the training data, and (2) is from the same distribution of the training data, then we can mitigate membership inference attacks while retaining the test-set accuracy of the original classifier.

One straightforward way is to transform individual training examples, e.g., by adding noise. However, this approach has an irreconcilable dilemma: If the transformation is small, then the transformed example will be highly correlated with the original example; on the other hand, if the transformation is large, then the transformed example may deviate from the distribution.

We proposed a new approach to satisfy both properties. Instead of transforming individual example, we first learned the distribution of the training data (Sect. 3.2.1), and then sampled from the distribution to create a new dataset (Sect. 3.2.2). Finally, we used the new dataset to train the classifier (Sect. 3.2.3). Figure 1 summarizes our pipeline of performing these steps.

3.2.1 Train the generative model

We learned the distribution of the training data by training a generative model. GAN is a popular option, but we would need to design a suitable GAN architecture and carefully tune its hyper-parameters to suit the training data. We aimed to achieve the goal by exploiting the original classifier's generative power such that no extra architecture engineering is needed. On this point, we created a Joint Energy-based Model (JEM) (Grathwohl et al., 2019), which reused the architecture of the original classifier but changed its log-likelihood loss to

$$\log p_\theta(x, y) = \log p_\theta(x) + \log p_\theta(y|x), \tag{4}$$

where

$$p_\theta(x) = \frac{\exp(-E_\theta(x))}{\sum_{x' \in X} \exp(-E_\theta(x'))} \quad \text{and} \quad p_\theta(y|x) = \frac{\exp(f_\theta(x)[y])}{\sum_{y'} \exp(f_\theta(x)[y'])}. \tag{5}$$

The energy function here is defined as $E_\theta(x) = -\log \Sigma_y \exp(f_\theta(x)[y])$ and $f_\theta(x)[y]$ is the output logit of the DNN corresponding to the label y on the input x computed by the original classifier.

When training the JEM, the gradient of $\log p_\theta(y|x)$ is readily available from the DNN using back-propagation (as $\log p_\theta(y|x)$ is simply the negative cross-entropy loss of the DNN when used as a classifier). To compute the gradient of $\log p_\theta(x)$, we approximated the expectation in Eq. (2) by sampling from the current model. Specifically, we applied Stochastic Gradient Langevin Dynamics (SGLD, Eq. 3) for sampling, which is the common choice in many recent energy-based models' training.

The original goal of JEM was to train both a discriminative model and a generative model from scratch (Grathwohl et al., 2019), but that requires sophisticated parameter turning and converges slowly. By contrast, since we already had the original classifier as a descent discriminative model on which we wished to defend against membership inference attacks, we attempted to transfer the classifier to the JEM. Since the JEM and the classifier had the same architecture (but with different loss functions), we simply used the weights of the original classifier to initialize the JEM. We compared different transfer strategies (and the detailed results are deferred to Sect. 4.2):

- *Transfer all*: initialized all the weights in the JEM to their corresponding values in the original classifier.
- *Transfer some*: for some layers in the JEM, initialized their weights to their corresponding values in the original classifier; for the other layers, initialize their weights randomly.
- *Transfer none*: initialized all the weights in the JEM randomly.

3.2.2 Sample from the generative model

In fact, when we trained the JEM, according to Eqs. (2) and (3), we needed to sample from the generative model of the JEM at every training iteration. However, sampling that starts from random noise every time can result in a large computational overhead. In practice, we kept the examples that we used to approximate the gradient of $\log p_\theta(x)$ during SGLD in a *replay buffer*. Specifically, the replay buffer works under the following procedure. Initially, the fixed-sized replay buffer is filled with random noise. Whenever we sample a new example, we will randomly select an entry in the replay buffer and use it as the starting point of SGLD, and the new sample will replace the original entry as a new seed for the next time selected. The replay buffer is dynamically updated at each training iteration and thus can gradually boost the sampling speed and quality.

After we obtained a well-trained JEM, we sampled from its generative model $p_\theta(x)$ to create a set for training the final privacy-preserving classifier. We leveraged the replay buffer to get samples from the generative model similar to in training. We also randomly selected seeds from the replay buffer and then used SGLD to acquire samples according to Eq. (3). We will discuss how the replay buffer size would affect the final classifier's generation quality and performance in Sect. 4.2.1.

3.2.3 Fine-tune the classifier

After we collected enough samples as a new training set from the generative model, we were ready to train the final privacy-preserving classifier. A straightforward way was to retrain the final classifier from scratch by randomly initializing all its weights. Retraining provides the best privacy protection because no information was left over from the original privacy-sensitive training data. However, if the generated data were inadequately diversified as they cannot approximate the true data distribution better than the original training data, the resulting classifier's test accuracy might deteriorate comparing to the original one. Considering the trade-off between classification accuracy and membership robustness, we can also fine-tune the original classifier either fully or partially. We compared these three strategies (train from scratch, fine-tune partially, and fine-tune fully) in terms of test accuracy and robustness against membership inference attack in Sect. 4.3, where the proposed framework will also be compared with other effective defenses to membership inference attacks empirically.

As we will see in Sect. 4.3, by the nature of JEM and SGLD sampling, the generated dataset would have a similar distribution with the real dataset. Thus, the model would only drop little accuracy even if it was retrained from scratch by the generated dataset. We provide more visualized evidence in "Appendix 5".

4 Evaluation

4.1 Experiment settings

To evaluate the effectiveness of our approach, we performed an extensive empirical study. First, we evaluated our JEM on its discriminative model's accuracy and the quality of the examples sampled from its generative model. Then we evaluated how different training

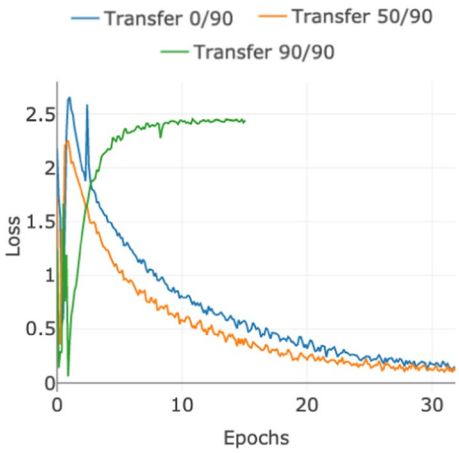


Fig. 2 Training curves of different transfer learning schemes

settings, including replay buffer size and early stopping, would affect our result. Next, we evaluated the retrained/fine-tuned classifier on classification accuracy and robustness against membership inference attacks. Finally, we compared our defenses with other state-of-the-art defenses to further demonstrate the effectiveness of our defense.

We used WideResNet (Zagoruyko & Komodakis, 2016) as our network architecture, the same as used by Grathwohl et al. (2019). We used two datasets: CIFAR-10 contains colored natural objects of 10 classes (Krizhevsky et al., 2009), and SVHN contains colored digital photos of street house numbers (Netzer et al., 2011).

Unless specified otherwise, all the models (JEMs, shadow models, and attack models) were selected at the point of best validation accuracy.

4.2 Training efficiency of JEM

JEM contains a discriminative model (classifier) $p(y|x)$ and a generative model $p(x)$, both sharing the same network architecture. When training JEM from scratch, it requires sophisticated parameter tuning and considerable time to converge. Since we already had the pre-trained original classifier, our JEM used the same architecture as the original classifier. We compared the three schemes for transferring the weights from the original classifier to our JEM introduced in Sect. 3.2.1. The WideResNet network in our JEM has 90 convolutional layers (the others are mainly batch-normalization layers). The three schemes are:

- Transfer all (or transfer 90/90): Transfer all the 90 convolutional layers from the original classifier to JEM.
- Transfer some (or transfer 50/90): Transfer the early 50 convolutional layers from the original classifier to JEM.
- Transfer none (or transfer 0/90): Do not transfer.

Any weight not transferred from the original classifier was randomly initialized in the JEM.

Figure 2 shows the training curves of the transfer learning schemes. It shows that when we used the entire pre-trained classifier to initialize the JEM (i.e., transfer all), training was hard to converge. We explain this difficulty by the following phenomenon. Given an example and its label (x, y), a trained classifier is expected to output a high confidence in class

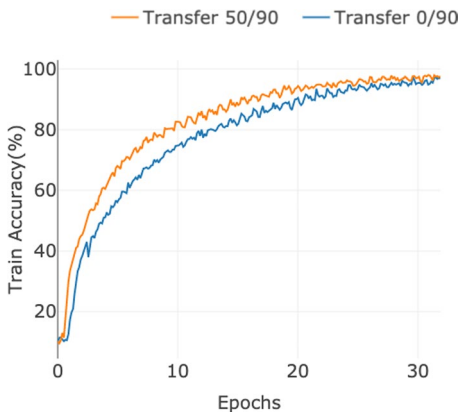
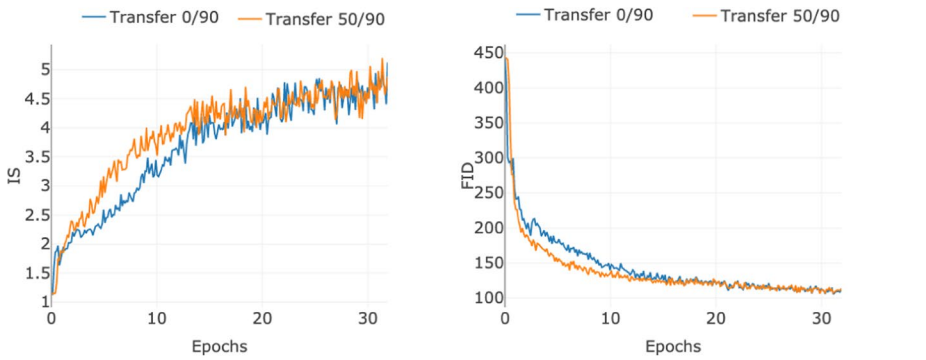


Fig. 3 Accuracy of JEM's discriminative model during training



(a) Inception score (higher is better) **(b)** Fréchet Inception Distance (lower is better)

Fig. 4 Image quality during training. At each epoch, we randomly sampled 500 images for calculating the Inception Score and Fréchet Inception Distance

label y, e.g., a high $p(y|x)$. However, the converse is not true, which means that even when the classifier outputs a high confidence in the label y on another example x', we cannot conclude that x' is likely from the same distribution as x. So the original network, which was solely optimized over $p(y|x)$, may have very poor performance in terms of describing $p(x)$. This phenomenon also makes deep learning models vulnerable to adversarial examples (Szegedy et al., 2013), which are specially crafted examples to make models output different decisions than humans.

By contrast, when we transfer only some of the convolutional layers from the original classifier to the JEM, training became more stable and easily converged. This is likely because early convolutional layers mostly contain low-level features extracted from the training images, which are more likely to be shared between the discriminative model $p(y|x)$ and the generative model $p(x)$. Consequently, between transferring some layers (i.e., transfer 50/90) and training from scratch (i.e., transfer 0/90), the former also converged faster. See for example Fig. 3, it shows that the accuracy of the discriminative model during training converged faster when we transferred some convolutional layers than training from scratch.

Table 1 The impact of the size of the replay buffer on the accuracy of the retrained classifier. We retrained the classifier for 10 epochs

Buffer size	Classification accuracy (%)	
	CIFAR-10	SVHN
10,000	54.27	81.62
50,000	74.34	84.14
100,000	80.19	87.98

Figure 4 further shows how the Inception Score (IS) (Salimans et al., 2016) and Fréchet Inception Distance (FID) (Heusel et al., 2017) of the images sampled from the generative model of the JEM changed during training. Inception Score and Fréchet Inception Distance are commonly used to assess image generation in terms of image quality and variety. We can see that the transfer learning scheme (i.e., transfer 50/90) improved image quality faster during training than training from scratch.

4.2.1 Size of the replay buffer

After training the JEM, we sampled images from its generative model to create a dataset for fine-tuning/retraining the classifier for protecting data privacy. As Sect. 3.2.2 described, during each iteration of JEM training, we stored the example that was used to approximate the gradient of $p(x)$ in a replay buffer. To get a sample from the generative model, we randomly selected an entry from the replay buffer as the start point and then used SGLD to acquire a sample. In other words, the replay buffer contained high-quality seeds based on which we sampled from the distribution. On the one hand, larger replay buffers allow us to acquire samples with greater variety, which will improve the generalizability of the retrained classifier. But on the other hand, since the size of the replay buffer is linear in the number of training iterations, a larger buffer will require longer training time.

Table 1 shows how replay buffer size (i.e., number of seeds in the buffer) affects the accuracy of the retrained classifier. We trained the JEM with different buffer sizes. On each buffer size, we sampled the same number of images from the replay buffer and used them to retrain the original classifier (i.e., used the architecture of the original classifier but initialized its weights randomly). As we expected, a larger buffer size increased the accuracy of the retrained model. The user of our framework needs to balance the trade-off between the training time of the JEM and the accuracy of the retrained classifier.

In the following experiments, we set the replay buffer size to 100 000 and transferred the early 50 convolutional layers of the original classifier to the JEM.

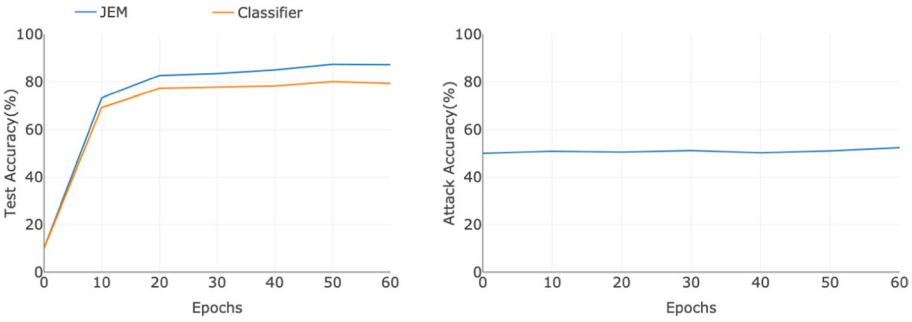
4.2.2 Early stopping

Normally we stop the training of the JEM when its generative model achieves the best validation accuracy. However, this usually takes a long time. Since our goal is not to train the best generative model but to train a good enough generative model to generate high-quality examples, we could stop the training early, but how will this affect the quality of the sampled images and the retrained/fine-tuned classifier's accuracy and robustness against membership inference attacks?

Figure 5 compares the images sampled at the end of the 10th epoch (Fig. 5a) with those at the end of the 50th epoch (Fig. 5b). It shows that even though more photo-realistic images appeared at the later epoch, some photo-realistic images also appeared at the earlier

(a) Sample from the 10th epoch **(b)** Sample from the 50th epoch

Fig. 5 Synthesized images from different training epochs for CIFAR-10. The left were sampled from the 10th epoch, and the right were sampled from the 50th epoch. While the right samples have higher quality, there are also lots of photo-realistic samples in the left, which only took 1/5 training time



(a) Classification accuracy **(b)** Attack accuracy

Fig. 6 Impact of early stopping on the regular accuracy and on the attack accuracy of the retrained CIFAR-10 classifier

epoch. In our experiments, we used the original classifier as a quality filter where a sampled image passes the filter only if the original classifier outputs a high confidence in its class label. This filter allows us to stop the JEM training early "safely".

Figure 6 shows the impact of early stopping of JEM training on the privacy and test-set accuracy of the retrained classifier. Figure 6a shows that the accuracy of the discriminative model of the JEM increased rapidly during the first 10 epochs but much slower after that, and more importantly, the same trend can be observed in the test-set accuracy of the retrained classifier (trained using data generated by the JEM model collected at the corresponding epoch). Therefore, early stopping may be desirable for better efficiency of our framework when the discriminative model of JEM reaches an acceptable classification accuracy. Figure 6b shows that the attack accuracy barely changed as the training of JEM progressed, i.e., in the context of privacy-preserving, retraining or fine-tuning the final

classifier using data generated after 10 epochs of the JEM training can be as effective as using data generated after 50 epochs. This phenomenon supports our hypothesis that the data sampled from the generative model of JEM are independent of the original training data, no matter how long the JEM has been trained.

4.3 Privacy of the final classifier

4.3.1 Robustness against shadow-model attack

After retraining or fine-tuning the original classifier, we performed the shadow-model attack (Shokri et al., 2017) on (1) the original classifier, (2) the fine-tuned classifiers, and (3) the retrained classifier using our framework. We use *Membership advantage* (or simply *advantage*) to measure the amount of information leaked to the attacker (Yeom et al., 2018):

$$\text{Membership advantage} = 2 \cdot \text{Attack accuracy} - 1$$

which equals the difference between the true positive rate and the false positive rate of the attacker.

The row with the label "Ours" in Table 2 reports the performance of our defenses. It shows that our retrained or fine-tuned classifiers significantly reduced the attacker's advantage, from 32.91 to 2.66% on CIFAR-10 and from 25.28 to 4.55% on SVHN. It also compares fine-tuning the original classifier and retraining it. The results show that the fully fine-tuned classifier (i.e., obtained by fine-tuning all the 90 convolutional layers) achieved the highest accuracy while giving the attacker slightly more advantage than the retrained classifier, while the retrained classifier gave the attacker the smallest advantage with a slight decrease in the accuracy. Besides classification accuracy and attacker advantage, another factor to consider when making the trade-off is training time, as fine-tuning takes less time to converge than retraining.

4.3.2 Comparison with other defenses

For comparison, we also implemented several state-of-the-art defenses that have been shown effective against membership inference attacks.

We have selected the following defenses for comparison: standard regularizations (L^2 Reg. and Dropout) (Shokri et al., 2017), Min–Max adversarial regularization (Nasr et al., 2018), DP-SGD (Abadi et al., 2016). Note that we mostly selected regularization-based defenses for comparison, since output masking based defenses, such as MemGuard and output vector truncation, have been shown ineffective under black-box label-only attacks (Choo et al., 2020). Regularization-based attacks can keep their performance as they directly adjust the discriminative features learned by the model. Our attack can also be categorized as a regularization based defense. Additional evaluations for label-only attacks can be found in "Appendix 3".

As successfully defending against vanilla attacks does not really guarantee privacy, we considered adaptive attacks when evaluating all defenses, which means the attackers have the knowledge of which defense algorithm and what hyper-parameters may have been applied to training the target model, so that they can plug these algorithms into their own shadow models. In most cases, there is no straightforward adaptive attack to our defense, since we essentially substituted the training dataset with a randomly generated dataset that

Table 2 Membership advantage of the shadow-model attack and regular accuracy of different target models, including the original classifier and all defended classifiers. Note that Membership advantage = 2 * Attack accuracy − 1. Detailed attack accuracy table for each class label can be found in the "Appendix 2"

Defense	Setting*	Membership advantage (%)	Regular accuracy (%)
(a) CIFAR-10			
None	–	32.91	85.60
Ours	Fine-tune (90/90)	14.01	83.97
	Fine-tune (50/90)	9.45	81.32
	Retrain	2.66	80.19
	Retrain†	12.86	80.19
Dropout	d = 0.2	33.63	87.26
	d = 0.4	32.60	88.59
	d = 0.6	31.42	90.47
	d = 0.8	27.04	91.07
L^2-Reg.	w = 0.001	20.45	84.33
	w = 0.01	8.78	71.94
DP-SGD	$\sigma = 0.001$	24.05	83.91
	$\sigma = 0.01$	23.06	83.79
Min–Max	–	23.94	79.81
Combined	Retrain†+DP(σ=0.001)	12.74	80.55
	Retrain†+DP(σ=0.01)	2.41	79.89
	Retrain† + Min–Max	2.55	80.14
(b) SVHN			
None	–	25.28	94.98
Ours	Fine-tune (90/90)	8.81	90.72
	Fine-tune (50/90)	6.57	88.42
	Retrain	4.55	87.98
	Retrain†	19.74	87.98
Dropout	d = 0.2	22.63	94.91
	d = 0.4	22.62	95.45
	d = 0.6	24.14	96.07
	d = 0.8	23.53	96.25
L^2-Reg.	w = 0.001	22.05	95.28
	w = 0.01	–	Not converged
DP-SGD	$\sigma = 1.0$	16.78	82.19
	$\sigma = 2.0$	14.59	70.05
Min–Max	–	14.45	94.75
Combined	Retrain† + DP($\sigma = 1.0$)	20.54	74.54
	Retrain† + DP($\sigma = 2.0$)	12.39	60.80
	Retrain† + Min–Max	12.84	86.51

†Evaluation under the worst-case scenario which is somewhat unlikely to occur

*d: dropout ratio, w: weight-decay rate, σ: standard derivation of the white Gaussian noise

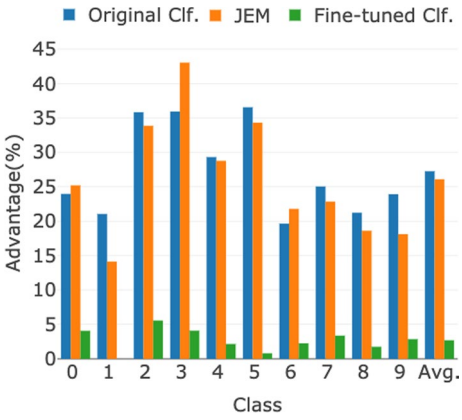


Fig. 7 Comparison of the attacker's advantage when launching the shadow-model attack on the original classifier, the JEM's discriminative model, and our fully fine-tuned classifier, respectively

would be different every time we do the sampling. However, in order to perform a more thorough and fair evaluation, we considered the following case: the attacker has direct access to the exact generated samples we used for fine-tuning or retraining. Although not quite possible in reality, it is the worst-case scenario for our defense, where all the training set information included in the final model, if any, would be revealed to the attacker.

Moreover, we observed that our defense is orthogonal with the other defenses. In other words, our defense can work simultaneously with other defenses to achieve possibly even better performance, so we also evaluated several combined methods to see whether our defense can boost other defenses' performance. Note that all the combined methods with our defense were evaluated in the worst-case scenario described in the previous paragraph.

The detailed results are also shown in Table 2. The most effective defenses under adaptive attack are large L^2 regularization or large DP. However, as shown in multiple previous works, large regularization can result in degradation in model accuracy. From the "Regular accuracy" column, we can see that large L^2 regularization or DP severely harmed the model's regular accuracy with a more than 20% decrease of accuracy. At the same time, even if our defense is under the worst-case adaptive attack, the defended model can still keep an acceptable accuracy while achieving high membership robustness. Interestingly, we found that the combined defenses often achieve much better robustness against adaptive attacks than individual defenses while having better regular accuracy than large L^2 regularization or DP. Although the trade-offs between the model accuracy and membership robustness are inevitable, a proper combination of defenses may provide us with smaller trade-offs to achieve better performance in both.

5 Discussion

5.1 Privacy analysis of JEM

As described in Sect. 3.2.3, we either retrained or fine-tuned the original classifier by using new data generated from a JEM, which is transferred from the original classifier, aiming to make it more privacy-preserving. A natural curiosity is if we can directly use the discriminative model of the JEM as a robust classifier against membership inference attacks since JEMs are more likely to have learned from the underlying data distribution and have better

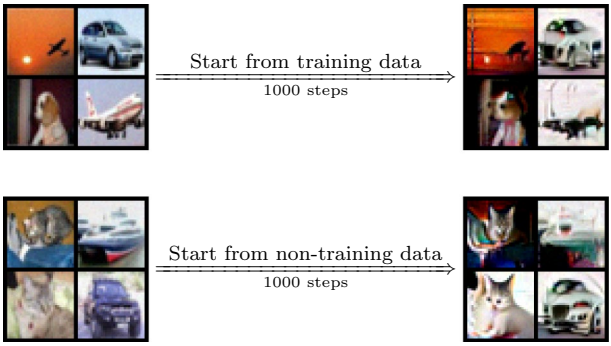


Fig. 8 SGLD results start from samples inside/outside the training dataset. If starting from training data, the SGLD generated samples are more likely to keep the original objects inside the images. On the other hand, if starting from non-training data, the samples may become other randomly generated objects of the same classes as the original ones

generalization ability by their nature. In fact, it has been shown that JEMs are more robust against adversarial examples (Grathwohl et al., 2019).

Therefore, we performed a direct evaluation by applying the shadow-model attack on JEM's discriminative part. Figure 7 compares the advantage of the attacker using the shadow-model attack on the JEM's discriminative model, the original classifier, and our fully fine-tuned classifier, respectively. Unfortunately, it shows that the JEM's discriminative model is no better than the original classifier in terms of robustness against membership inference attacks, but our fine-tuned model is much more robust than both of them.

Since JEM is a joint model with both discriminative and generative ability, we also try to provide some insights into JEM's generative model's robustness against membership inference attacks for completeness. In the black-box setting, the attacker has no information about the generative model. However, if we consider the white-box scenarios where the attacker can access the JEM weights, we found that JEM's generative model might also be vulnerable to membership inference attacks. Specifically, if we apply the training data as the initialization of the SGLD sampling using a well-trained JEM and proceed with many sampling steps, the resulting images will be more likely to be close to the original ones. On the other hand, if the sampling starts from other images from the same distribution but not inside the training data, the resulting images have a high probability of containing a different object of the same class. Figure 8 shows the phenomenon described above. This phenomenon might be leveraged to design membership inference attacks on JEM's generative model.

A question following our observations of JEM's privacy would be: can we build a robust JEM? In this paper, our framework is a multi-phase transfer learning process designed for pre-trained classifiers. It requires us to train two models: the JEM transferred from the original classifier and the fine-tuned privacy-preserving classifier. However, it is possible to apply an integrated training strategy that directly trains a privacy-preserving JEM by gradually replace the training data by generated samples during training.

Since studying the robustness of JEM against privacy attacks is outside the scope of this paper, we only provide some preliminary results and will leave the topic of improving the privacy robustness of JEMs as future work.

5.2 Other limitations and future work

Different threat models The threat model that we are using considers the grey-box attack. It allows the attacker to have knowledge of the target model's architecture. On the one hand,

we consider a stronger attacker to evaluate the worst-case vulnerability of the target model and our defenses, and we should even consider a white-box scenario where the attacker has full access to the models in the future. On the other hand, we may also want to know how is the attacker's realistic performance when less or even no information about the model is provided. We showed some empirical results in "Appendix 4" to provide some insights.

Moreover, our work only protects against membership inference attacks, and it would be interesting to apply the key ideas behind our framework to defending against other privacy attacks, such as model inversion attacks.

Training efficiency Though much simpler than training GANs, training JEMs is still slower than training classifiers. Moreover, JEM training faces many similar problems as training other generative or hybrid models, e.g., requiring careful selection of hyper-parameters. This paper introduced a new, efficient training strategy by transferring from pre-trained classifiers. Since optimizing the JEM training algorithm is out of the scope of this paper, we leave as future work ways to maximize the features and information that we can borrow from the pre-trained classifier.

6 Related work

Membership inference defenses Though it seems impossible to eliminate privacy attacks due to the intrinsic property of statistical inference, many methods were proposed to reduce the advantage that the attacker can obtain from such attacks. Current defenses can be roughly put into two categories. The first is regularization based defense, and the second is model output vector obfuscation.

Regularization based defenses are shown effective towards every kind of membership inference attacks. Using standard regularization such as L^2 regularization and Dropout as membership inference mitigation strategies are proposed together with the shadow model attack by Shokri et al. (2017). These standard regularization techniques also have been shown effective in enhancing privacy in other literature (Jain et al., 2015; Salem et al., 2018). Nasr et al. (2018) proposed an adversarial regularization technique, which tries to optimize a min–max game that simultaneously takes into account the model accuracy and membership robustness. Besides, Differential Privacy (DP) (Dwork 2008) has been frequently leveraged in defending against membership inference attacks, such as (Chaudhuri et al., 2011; Abadi et al., 2016; Wang et al., 2018). DP-based defenses try to theoretically ensure a differentially private training algorithm by inducing random noises, either in loss functions or in each optimization step. Once a model is differentially private, it can guarantee each data point's worst-case privacy risk.

The second category of defenses tries to increase the difficulty of training an attack model. The most straightforward strategies are output vector masking strategies, such as truncating the output vector to top-k classes, restricting the precision of the output that is revealed to the attacker, or increasing the entropy of the output towards uniform distribution. Besides simple output vector masking strategies, Jia et al. (2019) proposed another direction to mislead the attack models via adversarial attacks (Szegedy et al., 2013). By adding specially crafted adversarial perturbations to the model output, they could fool the attacker to infer given examples as wrong classes, thus increasing the attack difficulty. Unfortunately, these defenses have been shown ineffective against label-only attacks (Choo et al., 2020), which requires no information on the output vector.

Other defenses are from the perspective of keeping part of the information secret. Xiang et al. (2019) proposed to leverage the complex-valued neural networks, and to use a randomly generated rotation angle θ and complex counterpart b secrets, thus making it hard for the attacker to infer information from the transformed input $\exp(i\theta)[a + bi]$.

The aforementioned defenses either face restrictions to meet the theoretical conditions, require additional architectures or require large computational overhead. More importantly, the original training data are still directly exposed to the model training procedure. By contrast, we propose to defend against membership inference attack from a different perspective: generating samples from the same data distribution and hide the real training data. As we have shown, our method is straightforward and applies to all the current classifiers without requiring any additional architecture.

Other privacy attacks Besides membership inference attacks introduced in Sect. 2, there are also other privacy attacks. For example, attribute inference attacks aim to infer properties that should not be exposed to the public, such as background environment or gender information of a target in human face recognition systems. Ganju et al. (2018) tried to obtain properties by permutation invariant representations. Melis et al. (2019) also considered attribute inference in collaborative learning settings. Yeom et al. (2018) showed that membership inference attack is highly related to property inference attack in the way that they are both highly related to the overfitting problem. They also showed that property inference attack is harder to succeed than membership inference attacks.

Another category of attacks that also tries to extract information of data is the *model inversion attack*. Model inversion attack was first proposed by Fredrikson et al. (2015). It tries to inversely generate the original inputs of a target machine learning model given outputs or activations of the intermediate layers. Besides the attacks towards data, there also exist attacks that aim to steal the machine learning models (Tramèr et al., 2016; Wang & Gong, 2018).

Among all these attacks, the privacy vulnerabilities in machine learning models can cause severe privacy leaks when it comes to models used in areas where each single data point should be kept confidential. In this paper, we focus on defending against membership inference attacks.

7 Conclusion

We proposed a simple yet effective framework to protect pre-trained classifiers from membership inference attacks. We exploited the hidden generative power in a classifier by transferring it to a Joint Energy-based Model (JEM). We efficiently sampled data from the JEM to create a new dataset, which is independent of the original training set and is from the same distribution. Then, we used this new dataset to retrain or fine-tune the original classifier. We performed extensive empirical studies to evaluate different learning strategies and our framework's effectiveness against membership inference attacks. Our framework significantly reduced the attacker's membership advantage, from 32.91% on the original model to 2.66% on the retrained model on CIFAR-10, and from 25.28% on the original model to 4.55% on the retrained model on SVHN, while maintaining acceptable classification accuracy, which means it is an effective defense against membership inference attacks. Moreover, by comparing with other state-of-the-art defenses, we showed that our defense could maintain effective under the worst-case scenario and provide better accuracy-robustness trade-off when combined with other defenses.

Appendix 1: Derivation of Eq. (2)

We provide a detailed derivation of Eq. (2) which computes the gradient of $p_\theta(x)$. Given $p_\theta(x) = \frac{\exp(-E_\theta(x))}{Z(\theta)}$ and $Z(\theta) = \sum_{x' \in X} \exp(-E_\theta(x'))$, we have:

$$
\begin{aligned}
\nabla_\theta \log(p_\theta(x)) &= \nabla_\theta \log(\frac{\exp(-E_\theta(x))}{Z(\theta)}) \\
&= \nabla_\theta(\log(\exp(-E_\theta(x))) - \log(Z(\theta))) \\
&= -\nabla_\theta E_\theta(x) - \nabla_\theta \log(Z(\theta)) \\
&= -\nabla_\theta E_\theta(x) - \frac{\nabla_\theta Z(\theta)}{Z(\theta)} \\
&= -\nabla_\theta E_\theta(x) - \frac{\nabla_\theta \sum_{x' \in X} \exp(-E_\theta(x'))}{Z(\theta)} \\
&= -\nabla_\theta E_\theta(x) - \sum_{x' \in X} \frac{(\nabla_\theta(-E_\theta(x'))) \cdot \exp(-E_\theta(x'))}{Z(\theta)} \\
&= -\nabla_\theta E_\theta(x) + \sum_{x' \in X} [(\nabla_\theta E_\theta(x')) \cdot p_\theta(x')] \\
&= \mathbb{E}_{p_\theta(x')} \nabla_\theta E_\theta(x') - \nabla_\theta E_\theta(x)
\end{aligned}
$$

Appendix 2: Detailed attack results

For best performance, shadow-model attack (Shokri et al., 2017) trains a separate attack model for each class label of the target model. Table 3 shows the detailed attack accuracy for each class label for the CIFAR-10 and SVHN dataset. The membership advantage in Table 2 is computed based on these raw results.

Appendix 3: Evaluation of the label-only attack

Recently, Choo et al. (2020) proposed the label-only membership inference attack, which only requires output labels instead of output logits from the target model. The label-only attack has close performance compared with shadow-model attack. Surprisingly, though regularization based defenses are still effective, their attack can easily break output masking defenses such as MemGuard (Jia et al., 2019). We mainly evaluated the shadow-model attack in the paper. Since our defense directly functions during model training, it should be resistant to label-only attack if it can perform well under the shadow-model attack.

For completeness, we also evaluated our framework on CIFAR-10 and SVHN under the label-only attack to provide empirical evidence that our defense is still robust under the label-only attack. In this experiment, we use the same experiment settings in the label-only attack paper: a set of 5000 images, and the HopSkipJumpAttack (Chen et al., 2020) for the black-box adversarial attack. The results are listed in Table 4.

Table 3 Accuracy of the shadow-model attack for each target class and average attack accuracy of the original classifier and all defended models

Defense	Setting	Attack accuracy (%)										
		Target class										Average
		0	1	2	3	4	5	6	7	8	9	
(a) CIFAR-10												
None	–	65.74	59.89	71.40	75.11	68.39	70.81	65.13	64.20	61.64	62.25	66.46
Ours	Fine-tuned (90/90)	58.40	54.33	58.76	59.60	57.59	58.97	55.75	55.84	54.94	55.91	57.01
	Fine-tuned (50/90)	52.74	53.31	56.50	56.05	54.96	55.37	54.10	55.82	54.53	53.89	54.73
	Retrain	52.02	50.00	52.77	52.03	51.06	50.40	51.11	51.67	50.86	51.41	51.33
	Retrain†	56.80	54.86	57.45	56.83	56.11	58.18	55.73	56.48	55.61	56.22	56.43
Dropout	d = 0.2	65.65	59.44	70.51	75.91	68.70	71.40	66.03	66.03	62.36	62.12	66.82
	d = 0.4	66.28	59.87	71.94	75.61	67.46	68.98	65.60	64.22	61.41	61.64	66.30
	d = 0.6	65.24	58.79	70.04	74.89	66.94	70.40	63.43	64.18	61.25	61.93	65.71
	d = 0.8	63.47	57.65	64.90	69.11	65.51	64.86	62.45	63.20	61.41	62.61	63.52
L^2	w = 0.001	60.55	56.23	64.08	64.29	60.88	61.92	59.48	59.07	58.6	57.17	60.23
	w = 0.01	55.12	51.04	56.17	55.35	55.59	54.29	53.54	55.55	54.46	52.75	54.39
DP-SGD	$\sigma = 0.001$	60.39	55.56	67.10	69.20	64.15	65.71	61.33	60.40	58.12	58.29	62.03
	$\sigma = 0.01$	61.00	56.75	65.10	68.09	62.62	65.50	60.15	59.81	57.56	58.73	61.53
Min–Max	–	60.54	55.40	65.72	68.13	64.84	66.06	59.99	61.89	57.27	59.85	61.97
Combined	Retrain† + DP($\sigma = 0.001$)	56.56	54.03	57.74	59.09	56.19	56.35	56.10	56.56	55.75	55.34	56.37
	Retrain† + DP($\sigma = 0.01$)	52.05	51.06	51.86	50.86	50.66	51.31	50.84	51.35	51.18	50.85	51.20
	Retrain† + Min–Max	50.65	51.10	51.85	51.42	51.44	50.10	51.64	51.77	50.60	52.17	51.27
(b) SVHN												
None	–	65.29	59.75	64.60	63.61	59.90	62.45	64.10	59.44	65.17	64.71	62.90
Ours	Fine-tuned (90/90)	58.78	55.03	56.13	56.18	52.13	49.36	55.92	49.98	57.81	52.72	54.40
	Fine-tuned (50/90)	58.52	55.42	51.81	48.52	53.91	54.38	56.13	46.75	57.95	49.43	53.28
	Retrained	54.92	54.34	55.17	51.48	50.58	48.70	53.60	47.50	56.13	50.34	52.28
	Retrain†	65.13	58.37	62.72	59.09	59.35	54.53	60.33	56.77	62.9	59.51	59.87

Table 3 (continued)

Defense	Setting	Attack accuracy (%)										
		Target class										Average
		0	1	2	3	4	5	6	7	8	9	
Dropout	d = 0.2	63.15	58.58	62.06	61.67	57.24	63.20	63.35	59.77	63.31	60.80	61.32
	d = 0.4	63.73	60.36	61.70	63.33	57.84	59.76	58.90	60.83	64.64	62.02	61.31
	d = 0.6	61.22	59.06	64.05	63.72	61.91	60.68	64.43	61.87	64.83	58.94	62.10
	d = 0.8	64.75	60.06	63.14	63.56	61.39	61.17	60.35	59.53	61.10	62.63	61.77
L^2	w = 0.001	66.19	57.72	62.16	56.98	60.57	61.38	64.1	56.25	63.37	61.5	61.02
	w = 0.01	Not converged										
DP-SGD	$\sigma = 1.0$	60.42	57.41	60.54	58.61	57.0	55.68	57.95	55.45	62.43	58.4	58.39
	$\sigma = 2.0$	60.31	54.25	57.62	56.41	55.68	58.94	57.31	55.55	60.36	56.51	57.29
Min–Max	–	59.93	56.33	53.99	57.8	60.04	58.47	52.56	57.25	58.26	57.62	57.23
Combined	Retrain† + DP($\sigma = 1.0$)	63.85	57.13	61.97	60.2	57.55	59.79	61.87	57.03	62.08	61.21	60.27
	Retrain† + DP($\sigma = 2.0$)	55.78	54.16	58.12	56.42	56.43	53.71	55.41	53.56	61.71	56.66	56.19
	Retrain† + Min-Max	58.37	54.24	59.79	57.16	54.50	57.42	53.91	53.85	58.21	56.76	56.42

†Evaluation under the worst-case scenario which is somewhat unlikely to occur

*d: dropout ratio, w: weight-decay rate, σ: standard derivation of the white Gaussian noise

Table 4 Evaluation results of the label-only attack

Defenses	Attack performance (%)			
	CIFAR-10		SVHN	
	Accuracy	Precision	Accuracy	Precision
No defense	81.08	75.48	60.15	66.67
MemGuard	80.90	53.06	59.35	55.17
Ours (retrain)	51.28	50.14	50.55	53.81

Table 5 Results of shadow model attacks on CIFAR-10 of different shadow model architectures. The target model of the attack uses WRN(28×10)

Architecture	Attack accuracy (%)										
	Target class										Average
	0	1	2	3	4	5	6	7	8	9	
WRN(28×10)	65.74	59.89	71.40	75.11	68.39	70.81	65.13	64.20	61.64	62.25	66.46
WRN(34×10)	64.4	58.2	66.85	73.78	67.59	69.92	64.48	63.12	60.38	61.43	65.01
VGG19	50.27	49.66	49.86	53.5	49.91	50.3	49.89	50.28	49.84	49.64	50.31

From the table, we can find that our defense can still reduce the attack performance to nearly random guess (around 50%).

Appendix 4: Shadow models trained by different architectures

Current literature in membership inference attack usually considers the grey-box threat model proposed in Shokri et al. (2017), where the attacker has full knowledge of the target model's architecture. However, in this section, we will show some empirical results to provide insights on how using a different model architecture for the shadow model will affect the attack performance.

In the experiment, we consider three possible scenarios: (1) the attacker uses the same architecture as the target model, (2) the attacker uses a minor variation of the target model, and (3) the attacker has no knowledge of the target model and use an entirely different architecture for shadow models.

Specifically, the original model is a 28×10 WideResNet (Zagoruyko and Komodakis 2016) and is undefended. We applied a 34×10 WideResNet for the second scenario and a VGG19 (Simonyan and Zisserman 2014) network for the third scenario. The training parameters for the first and second scenarios are the same. All experiments are done on CIFAR-10.

The results are shown in Table 5. We can see that using a minor variation of the original model for the shadow model attack can achieve very close performance to using the real architecture. However, using an entirely different model architecture resulted in failed attacks, which indicates that hiding the original model architecture from the attacker is a feasible strategy to prevent membership inference attacks based on shadow models. Further study on this topic will be left as future work.

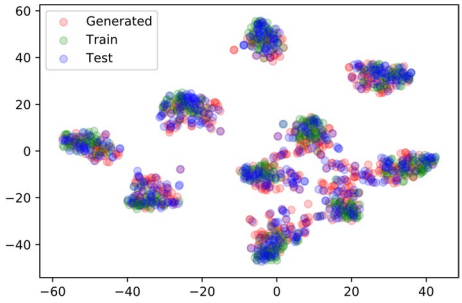


Fig. 9 The t-SNE dimensional reduction result of 500 random samples from the CIFAR-10 training dataset, 500 random samples from the CIFAR-10 testing dataset, and 500 random samples from the sampled dataset

Appendix 5: Quality of the generated dataset

Since the accuracy of our defended model is highly dependent on the quality of the generated dataset. The divergence between the real data distribution and the generated data distribution may cause a great loss in the model accuracy. In addition to Fig. 5, we provide more visualized evidence to show how well is the dataset sampled from the JEM in Fig. 9. Specifically, we applied t-SNE (Maaten and Hinton 2008) on 500 random samples from the CIFAR-10 training dataset, 500 random samples from the CIFAR-10 testing dataset, and 500 random samples from the generated dataset. We can observe that the generated dataset can very well catch the real data distribution.

Funding This material is based upon work supported by the National Science Foundation under Grant No. 1801751. This research was partially sponsored by the Combat Capabilities Development Command Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA).

Availability of data and material All datasets used in the paper is publicly available.

Code availability Code will be open-sourced: https://github.com/ChenJiyu/meminf-defense.git

References

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

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive science, 9*(1), 147–169.

Chaudhuri, K., Monteleoni, C., & Sarwate, A. D. (2011). Differentially private empirical risk minimization. *Journal of Machine Learning Research, 12,* 1069–1109.
Chen, J., Jordan, M. I., & Wainwright, M. J. (2020). Hopskipjumpattack: A query-efficient decision-based attack. In *2020 IEEE symposium on security and privacy (sp)* (pp. 1277–1294). IEEE
Choo, C. A. C., Tramer, F., Carlini, N., & Papernot, N. (2020). Label-only membership inference attacks. arXiv:200714321.
Du, Y., & Mordatch, I. (2019). Implicit generation and generalization in energy-based models. arXiv: 190308689.
Dwork, C. (2008). Differential privacy: A survey of results. In *International conference on theory and applications of models of computation* (pp. 1–19). Springer
Fredrikson, M., Jha, S., & Ristenpart, T. (2015). 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).
Ganju, K., Wang, Q., Yang, W., Gunter, C. A., & Borisov, N. (2018). Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security* (pp. 619–633).
Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).
Grathwohl, W., Wang, K. C., Jacobsen, J. H., Duvenaud, D., Norouzi, M., & Swersky, K. (2019). Your classifier is secretly an energy based model and you should treat it like one. arXiv:191203263.
Hayes, J., Melis, L., Danezis, G., & De Cristofaro, E. (2019). Logan: Membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies, 1,* 133–152.
Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems* (pp. 6626–6637).
Hilprecht, B., & Härterich, M. (2019). Monte carlo and reconstruction membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies, 4,* 232–249.
Hinton, G., Osindero, S., Welling, M., & Teh, Y. W. (2006). Unsupervised discovery of nonlinear structure using contrastive backpropagation. *Cognitive Science, 30*(4), 725–731.
Jain, P., Kulkarni, V., Thakurta, A., & Williams, O. (2015). To drop or not to drop: Robustness, consistency and differential privacy properties of dropout. arXiv:150302031.
Jia, J., Salem, A., Backes, M., Zhang, Y., & Gong, N. Z. (2019). Memguard: Defending against black-box membership inference attacks via adversarial examples. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, (pp. 259–274).
Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv:13126114.
Krizhevsky, A., & Hinton, G. et al. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto.
LeCun, Y., & Huang, F. J. (2005). Loss functions for discriminative training of energy-based models. In *AIStats, Citeseer* (Vol. 6, p 34).
Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research, 9,* 2579–2605.
Melis, L., Song ,C,. De Cristofaro, E., & Shmatikov, V. (2019). Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)* (pp. 691–706). IEEE
Nasr, M., Shokri, R., & Houmansadr, A. (2018). Machine learning with membership privacy using adversarial regularization. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security* (pp. 634–646).
Nasr, M., Shokri, R., & Houmansadr, A. (2019). Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)* (pp. 739–753). IEEE
Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011.
Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M., & Backes, M. (2018). Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. arXiv: 18060 1246.
Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. In *Advances in neural information processing systems* (pp. 2234–2242).
Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)* (pp. 3–18). IEEE

Shokri, R., Strobel, M., & Zick, Y. (2019). Privacy risks of explaining machine learning models. arXiv: 190700164.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv:14091556.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research, 15*(1), 1929–1958.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. arXiv:13126199.

Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., Ristenpart, T. (2016). Stealing machine learning models via prediction APIS. In *25th USENIX security symposium (USENIX security 16)* (pp. 601–618).

Wang, B., & Gong, N. Z. (2018). Stealing hyperparameters in machine learning. In *2018 IEEE symposium on security and privacy (SP)* (pp. 36–52). IEEE

Wang, J., Zhang, J., Bao, W., Zhu, X., Cao, B., & Yu, P. S. (2018). Not just privacy: Improving performance of private deep learning in mobile cloud. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, (pp 2407–2416).

Welling, M., & Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 681–688).

Xiang, L., Ma, H., Zhang, H., Zhang, Y., Ren, J., & Zhang, Q. (2019). Interpretable complex-valued neural networks for privacy protection. arXiv: 190109546.

Yeom, S., Giacomelli, I., Fredrikson, M., & Jha, S. (2018). Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)* (pp. 268–282). IEEE

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. arXiv:160507146.