Research Article
Towards Attack to MemGuard with Nonlocal-Means Method

Guangxu Xie and Qingqi Pei

The State Key Laboratory of Integrated Services Networks, Xidian University, Xi’an, China

Correspondence should be addressed to Qingqi Pei; qqpei@mail.xidian.edu.cn

Received 1 January 2022; Revised 13 March 2022; Accepted 28 March 2022; Published 18 April 2022

Academic Editor: AnMin Fu

Copyright © 2022 Guangxu Xie and Qingqi Pei. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

An adversarial example is the weakness of the machine learning (ML), and it can be utilized as the tool to defend against the inference attacks launched by ML classifiers. Jia et al. proposed MemGuard, which applied the idea of adversarial example to defend against membership inference attack. In a membership inference attack, the attacker attempts to infer whether a particular sample is in the training set of the target classifier, which may be a software or a service whose model parameters are unknown to the attacker. MemGuard does not tamper the training process of the target classifier, meanwhile achieving better tradeoff between the privacy and utility loss. However, many defenses of the adversarial example have been proposed, which decreases the effectiveness of the adversarial example. Inspired by the defenses of the adversarial example, we try to attack the MemGuard. As a result, we utilize the nonlocal-means method to do the attack by using the inherent relationship between neighbor entries to remove the added noise. Due to the low dimensionality of the confidence score vector, our attack avoids the high computation overhead of the nonlocal-means method. Besides, we use practical datasets to test our attack, and the experimental results demonstrate the effectiveness of our attack.

1. Introduction

Recently, we have witnessed the great development of the ML. On the one hand, ML can be used as the engine to improve the productivity in industry. On the other hand, it can be used as the tool to infer the sensitive information of the Internet users automatically. In this paper, we focus on the membership inference attack equipped with ML classifier. We consider ML model provider as black-box software or services, which return the confidence score vectors for query data samples from the Internet users. We consider a model provider deploys an ML classifier (called target classifier) as a black-box software or service, which returns a confidence score vector for a query data sample from a user. The confidence score vector is a probability distribution over the possible labels, and the label of the query data sample is predicted as the one that has the largest confidence score. Multiple studies have shown that such black-box ML classifier is vulnerable to membership inference attacks [1–4]. Specifically, an attacker trains a binary classifier, which takes a data sample’s confidence score vector as input and predicts whether the data sample is in the training dataset of the target classifier. We call the data sample member if the data sample is in the training dataset of the target classifier, and we call the data sample nonmember if the data sample is not in the training dataset of the target classifier. Membership inference attacks pose severe privacy and security threats to ML. In particular, in application scenarios where the training dataset is sensitive (e.g., biomedical records and location traces), successful membership inference leads to severe privacy violations. Beyond privacy, membership inference also damages the model provider’s intellectual property of the training dataset as collecting and labeling the training dataset may require lots of resources.

Therefore, defending against membership inference attack is an urgent research problem. Many defenses including [1, 3, 5, 6] have been proposed. A major reason why membership inference attacks succeed is that the target classifier is overfitted. The works [1, 3, 6] essentially regularize the training process of the target classifier to reduce overfitting and the gaps of the confidence score vectors between members and non-members of the training dataset. Since tampering the training process has no guarantees on the confidence score vectors, these defenses have no formal utility-loss guarantees on the
confident score vectors. Moreover, these defenses achieve suboptimal tradeoffs between the membership privacy of the training dataset and utility loss of the confidence score vectors. The work [5] proposed MemGuard (https://github.com/jjj1994/MemGuard), the first defense with formal utility-loss guarantees against membership inference attacks under the black-box setting. Instead of tampering the training process of the target classifier, MemGuard randomly adds noise to the confidence score vector predicted by the target classifier for any query data samples. Their empirical results showed that MemGuard can effectively defend against state-of-the-art black-box membership inference attacks [2, 3], and it achieves better privacy-utility tradeoffs than state-of-the-art defenses.

However, MemGuard leveraged the idea of the adversarial example, which is the weakness of the ML classifier. It has been proven that the adversarial example can be used as the tool to defend against the inference attack launched by the ML classifier. Many works including [7–13] have been proposed. Nevertheless, many defenses [14–18] against the adversarial example have been proposed. Now, we will ask whether the adversarial example based privacy protection is still effective, which is important for the privacy of the Internet user. This paper is trying to find the answer of this question. In this paper, we utilize the advanced defense of the adversarial example to attack the adversarial example based privacy protection scheme. We plan to apply the defense scheme proposed in [14] as our attack tool to launch the attack to the MemGuard. Xie et al. [14] proposed a defense of the adversarial example, which was proven to be generally effective. Their defense consists of a denoising block using the nonlocal-means method [19], ResNet classifier [20], and adversarial training using the adversarial examples generated by Projected Gradient Descent (PGD) [21]. We used their way to do our attack against the MemGuard, but we failed. The denoising block and ResNet model became inefficient when they were used in our attack. The adversarial training had a little effect when it was used solely and when it was combined with the denoising block. We find the main reasons why our attack failed. Firstly, different from the datasets used in [14], the confidence vector of the target classifier is input in our scheme, while the image datasets were used in their scheme. Besides, the datasets in our scheme are 1D, but their datasets are more than 1D. Secondly, the ResNet is suitable for the recognition of the image, but not for the 1D confidence vector. Thirdly, the 2D conventional layers do not have any meaning for the 1D vector, so the denoising block failed. As a feedback, we abandon the attack tool proposed in [14]. Finally, we succeeded by using the nonlocal-means method as our attack tool.

Nonlocal-means method is proposed in [19]; it was proposed to remove the noise in the digital image based on a nonlocal averaging of all pixels in the image. It was proven to be effective to remove the noise. But the nonlocal-means method has its own weakness, such as the high computation overhead. As an attacker, what we care about more is the time cost, and we will talk about it in the following sections.

In this paper, we use the nonlocal-means method proposed in [19] as our tool to attack the MemGuard. Different from [19], we regard the confidence 1D vector as our input instead of images. We will introduce our nonlocal-means method based attack tool in the following sections. Finally, we utilize the datasets used in MemGuard for experiment. As a result, the efficiency can be proved of our attack to the MemGuard. We show our attack scheme in the Figure 1.

In summary, the contributions are as follows:

(i) Firstly, we use nonlocal-means method as a tool to attack MemGuard, which was proven to be the state-of-the-art defense of membership inference attack.

(ii) Secondly, we prove that we can overcome the weakness of the nonlocal-means method, and our attack model can be used as a general tool to remove the noise added in the dataset when the dimension of the dataset is not large.

(iii) Finally, the experiment by using the practical datasets can prove the effectiveness of our attack. The code of our work can be found in GitHub (https://github.com/gxx1506215897/Towards-Attack-to-MemGuard).

The rest of the paper is organized as follows. In Section 2, we discuss the related work. In Section 3, we give the introduction of our attack trails, our nonlocal-means attack method used in this paper, and the different classifiers used in this paper. In Section 4, we present the experimental setups and experimental results of our attack. In Section 5, we give the conclusion of this work.

2. Related Work

The adversarial example is based on privacy protection schemes. The adversarial example is the weakness of the ML classifier, and it can be used as a tool to defend against the inference attack launched by the ML classifiers. Many works including [7–13] have been proposed to defend against inference attack using the idea of the adversarial example. Liu et al. [7] proposed the stealth algorithm of elaborating adversarial examples to resist the automatic detection system based on the Faster RCNN framework. Oh et al. [8] proposed a game-theoretical framework and studied the effectiveness of adversarial image perturbations for privacy protection. Jia and Attriguard [9] proposed a two-phase framework to defend against attribute inference attacks launched by a classifier. In the first-phase, the defender produced the noise using the adversarial example of the generation method. In the second-phase, the defender added the noise produced in the first-phase into the original data of the user randomly. They considered the influence of the more robust attack models with adversarial training, and they proposed to increase the noise budget when the attacker use the more robust attack model. Friedrich et al. [10] proposed a privacy-preserving shareable representation of medical texts for a deidentification classifier. They presented an adversarial learning based private representation of medical text. Liu et al. [11] proposed the schemes to protect the privacy of the Internet user, which used the idea of
adversarial example. Shao et al. [12] proposed the robust text CAPTCHAs scheme based on the adversarial example. Li and Lin [13] proposed to use adversarial perturbation for face deidentification.

The defenses of the membership inference attack: these kinds of works include [1, 3, 5, 6]. Reference [1] proposed $L_2$-Regularizer scheme; they considered the overfitting: that is, ML classifiers are more confident when facing data samples they are trained on (members) than others, and this is one major reason why membership inference is effective. Therefore, to defend against membership inference, [1] explored to reduce overfitting using $L_2$ regularization. Salem et al. [3] proposed using dropout and model stacking to mitigate membership inference attacks. Roughly speaking, dropout drops a neuron with a certain probability in each iteration of training a neural network. Model stacking is a classical ensemble method, which combines multiple weak classifiers’ result as a strong one. Specifically, the target classifier consists of three classifiers organized into a two-level tree structure. The first two classifiers on the bottom of the tree take the original data samples as input, while the third one’s input is the output of the first two classifiers. The three classifiers are trained using disjoint sets of data samples, which reduces the chance for the target classifier to remember any specific data sample, thus preventing overfitting. Nasr et al. [6] proposed a min-max game-theoretic method to train a target classifier. Specifically, the method formulates a min-max optimization problem that aims to minimize the target classifier’s prediction loss while maximizing the membership privacy. This formulation is equivalent to adding a new regularization term called adversarial regularization to the loss function of the target classifier. These works all essentially regularize the training process of the target classifier to reduce overfitting and the gaps of the confidence score vectors between members and nonmembers of the training dataset. They all tampered the training process of the target classifier, and they all do not have formal utility-loss guarantees on the confidence score vectors. Moreover, these defenses achieve suboptimal tradeoffs between the membership privacy of the training dataset and utility loss of the confidence score vectors.

Different from the above defense, Jia et al. [5] proposed MemGuard. MemGuard did not tamper the training process of the target classifier, and they also had the formal utility-loss guarantees on the confidence score vectors. They did experiments to prove that the MemGuard achieves the better privacy-utility tradeoffs than existing defenses against the membership inference attack. MemGuard adds the adversarial noise into the output of target classifier; however, the adversarial example can also be removed by some denoising scheme, but they did not talk about this. The defenses of adversarial example: Xie et al. [14] proposed new network architectures that increase adversarial robustness by performing feature denoising. Specifically, their networks contain blocks that denoise the features using nonlocal-means or other filters, and they obtain the robust classifier with adversarial training with the adversarial examples produced by the PGD method; they did their experiments on the ImageNet, and their method was ranked first in Competition on Adversarial Attacks and Defenses (CAAD) 2018. Except for [14], many works have been proposed to defense against adversarial examples; we choose some works including [15–18, 21] about this theme. Madry et al. [21] proposed to obtain the more robust classifier using the adversarial training with the adversarial example generated by the PGD method. They prove that their defense is generally effective and achieves the state of the art of the defense against the adversarial example. Tramèr et al. [15] introduced ensemble adversarial training, a technique that augments training data with perturbations transferred from other models. Their robust model won the first round of the NIPS 2017 competition on Defenses against Adversarial Attacks. Meng and Chen [18] proposed a framework, MegNet, to defend against adversarial examples. MegNet neither modifies the protected classifier, nor requires knowledge of the process for generating adversarial examples. MegNet includes one or more separate detector networks and a reformer network. The detector networks learn to differentiate between normal and adversarial examples by approximating the manifold of normal examples. Since they assume no specific process for generating adversarial examples, they generalize well. The reformer network moves adversarial examples towards the manifold of normal examples, which is effective for correctly classifying adversarial examples with small perturbation. They use the autoencoder as the detector, and they proved that their MegNet is effective. Wong et al. [17] tried to extend the previous work that is provably robust to norm-bounded adversarial perturbation in three directions. First, they presented a technique for extending these training procedures to much more general networks, with skip connections (such as ResNets) and general nonlinearities; the approach is fully modular and can be implemented automatically (analogous to automatic differentiation). Second, in the specific case of $l_\infty$ adversarial

\[ X \rightarrow \text{MemGuard} \rightarrow X' \]

**Figure 1:** Data sample is fed into the target classifier, and the classifier can be a software or a service, to output the confidence vector $x$. The parameter of the target classifier cannot be known except for the model providers. The defense scheme MemGuard is to add some noise into the $x$ to get the noised data $x'$. The attack model can not get some useful information from $x'$. To do our attack, we use the nonlocal-means method to remove the noise added into the $x$, we get the data $x^*$, and $x^*$ is close to $x$, so the attack model can get meaningful information from it.

\[ X \rightarrow \text{Target Classifier} \rightarrow \text{Data Sample} \]

\[ X \rightarrow \text{MemGuard} \rightarrow X' \]

\[ X'^* \rightarrow \text{Non-Local-Means} \rightarrow \text{Attack Model} \]
perturbations and networks with ReLU nonlinearities, they adopt a nonlinear random projections for training, which scales linearly in the number of hidden units (previous approaches scaled quadratically). Third, they showed how to further improve robust error through cascade models. Raghunathan et al. [16] proposed a method based on a semidefinite relaxation that outputs a certificate that, for a given network and test input, no attack can force the error to exceed a certain value. Since the certificate is differentiable, they jointly optimize it with the network parameters, providing an adaptive regularizer that encourages robustness against all attacks.

The attacks to the MemGuard: theoretically, all the defenses of the adversarial example can be utilized as the tools to attack the MemGuard, and we do some comparison between our method and others. For [15, 21], all of them use the adversarial training to improve the robustness of the classifier against the adversarial example. We can use the adversarial training to improve the inference ability of the attack model. What is more, we utilize the PGD to generate adversarial examples to do the adversarial training, but the inference accuracy of the attack model does not increase. If we can find the same noise generated by MemGuard, we may succeed. To get the same noise as generating by MemGuard, the attacker must know the parameter of the MemGuard. However, it is hard for the attacker to get any information about the parameter of the defender's model, which can be changed by the defender. For [18], they proposed a method to detect the adversarial example. We may get the higher detection accuracy by finding the appropriate parameters, but we failed. MegNet has many parameters to be set, and it has to be trained with the training data compared with nonlocal-means method, so our nonlocal-means method can be easier than MegNet.

3. Our Attack Scheme

3.1. Attack Trails. We start our attack work with the method proposed in [14] (https://github.com/facebookresearch/ImageNet-Adversarial-Training). Their method includes three parts, residual network, nonlocal-means denoising block, and adversarial training with the adversarial example generated by PGD [21] white-box attacks. The method in [14] has good performance compared with other defense against the adversarial example. We try to use the method in [14] to remove the adversarial noise.

Firstly, we change the structure of the attack classifier used in [5]; specifically, we combine the denoising block used in [14] with the fully connected neural network. Secondly, we use the PGD white-box attacks to generate the adversarial example. Thirdly, we train the modified attack classifier with the adversarial example and get the more robust attack model. Finally, we use the more robust attack model to infer the privacy of training data. The result of this attack is not good, we did not increase the inference accuracy of the attack model.

We tried to find the reason why our attack was not success. We first considered the residual network, and we thought if we used the residual network, we might get the better result. We try to change our 1D data to 2D data to use the ResNet-18 proposed in [20]. We thought that the inference accuracy may be increased because of the more advanced model structure; at the same time, we have the denoising block, and the result of our attack may be good. But we failed. Firstly, when we tried to use the 2D data to train the ResNet-18 to complete the 2-category task, the accuracy of the model is 0.5, which is the worst result. We failed in the first step. The reason why we failed is that there is no meaning to change the 1D data to 2D data to use the convolution layer, because the relationship among entries of our input data is different from the relationship among pixels on the images.

We did other experiments, of which we compared the result of the fully connected neural network with adversarial training with the result of the fully connected neural network with adversarial training and denoising block. We found that the first kind of setting gets the better result though the better result is not good enough to successfully attack MemGuard. This experiment proved that the denoising block does not work.

We did some trails, but we failed. Then, we try to use the nonlocal-means method in 1D data to remove the adversarial example; we succeeded. Our idea is inspired by the work [14], but our work is different from theirs. On the one hand, our datasets are different from the datasets used in [14]. Our data is 1D confidence scores, and the number of the entries of every data vector is not large; however, the data used in [14] is images, and the number of pixels of every image is large. On the other hand, we remove the noise from the noised data directly, while [14] removes the noise in the features of the input datasets.

3.2. Nonlocal-Means Method. The nonlocal-means method was proposed in [19] as the denoising algorithms for digital image. We use this method in the 1D data. The weakness of the nonlocal-means is its high computation overhead; we will discuss the problem of computation overhead when we launch our attack. In the following section, we firstly discuss the basic principle of nonlocal-means method, and then we will talk about the computation overhead when we deploy our attack use the nonlocal-means.

3.2.1. Basic Principle. Give a discrete noisy 1D vector \( v = \{v(i)|i \in I\} \), and the estimated value \( NL[v](i) \), for an entry \( i \), is computed as a weighted average of all the entries in the confidence vector,

\[
NL[v](i) = \sum_{j \in I} w(i, j)v(j),
\]

(1)

where the family of weights \( \{w(i, j)\}_i \) depends on the similarity between the entries \( i \) and \( j \) and satisfies the usual conditions \( 0 \leq w(i, j) \leq 1 \) and \( \sum w(i, j) = 1 \).

The similarity between two entries \( i \) and \( j \) depends on the similarity of the intensity vectors \( v(N_i) \) and \( v(N_j) \), where \( N_k \) denotes a line neighborhood of fixed size and centered at an entry \( k \). This similarity is measured as a decreasing
function of the weighted Euclidean distance, \( \|v(N_j) - v(N_i)\|_{\text{Euclidean}}^2 \), where \( \alpha \) is the standard deviation of the Gaussian kernel.

The entries with a similar neighborhood to \( v(N_j) \) have larger weights in the average. These weights are defined as

\[
w(i, j) = \frac{1}{Z(i)} e^{-\left(\|v(x_i) - v(x_j)\|_{\text{Euclidean}}^2 / h^2\right)},
\]

where \( Z(i) \) is the normalizing constant

\[
Z(i) = \sum_j e^{-\left(\|v(x_i) - v(x_j)\|_{\text{Euclidean}}^2 / h^2\right)},
\]

and the parameter \( h \) acts as a degree of filtering. It controls the decay of the exponential function and therefore the decay of the weights as a function of the Euclidean distances.

The nonlocal-means method compares not only the value of single entry of the input 1D vector, but the geometrical configuration in the whole neighborhood as well. This fact allows a more robust comparison. The reason why the nonlocal-mean method works when removing the noise in adversarial noised data is the noise change of the relationship among the neighborhoods of the specific entry. Our denoising method can be used to remove other types of noise more than adversarial noise, because we do not consider the noise generation scheme when we denoise. So, it can be used as a general method to remove the noise added into the data.

We list our nonlocal-method in Algorithm 1. The input of nonlocal-method is noised data produced by the defender, i.e., \( x' \), and the output of the nonlocal-method is denoised data, i.e., \( x^* \). Our goal is to let the denoised data \( x^* \) be close to the original data \( x \), i.e., \( x^* \approx x \).

In Algorithm 1, the input parameter \( M \) stands for the dimension of the vector \( x' \), the input parameter \( t \) stands for each range of search, the input parameter \( f \) stands for the numbers of the line neighborhoods of the specific entry of the vector, and the input parameter \( h \) stands as the degree of filtering. The kernel in Algorithm 1 can be generated by Algorithm 2.

**Algorithm 1**: Nonlocal-means attack method.

Input: \( x', M, t, f \) and \( h \).
Output: \( x^* \).

(1) for \( 0 \leq i \leq M - 1 \) do
(2) \( i_1 = i + f \)
(3) //obtain \( x'' \) by the symmetric padding of \( x' \);
(4) \( W_1 = x''(i_1 - f) \) to \( (i_1 + f + 1) \);
(5) \( w_{\text{max}} = 0 \);
(6) average = 0;
(7) sweight = 0;
(8) kernel = Makekernel \( (f) \);
(9) \( r_{\text{max}} = \max(i_1 - t, t + f) \);
(10) for \( r_{\text{min}} - 1 \leq r \leq r_{\text{max}} - 1 \) do
(11) if \( r - 1 = i_1 \) then
(12) continue;
(13) \( W_2 = x''(i_1 - f) \) to \( (i_1 + f + 1) \);
(14) \( d = \sum_{j=f}^{f} \text{kernel} \cdot (W_1 - W_2)^2 \);
(15) \( w = e^{d/h^2} \);
(16) \( w_{\text{max}} = w \);
(17) sweight = sweight + w;
(18) average = average + \( w \cdot x''_i \);
(19) \( \text{average} = \text{average} + w_{\text{max}} \cdot x''_i \);
(20) sweight = sweight + \( w_{\text{max}} \);
(21) if sweight \( > 0 \) then
(22) \( x_{i}' = \text{average} / \text{sweight} \);
(23) \( x_{i}' = x_{i}' \)

**Algorithm 2**: Makekernel method.

Input: \( f \).
Output: kernel.

(1) for \( 0 \leq i \leq 2f \) do
(2) \( \text{kernel} = 0 \)
(3) for \( 1 \leq d \leq f \) do
(4) \( \text{value} = 1/(2d + 1)^2 \)
(5) for \( -d \leq j \leq d \) do
(6) \( \text{kernel}_{i-j} = \text{kernel}_{i-j} + \text{value} \)
(7) \( \text{kernel} = \text{kernel} / f \)

3.2.2. Computation Overhead. The weakness of the nonlocal-means method is its high computation overhead. In our attack, if the dimension of the vector \( M \) is too large, the time cost of our attack will be very high, and our attack will fail. Fortunately, the defenses of the membership inference attack like MemGuard add noise to the confidence score vector of the target classifier. The target classifier will be introduced in detail later. The dimension of the confidence score vector is not always big, so the time cost in our attack can be accepted. We will measure the time delay of our attack in the experimental section, and we will prove that the time delay of our attack can be accepted.

3.3. ML Classifiers. In this section, we will introduce the ML classifiers used in our attack. The membership inference attack is to infer the information about whether the data samples are in the training datasets of the ML classifiers trained by the model provider. The weights and structure of the target classifier are not known by the attacker, but in MemGuard, the attacker is assumed to know about the structure of the ML classifier, and we follow the same assumption as MemGuard.

Target Classifier. The target classifier is trained by the model provider. It can be queried, and it can make predictions for the input data samples. In particular, we consider that the deployed classifier returns a confidence score vector for a query data sample.

Defense Classifier. The defense classifier uses the output of the target classifier as the training dataset, which can be used to decide whether a given data sample is in the training datasets of the target classifier or not. The defense classifier is trained by the defender, which is used to generate the adversarial noise to be added to the confidence score vector.
Shadow Target Classifier. The MemGuard assumes that the attacker knows the structure of the target classifier. The training dataset of the shadow target classifier is different from the target classifier, because the attack does not know the training dataset of the target classifier. If the attacker knows the training dataset of the target classifier, there is no meaning to infer the information.

Attack Classifier. The attacker uses the attack classifier to do the membership inference attack. In MemGuard, the attack classifier is used to evaluate the effectiveness of the defense of the membership inference attack. More generally, we use the attack classifiers different from the MemGuard to do the inference, and we will prove that the attack classifier is not the key reason why our attack is successful.

4. Experiments

In this section, we will elaborate the process of the attack to the MemGuard. We mainly focus on three parts, including the preparation of the attack, the setting of the attack, and the result of the attack.

4.1. Experimental Setup

4.1.1. Datasets. We use three datasets as MemGuard, which represent different application scenarios.

- Location: this dataset was used in [1], and it can be obtained from GitHub (https://github.com/privacytrustlab/datasets). The dataset has 5,010 data samples with 446 binary features, each of which represents whether a user visited a particular region or location type. The data samples are grouped into 30 clusters.

- Texas100: This dataset was obtained from [1], and it can be obtained from GitHub (https://github.com/privacytrustlab/datasets). The dataset has 67,330 data samples with 6,170 binary features. These features represent the external causes of injury (e.g., suicide, drug misuse), the diagnosis, the procedures the patient underwent, and some generic information (e.g., gender, age, and race). The data samples are grouped into 100 clusters, which represent 100 most frequent procedures.

- CH-MNIST. This dataset was obtained from Kaggle (https://www.kaggle.com/kmader/colorectal-histology-mnist). The dataset is used for classification of different tissue types on histology tile from patients with colorectal cancer. The dataset contains 5,000 images from 8 tissues. The size of each image is 64 $\times$ 64.

4.1.2. Models. We use the settings of the defense models as [5]. We use different settings of the attack models.

- Defense Models. For the Location and Texas100 datasets, we use a fully connected neural network with 4 hidden layers as the target classifier. The number of neurons for the four layers is 1024, 512, 256, and 128, respectively. We use the activation function ReLU for the neurons in the hidden layers. The activation function in the output layer is softmax. We adopt the cross-entropy loss function and use Stochastic Gradient Descent (SGD) to learn the model parameters. For CH-MNIST dataset, the target classifier is convolutional neural network, which includes three convolution layers and two fully connected layers. The activation function used in the hidden layers is ReLu, and the activation function used in the output layer is softmax. For more details, check [5]. Except for the target classifier, there must be a classifier to be used as the tool to produce the adversarial noise, that is, defense classifier. For three different datasets, we all utilize the fully connected neural networks, of which the hidden layers all have 512, 256, 128, and 64 neurons, and the output layers all have 1 neuron.

- Attack Models. In membership inference attack, the attacker first needs to know the output of the training dataset. We assume the attacker knows the structure of the target classifiers. The attacker can train the target classifier using the dataset he/she can obtain. We call the target classifier trained by the attacker as shadow target classifier. Except for the shadow target classifier, the attacker must train the attack classifier to decide whether the dataset is in the training dataset of the target classifier. We use the attack classifier as in [5]. For three kinds of datasets, the attack classifiers have the same hidden layers, which have 512, 256, and 128 neurons. For location, the number of input neurons is 30. For Texas100, the number of input neurons is 100. For CH-MNIST, the number of input neurons is 8.

4.1.3. Nonlocal-Means. For non-local-means method [19], there are three parameters, i.e., $t$, $f$, $h$. The notion $t$ controls the range of search. The notion $f$ controls the fixed numbers

---

### Table 1: The details of the datasets splitting.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Location</th>
<th>Texas100</th>
<th>CH-MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1$</td>
<td>0–999</td>
<td>0–999</td>
<td>0–999</td>
</tr>
<tr>
<td>$D_2$</td>
<td>1,000–1,999</td>
<td>10,000–19,999</td>
<td>1,000–1,999</td>
</tr>
<tr>
<td>$D_3$</td>
<td>2,000–2,999</td>
<td>20,000–29,999</td>
<td>2,000–2,999</td>
</tr>
<tr>
<td>$D_4$</td>
<td>3,000–3,499</td>
<td>30,000–39,999</td>
<td>3,000–3,499</td>
</tr>
<tr>
<td>$D_5$</td>
<td>4,000–4,499</td>
<td>40,000–49,999</td>
<td>4,000–4,499</td>
</tr>
</tbody>
</table>

### Table 2: Time cost of our attack.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Location</th>
<th>Texas100</th>
<th>CH-MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time cost (ms)</td>
<td>$8.00 \times 10^{-2}$</td>
<td>9.91</td>
<td>$1.35 \times 10^{-2}$</td>
</tr>
</tbody>
</table>
of the line neighborhood. $h$ acts as a degree of filtering. We use the inference of attack model as the indicator, and we make the structure of the attack model, the parameter of the attack model, and two parameters of the nonlocal-means method fixed; we choose the most suitable $t$, $f$, $h$ when the inference accuracy of the attack model gets the highest inference accuracy. We use the dataset Location, the attack classifier as MemGuard, and the same parameters of the model to choose the value of the above three parameters of nonlocal-means. We finally set $t = 9$, $f = 2$, $h = 3$. For CH-MNIST, we set the $t = 8$, because there are only 8 different classes in the output of the target classifiers. Our attack is simple; we use the nonlocal-means method to remove the noise in the noised dataset produced by the MemGuard. We only look the inference accuracy of the attack model to look the effectiveness of our attack. The weakness of the nonlocal-means method is its computation complexity, but in the membership inference attack, the dimension value of the datasets is small, and this weakness is avoided in the membership inference attack. We can launch the attack in a very small amount of time, and we will list the time cost of our attack in the following section.

4.1.4. Datasets Splitting. To make thing clear, we think it is necessary to talk the detail about the datasets splitting. We notice that the numbers of the datasets Location and CH-MNIST are approximately equal. We choose to use 5,000 samples when dealing with the datasets Location and CH-MNIST. We choose to use 50,000 samples when dealing with the dataset Texas100. We list our datasets splitting scheme in Table 1. In the table, the number is the index of the samples in the dataset, and "-" stands for one index to another index. We firstly shuffle datasets before we choose the index of every sample. The dataset $D_1$ is used for training the target classifiers, with the member training dataset for the defense classifiers and with the member evaluating dataset of the MemGuard. Specifically, the evaluating dataset of MemGuard will be fed into the defense scheme and become the noised dataset, and the member that stands for the dataset is in the training dataset of the target classifier. The dataset $D_2$ is used for testing the target classifiers, with the nonmember training dataset for the defense classifiers. The nonmember that stands for the dataset is not in the training dataset of the target classifier. The dataset $D_3$ is used for the nonmember evaluating dataset of the MemGuard. The datasets $D_2$ and $D_3$ are used for training and testing the shadow target classifiers for the attacker.

4.1.5. Other Parameters Setting. There are some other parameters that have to be set. For the target classifiers and the datasets Location with Texas100, the round of the model training is 200, the batch size is 64, the learning rate in the beginning is 0.01, and the learning rate is decayed by 0.1 in the 150th epoch for better convergence. For target classifiers and CH-MNIST, the round of the model training is 400, the batch size is 64, the learning rate is 0.01 in the beginning, and it is decayed by 0.1 in the 350th epoch. For defense classifiers, the round of the model training is 400, the batch size is 64, and the learning rate is 0.001. For shadow target classifiers, the round of the model training is 200, the batch size is 64, the learning rate is 0.01 in the beginning, and it is decayed by 0.1 in the 150th epoch. For attack classifiers, we choose to use the same setting, the round of model training is 400, the batch size is 64, the learning rate in the beginning is 0.01, and it is decayed by 0.1 in the 300th epoch.

4.2. Experimental Result. In this section, we will present our result of the experiment. We mainly focus on two kinds of indicators: one is the time cost of the nonlocal-method, and the other is the final inference accuracy of our attack classifiers. The reasons why we mainly focus on these two indicators are the high computation overhead; that is, the weakness of the nonlocal-means method and the higher inference accuracy of the attack classifier indicates the effectiveness of our attack.
4.2.1. Time Cost. We show the mean time delays for every sample for the three datasets when we use the nonlocal-means method to remove the noise. We list the time delays in Table 2.

4.2.2. Inference Accuracy. We use the nonlocal-means method to remove the noise in the noised data produced by the defender. We use three different attack models to do the final inference. We present our result of inference accuracies for the three datasets in Figure 2. In the figure, we can find that the inference accuracies of our attack schemes have big improvement compared to the evaluation of MemGuard, so it proves the effectiveness of our attack.

5. Conclusion
Adversarial example is the weakness of the ML, but it can be applied to defend against the inference attack equipped with ML. Many researchers have devoted themselves to applying it as the tool to protect the sensitive information of the Internet user from being inferred by the ML. But whether the adversarial example based protection schemes are secure is still an unanswered question. In this paper, we tried to use the nonlocal-means method, which was used to remove the noise in the image, to remove the noise added by the defense scheme of membership inference attack, i.e., MemGuard. The results of our experiment prove the effectiveness of our attack. Moreover, the results show that no matter the kinds of attack models we choose, we can always improve the inference accuracy, and it also proves that the nonlocal-method is effective. For traditional nonlocal-means, the high computation overhead is its weakness, and we measure the time of denoising, the time delay is little, and it proves that the time cost of our attack is little. Generally speaking, our attack work is successful; it proves that the adversarial example based privacy protection schemes are not always safe. In the future, we need to design more secure schemes to defend against the membership inference attack.

Data Availability
The code of this work can be found in https://github.com/gxx1506215897/Towards-Attack-to-MemGuard.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Acknowledgments
This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB2700600, in part by the National Natural Science Foundation of China under Grant 62132013, and in part by the Key Research and Development Programs of Shaanxi under Grants 2021ZDLGY06-03.

References


