# MEMBERSHIP INFERENCE ATTACK IN FACE OF DATA TRANSFORMATIONS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Membership inference attacks (MIAs) on machine learning models, which try to infer whether an example is in the training dataset of a target model, are widely studied in recent years as data privacy attracts increasing attention. One unignorable problem in the traditional MIA threat model is that it assumes the attacker can obtain exactly the same example as in the training dataset. In reality, however, the attacker is more likely to collect only a transformed version of the original example. For instance, the attacker may download a down-scaled image from a website, while the smaller image has the same content as the original image used for model training. Generally, after transformations that would not affect its semantics, a transformed training member should still be treated the same as the original one regarding privacy leakage. In this paper, we propose extending the concept of MIAs into more realistic scenarios by considering data transformations and derive two MIAs for transformed examples: one follows the efficient loss-thresholding ideas, and the other tries to approximately reverse the transformations. We demonstrated the effectiveness of our attacks by extensive evaluations on multiple common data transformations and comparison with other state-of-the-art attacks. Moreover, we also studied the coverage difference between our two attacks to show their limitations and advantages.

## 1 INTRODUCTION

Privacy issues in machine learning models have been attracting more concerns in recent years. Membership inference attacks (MIA) (Shokri et al., 2017) are one of the most studied ones, designed to infer whether an example owned by the attacker is from the training dataset. MIA can cause severe privacy leakage in privacy-critical domains, such as medical models trained on patients' medical records or facial recognition systems trained on a particular set of identities.

However, though empirical studies (Hu et al., 2021) show that undefended machine learning models are extremely vulnerable to MIAs, realistic MIAs are hardly reported in the industry. We notice an unignorable problem in the threat model of all current MIA research, which comes from the assumption that the attacker can always obtain the original member that appears in the training dataset. However, in the real world, examples collected by the attacker from different sources may have been transformed. For example, images downloaded from the Internet may have been re-scaled or compressed; stolen medical records might have missing features. More importantly, examples with small transformations that preserve their semantics would still leak private information. It is natural to ask: can adversaries infer the membership, even if they only have a transformed version of a training member? In this paper, we show that the answer is yes, and the current MIAs assumptions may underestimate attackers' capability and lead to a biased privacy risk assessment.

We aim to extend the original MIAs' targets space by considering semantic-preserving data transformations. Assuming the attacker can have partial knowledge about the transformation functions, we propose two MIAs to infer the membership of transformed examples. The first attack improves the traditional loss-thresholding attacks, which try to set a threshold by leveraging the different loss distribution of training and testing examples of the target model. The second attack is based on the assumption that the transformed training members would be closer to their $\epsilon$-robust areas (defined in Definition3.1) than those non-members, and it infers membership via comparing the distances required to reverse transformed examples back to their robust areas.

In our evaluations, we studied datasets in both Image and non-Image domains: CIFAR-10 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011), Purchase-100 (Kaggle, 2014). For images, we evaluated six commonly seen transformations: Gaussian noise, adversarial noise (Szegedy et al., 2013), JPEG compression (Wallace, 1992), scaling, photo filtering, and rotation. For Purchase-100, which has binary feature vectors, we considered the scenario that a part of the entries is randomly flipped. Our experiments show that our attacks can achieve higher accuracy than traditional attacks, indicating the risk of underestimating privacy leakage in the real world caused by ignoring data transformations. Additionally, we compared the coverage difference rate between our two attacks to show how different the two attacks can be in terms of individuals training members discovered. We show that while the loss-thresholding attack is efficient and straightforward, the second attack can successfully infer examples with large loss values. As a result, we hope our work can broaden the scope of current MIAs and help ML service providers better assess the actual privacy risks under a more realistic threat model.

## 2 Related work

In recent years, researchers thoroughly explored the domain of MIA, such as strategies for black-box attacks (Shokri et al., 2017; Yeom et al., 2018), label-only attacks (Choo et al., 2020; Chen et al., 2020; Li & Zhang, 2020; Rahimian et al., 2020), white-box attacks (Leino & Fredrikson, 2020; Sablayrolles et al., 2019), attacks on generative models (Hayes et al., 2019; Hilprecht et al., 2019; Liu et al., 2019), attacks on federated learning (Melis et al., 2019; Nasr et al., 2019); defense mechanisms from the perspective of privacy-preserving algorithms (Abadi et al., 2016; Chen et al., 2018; Xie et al., 2018), regularization (Shokri et al., 2017; Nasr et al., 2018; Chen et al., 2021), and output obfuscation (Shokri et al., 2017; Jia et al., 2019; Rahimian et al., 2020); and also the root cause of MIA (Yeom et al., 2018).

The label-only attack (Choo et al., 2020) shares a similar intuition with ours. Their core assumption is that training members should be more robust to small noises and lie farther from the decision boundary than non-members. In Section 3, we assume that it is easier to move training members towards some loss boundaries defined as $\epsilon$-robust areas. Both assumptions are based on the fact that the models more or less overfit training data.

Yu et al. (2021) discussed the MIAs on models that are trained with augmented data, where the original training data do not directly participate. Though their goal is different from ours, our problem settings share a similar condition: what the attacker obtains is not in model training. However, the core difference between our attack scenarios comes from the transformation knowledge of the attacker. While they assume that the attacker knows how the MIA target data produce the data in training, we assume the attacker knows the opposite, partially. We will make further discussions in Appendix B.

## 3 Methodology

### 3.1 Threat model

Below is the threat model we assume for the attackers. Note that besides the assumptions for data transformations, the other assumptions are commonly applied in other MIA research (Hu et al., 2021). We also limit the discussion to classification models in this paper.

**Data distribution**: The attacker can access and sample from the underlying data distribution $\mathcal{D}$, but does not know any specific members of the training set. The target example for inference might be a transformed version. However, the attacker can guess the family of transformations $\mathcal{G}$. The attacker always has the correct label for the target examples.

**Target model**: The attacker has full knowledge of the model architectures, training methods, and hyper-parameters. The attacker does not know the detailed model parameters but can query the model as a black box and obtain confidence scores for each class label.

**The attacker's goal**: The attacker wants to obtain an inference function $\mathcal{M}$ for the target model $f$, so that $\mathcal{M}_f(x') = \mathbf{1}(x \in \mathcal{D}_0)$, where $x \in \mathcal{D}_0$ is the original training example, $x' = g(x)$ for some $g \in \mathcal{G}$ is the transformed example, and $\mathcal{D}_0 \subset \mathcal{D}$ is the training set for $f$.

### 3.2 MIAS ON TRANSFORMED DATA

The foundation of all current MIAs is the statistics' difference between training and testing examples caused by overfitting. A simple yet effective attack proposed by Yeom et al. (2018) is to place a threshold and classify all examples whose losses are lower than the threshold as training examples. However, the loss distributions of training and testing examples change significantly after data transformations so that the threshold set for original training data can no longer optimally distinguish transformed data. We illustrated the problem in Figure 1 by showing the loss CDFs of the original CIFAR-10 dataset and the dataset after JPEG compression. Similarly, all current attacks face the same problem. In order to compute the statistics for MIAs, current attacks apply some functions that input the target model's output logits. For example, the loss function in Yeom et al. (2018), the entropy metric in Song & Mittal (2021) or the binary classifier in Shokri et al. (2017). These function's output can significantly change when facing transformed logits, especially when the model is sensitive to those transformations, resulting in a loss of attack accuracy. Evaluations of state-of-the-art MIAs on transformed examples can be found in Table 1 and Table 3. In this section, we propose two attacks to deal with the problem above: the first one improves the loss-thresholding attack, and the other is based on reverse transforming the target example.

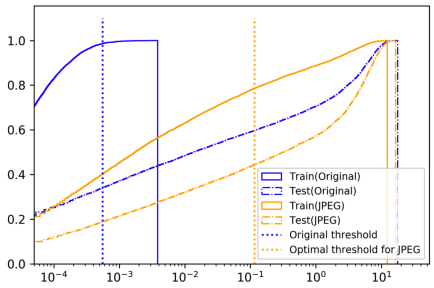



Figure 1: Loss CDFs for original and JPEG compressed CIFAR-10 images ($q = 10$).

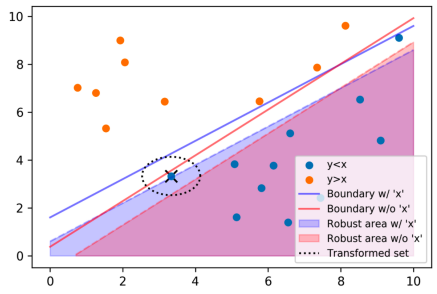



Figure 2: Intuition illustration for reverse transformation attacks.

Our first attack algorithm is in Algorithm 1. In order to accommodate the change in the loss distribution, it is straightforward to set a better threshold $\tau$. Instead of obtaining a threshold from the original training data, our attack transforms the training data and testing data based on the attacker's transformation knowledge and produces a loss threshold for transformed data distributions.

Thresholds in MIAs are commonly obtained from shadow models. Shadow models are proposed by Shokri et al. (2017), which are models trained with data sampled from the same data distribution as the target model. The attacker trains shadow models to mimic the behavior of the target model and transfer the attack threshold from the shadow models to the target model. Specifically, the threshold is selected to maximize the accuracy of the MIA on the shadow model since the attacker knows the exact data membership for the shadow model.

**Algorithm 1:** Loss-thresholding with transformations

**Input:** transformed example $x'$, true label $y$, target model $f$, loss function $\mathcal{L}$, threshold $\tau$
**Output:** membership $m \in \{\text{True}, \text{False}\}$
$l \leftarrow \mathcal{L}(f(x'), y)$;
**return** $l < \tau$;

Loss-thresholding is simple yet effective when a large generalization gap exists between the training and testing examples. By its nature, it would fail on training members with large losses. However, can we find transformed training members with large losses? The question leads to our second attack. We will start by the definition of an $\epsilon$-*robust area*, where $\epsilon$ serves as an important hyper-parameter to control the area range:

**Definition 3.1 ($\epsilon$-robust area)** *An $\epsilon$-robust area $R_\epsilon(x, f, \mathcal{G})$ of a given example $x$ and a target model $f$ and a family of transformation $\mathcal{G}$ is defined as:*

$$R_\epsilon(x, f, \mathcal{G}) = \{g(x, \delta) | \mathcal{L}(f(g(x, \delta)), y) < \epsilon, g \in \mathcal{G}, \delta \in \Delta\}$$

*where $g(\cdot, \delta)$ is some transformation function from a family of functions $\mathcal{G}$ parametrized by $\delta$ from the parameter space $\Delta$, $\mathcal{L}$ is the loss function w.r.t. label $y$, and $\epsilon \in \mathcal{R}^+$ is the radius of the area.*

Training members tend to be local minima of the model, especially for an overfitted model. So moving a transformed training example with a large loss towards its original version could decrease its loss more significantly than a testing example. An $\epsilon$-robust area essentially represents the neighborhood of the original example via some transformations with a loss lower than $\epsilon$. Our core assumption for the second attack is that a transformed training member is more likely to have smaller distances to the original one's $\epsilon$-robust areas. We illustrate the intuition with a toy model in Figure 2. We train a linear binary classifier to learn $f(x, y) = \mathbf{1}(x > y)$ with and without the data point in 'x' marker, respectively. The blue and red regions are their robust areas of linear transformations with the same $\epsilon$. Considering a linear transformation that adds $L^2$-bounded noise in the black circle, we can see that the 'x' and its transformed set are closer to the blue region as the model overfitted to the example.

In order to search for the minimum distance of a transformed example to its $\epsilon$-robust area, we define the *reverse transformation function* $g^{-1}(\cdot, \theta)$ in Definition 3.2. In our paper, we limit the search of the parameter $\theta$ of $\mathcal{G}'$ in a metric space $(\Theta, d)$ with a distance metric $d$, such that we can measure the amount of reverse transformation we applied to the example via $d(\theta)$.

**Definition 3.2 (Reverse transformation function)** *A reverse transformation function $g_\theta^{-1} = g^{-1}(\cdot, \theta)$ of a transformation $g_\delta = g(\cdot, \delta)$ is a function from a family of transformations $\mathcal{G}'$ parameterized by $\theta$ such that:*

$$\forall \delta \in \Delta, \exists \theta \in \Theta, g_\theta^{-1} \circ g_\delta = \mathcal{I}$$

*where $\Delta$ and $\Theta$ are parameter spaces for $\delta$ and $\theta$, respectively. $\mathcal{I}(x) = x$ is the identity function.*

Our attack algorithm is described in Algorithm 2, which tries to solve the optimization problem:

$$\begin{aligned} min \quad & d(\theta) \\ s.t. \quad & \mathcal{L}(f(g^{-1}(x', \theta)), y) < \epsilon \end{aligned}$$

Note that the reverse transformation function $g^{-1}$ is not necessarily the mathematical inverse of transformation $g$ since not all transformations are invertible. We abuse the notation of mathematical inverse for convenience. Generally, the reverse transformation can be any operation that can undo the transformation. For example, to undo transformations that modify pixel values such as Gaussian noises, we can simply select $g^{-1}(x', \theta) = x' + \theta$ as linear functions. The algorithm searches for the minimum distance (in terms of metric $d$) required to reversely transform a transformed data point into its original one's $\epsilon$-robust area. Though the reverse transformation function is only an approximation and does not guarantee the resulting example to strictly satisfy the definition of $\epsilon$-robust area (i.e., $g^{-1}(x', \theta_0) = g(x, \delta_0)$ for some $\theta_0 \in \Theta$ and $\delta_0 \in \Delta$), we will show its effectiveness in Section 4.

The update function $u$ in Algorithm 2 is a function that can update the reverse transformation function parameter $\theta$ in order to decrease the loss. Since the target model's parameters are unknown to the attacker according to the threat model, the attacker cannot directly update $\theta$ via gradient-based optimizations. For different datasets and transformations, we need to design different update functions. Detailed discussions on transformations and their corresponding update functions are in Section 4.

After the minimum distance is found, the attack strategy is the same as loss-thresholding: if the distance is smaller than a preset threshold $\tau$, the attacker will classify the corresponding example as a training member. The threshold is also selected via shadow models: the attacker finds the reverse transformation distances for the transformed training and testing data and then selects a threshold to maximize the inference accuracy on the shadow model.

# 4 Evaluation

## 4.1 Experiment settings

### 4.1.1 Datasets

In this section, we consider the following two datasets. Additional experiments on SVHN (Netzer et al., 2011) dataset can be found in Appendix E.2.

**Algorithm 2:** Reverse transformation

**Input:** transformed example $x'$, true label $y$, target model $f$, loss function $\mathcal{L}$, robust area radius $\epsilon$, reverse transformation function $g^{-1}$, update function $u$, attack threshold $\tau$, maximum iteration $n$, metric $d$
**Output:** membership $m \in \{\text{True}, \text{False}\}$
$\theta \leftarrow \mathbf{0}$;
$i \leftarrow 0$;
**while** $\mathcal{L}(f(g^{-1}(x', \theta)), y) >= \epsilon$ *and* $i < n$ **do**
  $\theta \leftarrow u(\theta)$;
  $i \leftarrow i + 1$;
**end**
**return** $d(\theta) < \tau \;\; \& \;\; i < n$; ▷ `False if number of iterations exceeded` $n$

**CIFAR-10** CIFAR-10 (Krizhevsky et al., 2009) is an image dataset consists of 10 classes of different natural objects. The dataset contains 60k of $32 \times 32$ images.

**Purchase-100**: The Purchase-100 (Kaggle, 2014) dataset is generated from Kaggle's "Acquire Valued Shoppers Challenge, Predict which shoppers will become repeat buyers" competition. The dataset consists of 60k customer records. Each record contains 600 binary features indicating each customer's purchase history, and all records are distributed in 100 classes.

In order to simulate the attacker's capability of accessing the real data distribution, we train the target model and shadow models by randomly selecting a training set of size 10000 and also a testing set of size 10000. Note that the training set for the target model and the shadow models are independent.

#### 4.1.2 TRANSFORMATIONS

Image datasets such as CIFAR-10 lie in a continuous space and also have spatial information among features. Since privacy usually resides in the semantics, we consider six kinds of semantic-preserving transformations for images:

**Gaussian noise**: Gaussian noise is one of the most common noises that would appear in images. In this case, the transformed example is $x' = x + \delta$, where $\delta \sim N(\mu, \sigma^2)$. We set $\mu = 0$ and assume the attacker knows the range of the standard deviation $\sigma \in (\sigma_{min}, \sigma_{max})$.

**Adversarial noise**: Adversarial noise (Szegedy et al., 2013) is defined as noises crafted to change the model output significantly. Compared with Gaussian noises, a tiny amount of adversarial noise can cause model misclassifications. We apply adversarial noises generated by PGD attacks (Madry et al., 2017) with a fixed $L^0$-norm of $0.05$. We assume that the attacker knows the attack algorithm, and we consider three different adversarial settings: 1-step PGD, 10-step PGD, and targeted-loss PGD (which runs until the loss exceeds the target loss).

**JPEG compression**: JPEG (Wallace, 1992) is a popular compression method for digital images. Since JPEG is a lossy compression method, the attacker cannot recover the image even knowing the algorithm and parameters. Our experiments assume that the attacker knows the compression quality parameter $q$ ($q \in [0, 100]$, higher $q$ means better image quality).

**Scaling**: Scaling is a common image processing operation when placing an image into a larger or smaller placeholder. We assume that the attacker does not know what interpolation strategy is applied. However, we also assume that the attacker knows the size of the original image (or the scaling factor $r$). In our experiment, we will consider both down-scaling ($r < 1$) and up-scaling ($r > 1$) scenarios.

**Filtering**: Photo filtering appears everywhere in social media applications when people share their photos. We assume the attacker can guess which filter the image applied to generate new images with the same filter. Our experiments evaluate attacks on several popular filters embedded in Instagram–a well-known photo-sharing social media. Detailed about the filters can be found in Appendix D.

**Rotation**: Rotation is a common spatial transformation that keep the image semantics. Since rotation is a fixed transformation once the degree of the rotation $\delta$ is given. So we only consider the scenario where the attacker does not know the exact rotation angle but can guess its range $\delta \in (\delta_{min}, \delta_{max})$.

Customer records in Purchase-100 lie in discrete space and have no explicit relationship among features. So noises and spatial transformations for images are unavailable and not worth considering for such datasets. Instead, we consider the following scenario for Purchase-100:

**Missing features**: Some (binary) features in the data records are corrupted and randomly flipped. We assume that the attacker knows the ratio $\gamma$ of flipped features in the original records, but the attacker does not know their exact positions. Note that for all records, the flipped positions may not be the same ones.

For the cases where there exists randomness in the transformations (e.g., Gaussian noise with a range of $\sigma$), we transform each example by a randomly selected parameter in the parameter space multiple times and compute the average loss or reverse transformation distance to select the threshold.

#### 4.1.3 TARGET MODELS

For CIFAR-10, we apply the VGG-19 (Simonyan & Zisserman, 2014) architecture. We apply an SGD optimizer with a learning rate of 1e-3 and weight decay rate of 5e-4 and trained for 100 epochs. The final model of CIFAR-10 has a training accuracy of 99% and testing accuracy of 69%.

For Purchase-100, we apply the same architecture used by Shokri et al. (2017), which is a fully connected neural network with one hidden layer of width 128, and $\tanh$ as the activation function. We apply an SGD optimizer with a learning rate of 1e-2 and weight decay rate of 1e-7 and trained for 200 epochs. The final model has a 98% training accuracy and 72% testing accuracy.

We did not apply any data augmentations in training for variable control. Also, all the models are undefended in this section. For evaluations on defended models, please refer to Appendix E.3.

#### 4.1.4 EVALUATION METRICS

To show the effectiveness of each attack, we report the accuracy, which is the average of the true positive rate and the true negative rate as we have the same amount of training and testing data. Additionally, to compare the set of training examples successfully discovered by our two attacks, we also compute a metric which we refer to as *coverage difference rate*, defined by:

$$\omega = \frac{|M_1 \cup M2 - M_1 \cap M2|}{|M|}$$

where $M_1$ and $M_2$ are training members discovered by the two attacks, and $M$ is the entire training set. The value represents the ratio of uniquely found training members in each attack.

### 4.2 EVALUATION ON CIFAR-10

Firstly, we evaluate how current MIAs performs against transformed examples. We evaluated five state-of-the-art attacks (Hu et al., 2021):(1) Classification correctness (CC) (Yeom et al., 2018), (2) Loss thresholding (LT) (Yeom et al., 2018), (3) Confidence score thresholding (ST) (Salem et al., 2018), (4) Entropy thresholding (ET) (Song & Mittal, 2021), and (5) NN-based attack (NN) (Shokri et al., 2017). Details of these attacks can be found in Appendix C.

Table 1: Accuracies of current attacks on CIFAR-10

| | Original | Gaussian($\sigma$) | | Adversarial | | JPEG($q$) | | Scaling($r$) | | Filtering | | Rotation($\delta$) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | [0,0.1] | [0,0.3] | 1-step | loss=10 | 50 | 1 | 0.5 | 10 | Clarendon | Moon | $[0°, 10°]$ | $[0°, 30°]$ |
| CC | 65.64% | 66.21% | 63.48% | 78.89% | 50.22% | 66.68% | 56.15% | 58.65% | 67.97% | 66.72% | 64.41% | 66.10% | 60.09% |
| LT | 80.47% | 77.32% | 64.78% | 62.63% | 50.02% | 74.13% | 51.98% | 53.46% | 70.32% | 72.78% | 60.56% | 69.51% | 59.03% |
| ST | 78.32% | 75.84% | 64.57% | 64.86% | 50.10% | 73.40% | 52.06% | 53.64% | 70.38% | 72.04% | 60.69% | 68.99% | 58.97% |
| ET | 77.58% | 76.44% | 65.50% | 68.49% | 50.03% | 75.59% | 52.93% | 54.94% | 73.11% | 73.81% | 63.20% | 71.32% | 60.51% |
| NN | 79.46% | 76.33% | 63.14% | 54.23% | 53.25% | 73.10% | 51.37% | 51.90% | 68.33% | 72.18% | 59.75% | 67.94% | 58.14% |

Table 1 shows the results of current attacks. When the transformations are small, we can see that all these attacks can still achieve a decent performance without considering data transformations due to the model's robustness to small transformations. However, when the transformation is large or adversarially designed, current attacks' accuracy would drop significantly. One observation worth

mentioning is that the classification correctness attack's accuracy gets better when the transformations are small, especially for adversarial noises. It directly results from the robustness difference of training and testing examples: a small adversarial noise can cause more misclassifications in the testing set than the training set, which is aligned with our intuition. The attack results of 10-step PGD (omitted) and fixed-loss PGD are nearly the same and very close to 50%, indicating that current attacks are ineffective in such cases.

For the evaluation of reverse transformation attacks, we apply a linear reverse transformation function $g^{-1}(x', \theta) = x' + \theta$ for all Gaussian noises, adversarial noises, JPEG compression, scaling, and filtering, since all these transformations are applied to pixel values. The update function $u$ is defined in Algorithm 3, which minimizes the loss $l$ w.r.t. the transformation parameter $\theta$ via gradient descent. Since the model is a black box assumed in the threat model, we apply zeroth-order optimization via gradient estimation. Note that we control the gradient via the gradient sign and a step size $\lambda$ at each update iteration. After reaching the $\epsilon$-robust area, we apply $L^2$-norm $d(\theta) = ||\theta||_2$ as the distance metric for membership inference.

**Algorithm 3:** Update function $u$ for transformations on pixel values

**Input:** transformed example $x'$, true label $y$, target model $f$, loss function $\mathcal{L}$, reverse transformation function $g^{-1}$ parameterized by the noise $\theta$, step size $\lambda$, small constant $\sigma_0$
**Output:** updated noise $\theta$
$\eta \leftarrow \text{sample}(N(0, \sigma_0^2))$;
$g \leftarrow \frac{\mathcal{L}(f(g^{-1}(x', \theta+\eta)), y) - \mathcal{L}(f(g^{-1}(x', \theta)), y)}{\eta}$; ▷ Gradient estimation
$\theta \leftarrow \theta - \lambda \cdot \text{sign}(g)$;
**return** $\theta$;

The reverse transformation function $g^{-1}$ in our experiment for rotations is also the rotation operation, with the parameter $\theta$ being the rotation degree required for rotating the example backward into the $\epsilon$-robust area. Since rotation is a fixed operation without randomness, we simply initialize $\theta = \delta_{min}$ as the lower bound of the transformation parameter $\delta$ and define the update function as a linear function that increase the backward rotation angle by a constant every time: $u(\theta, c) = \theta + c$. Then Algorithm 2 becomes looking for the first angle that makes the reversed example inside the robust area. In the extreme case, the algorithm would find the real rotation angle, which should be the local minimum of the target model and have the lowest loss among all rotation degrees.

Results of the loss-thresholding with transformation attack (LTT), the reverse transformation attack (RT), and their coverage difference rates ($\omega$) are shown in Table 2. We also collect the best accuracy among all five current attacks (CA) in Table 1 for comparison. As highlighted in the table, we can see that our attacks have higher accuracy than current attacks in all transformations. By evaluating three $\epsilon$-robust areas, we found that a proper $\epsilon$ can significantly help the reverse transformation. Sometimes a tiny $\epsilon$ can result in a low attack accuracy (e.g., 55.57% for loss=10 adversarial noises with $\epsilon$=0.001). The reason likely comes from the gradient estimation falling into some local minima since the transformation is too large. Furthermore, we observed a high coverage difference rate $\omega$ in many attacks, supporting our assumption that the reverse transformation attack can find transformed training members with large loss values. Note that some experiment results are omitted in Table 1 and Table 2 due to the page limit. Full tables can be found in Appendix E.1.

### 4.3 Evaluation on Purchase-100

Table 3 shows the performance of five state-of-the-art attacks on Purchase-100. Similar to the results of CIFAR-10, these attacks can still achieve a decent accuracy when facing a small missing ratio $\gamma$. However, the success rate decrease quickly as $\gamma$ increases.

Data in Purchase-100 have binary features. Therefore, the gradient estimation of the model loss with respect to each feature would be inaccurate. Instead, we design a greedy update function for the reverse transformation attack. The reverse transformation function $g^{-1}(x', \mathcal{S})$ is defined as flipping $x'$ by all the binary features whose indices are in the set $\mathcal{S}$. The update function is defined in Algorithm 4. Whenever we call the update function on a given record, we flip the single feature that can decrease loss values the most. We repeat this process until the data record is moved back to

Table 2: Accuracies and coverage difference rates $\omega$ of our attacks on CIFAR-10

(a) Gaussian noise, $\sigma$– range of standard deviation

| $\sigma$ | | $[0, 0.1]$ | | | $[0, 0.3]$ | |
|---|---|---|---|---|---|---|
| LTT | | 77.34% | | | 65.68% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 71.02% | 75.04% | **77.35%** | 65.58% | 64.14% | **65.91%** |
| $\omega$ | 8.37% | 6.96% | 1.35% | 7.63% | 2.52% | 8.51% |
| CA | | 77.32% | | | 65.50% | |

(b) Adversarial noise

| PGD | | 1-step | | | loss=10 | |
|---|---|---|---|---|---|---|
| LTT | | 78.87% | | | 56.17% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | **79.20%** | 79.19% | 77.92% | **64.45%** | 61.30% | 55.57% |
| $\omega$ | 2.49% | 5.62% | 10.33% | 42.47% | 42.90% | 47.45% |
| CA | | 78.89% | | | 53.25% | |

(c) JPEG compression, $q$–compression quality

| $q$ | | 50 | | | 1 | |
|---|---|---|---|---|---|---|
| LTT | | 75.43% | | | 57.00% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 71.10% | 74.68% | 75.31% | 57.31% | **57.51%** | 56.95% |
| $\omega$ | 9.02% | 5.62% | 2.17% | 10.97% | 18.88% | 23.11% |
| CA | | **75.59%** | | | 56.15% | |

(d) Scaling, $r$–scaling factor

| $r$ | | 0.5 | | | 10.0 | |
|---|---|---|---|---|---|---|
| LTT | | 58.87% | | | 73.78% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 61.09% | **61.46%** | 61.09% | 71.98% | 73.88% | **74.28%** |
| $\omega$ | 13.53% | 18.68% | 23.41% | 7.11% | 1.23% | 6.00% |
| CA | | 58.65% | | | 73.11% | |

(e) Filtering

| Filter | | Clarendon | | | Moon | |
|---|---|---|---|---|---|---|
| LTT | | **74.49%** | | | 65.49% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 71.00% | 74.03% | 74.15% | 65.51% | **65.56%** | 64.85% |
| $\omega$ | 7.59% | 3.70% | 4.47% | 0.80% | 5.21% | 10.58% |
| CA | | 73.81% | | | 64.41% | |

(f) Rotation, $\delta$–range of rotation degree

| $\delta$ | | $[0°, 10°]$ | | | $[0°, 30°]$ | |
|---|---|---|---|---|---|---|
| LTT | | 70.73% | | | 60.93% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 69.13% | 72.26% | **77.32%** | 61.52% | 66.44% | **75.57%** |
| $\omega$ | 8.47% | 9.84% | 14.2% | 13.69% | 18.72% | 39.47% |
| CA | | 71.32% | | | 60.51% | |

Table 3: Accuracies of current attacks on Purchase-100

| $\gamma$ | 0.00 | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 |
|---|---|---|---|---|---|---|---|---|---|
| CC | 63.33% | 61.72% | 57.54% | 53.74% | 51.61% | 50.54% | 50.18% | 50.17% | 50.13% |
| LT | 70.68% | 61.15% | 55.47% | 52.42% | 51.11% | 50.37% | 50.20% | 50.12% | 50.10% |
| ST | 59.36% | 56.51% | 53.78% | 52.26% | 51.30% | 50.64% | 50.32% | 50.39% | 50.21% |
| ET | 70.99% | 61.46% | 55.62% | 52.47% | 51.09% | 50.38% | 50.20% | 50.11% | 50.09% |
| NN | 65.65% | 60.58% | 56.01% | 52.93% | 51.37% | 50.63% | 50.28% | 50.16% | 50.09% |

the $\epsilon$-robust area. Finally, we define $d(\mathcal{S}) = |\mathcal{S}|$, which is the number of features flipped during the reverse transformation process, as the distance metric for membership inference.

**Algorithm 4:** Update function $u$ for missing features

**Input:** transformed example $x'$, true label $y$, target model $f$, loss function $\mathcal{L}$, reverse transformation function $g^{-1}$, set of flipped feature indices $\mathcal{S}$, number of features $n$
**Output:** updated set of flipped features $\mathcal{S}$

```
l ← L(f(g^{-1}(x', S)), y);
k ← −1;
for i ← 0 to n do
    if i ∉ S and L(f(g^{-1}(x', S + {i})), y) < l then
        l ← L(f(g^{-1}(x', S + {i})), y);
        k ← i;                                  ▷ Store the current best index
    end
end
S ← S + {k};                                    ▷ Update the flipped feature set
return S;
```

Table 4 shows the results of our attacks–LTT, RT, and their coverage difference rate–on Purchase-100. The best accuracy among current attacks is in row CA, and the best accuracy among all attacks is highlighted. In the experiment, we select a fixed $\epsilon = 0.5$ for the reverse transformation attack, but we vary the missing ratio $\gamma$ from 0.05 to 0.40 to see how it would affect our attacks. We can see that our attack can still successfully infer training members when the missing ratio is around 0.3, while all current attacks drop to random guesses when the missing ratio is 0.25. Similar to the results of CIFAR-10, there also exist large coverage difference rates between our two attacks.

Table 4: Accuracies and coverage difference rates $\omega$ of our attacks on Purchase-100

| $\gamma$ | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 |
|---|---|---|---|---|---|---|---|---|
| LTT | **64.09%** | **58.03%** | 55.40% | 52.79% | 51.96% | 50.70% | 50.11% | 49.83% |
| RT | 61.94% | 57.27% | **55.49%** | **53.70%** | **52.90%** | **52.02%** | **50.57%** | **50.41%** |
| $\omega$ | 21.08% | 24.27% | 25.06% | 24.44% | 24.85% | 22.93% | 20.64% | 19.09% |
| CA | 61.72% | 57.54% | 53.74% | 51.61% | 50.64% | 50.32% | 50.39% | 50.21% |

## 5 DISCUSSION

**Combination of attacks**: From the coverage difference rate shown in Section 4, we find that the members successfully inferred by the two attacks are not highly overlapped sometimes. Specifically, the reverse transformation attacks can find some transformed training examples with relatively high loss. The finding indicates that real-world attackers may combine different attack strategies to increase their attack success rate in the future.

**Knowledge of transformations**: One core assumption, and also limitation, in our threat model, is that the attacker needs to know partial information about the transformations applied to the data. However, is it possible to attack transformed examples without any information about the transformations? In the future, we hope to see that if the attacker can perform the attack by approximating the transformation without knowledge of parameters.

**Generalize to current attacks**: Our first attack is based on loss-thresholding. However, as we mentioned in Section 3.2, all current attacks share the same foundation so that the simple idea of our first attack can be generalized. For example, we can collect logits from transformed inputs and train an attack model similar to the NN-based attack. Since the main goal of our paper is to identify the transformation problem in the real world and show feasible solutions but not to evaluate all possible attacks, we will leave the discussions of those possibilities in the future.

## 6 CONCLUSION

In this paper, we identify a problem ignored by previous MIA works: transformed data, even if the target classifier misclassifies them, still are of the same value to attackers if they contain the same semantics information as the original ones. To solve the problem, we extend the assumptions in the traditional threat model by considering data transformations. We design two attacks outperform current state-of-the-art attacks on transformed data: loss-thresholding on transformed examples and reverse transformations. The former attack directly improves traditional loss-thresholding attacks by setting an optimal threshold based on the knowledge of transformations. The latter leverages the robustness difference between training and testing examples and tries to distinguish them by reversing the transformations. We show that our attacks can still be effective even when all the current attacks become random guesses through extensive evaluations. Additionally, we compare the two attacks via coverage difference rate and show that the reverse transformation attack can identify transformed training examples with high losses. As a result, we hope our work can broaden the scope of membership inference attacks and help improve the understanding of the privacy risk in reality.

## ETHICS STATEMENT

This paper discusses membership inference attacks on examples with transformations and shows that current threat models may cause underestimation of privacy risks in realistic scenarios. As we point out that privacy risks exist in all transformed examples semantically close to the model's actual training data, resulting in a broadened attack space, attackers may have a higher chance of inferring information from data with insufficient desensitization. On the other hand, our goal is to help the industry build a better privacy risk understanding. Specifically, machine learning service providers may reconsider how they share their data and how much information should be made public in the future.

## REPRODUCIBILITY STATEMENT

We provided detailed experiment settings in Section 4 and in Appendices. We submitted our source code inside the supplementary materials and will make the source code publicly available after paper publication.

## REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, 2016.

Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*, pp. 1277–1294. IEEE, 2020.

Jiyu Chen, Yiwen Guo, Qianjun Zheng, and Hao Chen. Protect privacy of deep classification networks by exploiting their generative power. *Machine Learning*, 110(4):651–674, 2021.

Qingrong Chen, Chong Xiang, Minhui Xue, Bo Li, Nikita Borisov, Dali Kaarfar, and Haojin Zhu. Differentially private data generative models. *arXiv preprint arXiv:1812.02274*, 2018.

Christopher A Choquette Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. *arXiv preprint arXiv:2007.14321*, 2020.

Facebook. Opacus: Train pytorch models with differential privacy. `https://opacus.ai/`, 2020.

Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. Logan: Membership inference attacks against generative models. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, volume 2019, pp. 133–152. De Gruyter, 2019.

Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte carlo and reconstruction membership inference attacks against generative models. *Proc. Priv. Enhancing Technol.*, 2019(4): 232–249, 2019.

Hongsheng Hu, Zoran Salcic, Gillian Dobbie, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *arXiv preprint arXiv:2103.07853*, 2021.

Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. 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, 2019.

Kaggle. Kaggle: Acquire valued shoppers challenge. `https://www.kaggle.com/c/acquire-valued-shoppers-challenge`, 2014.

Akiomi Kamakura. pilgram: A python library for instagram filters,. `https://github.com/akiomik/pilgram`, 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 1605–1622, 2020.

Zheng Li and Yang Zhang. Membership leakage in label-only exposures. *arXiv preprint arXiv:2007.15528*, 2020.

Kin Sum Liu, Chaowei Xiao, Bo Li, and Jie Gao. Performing co-membership attacks against deep generative models. In *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 459–467. IEEE, 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 691–706. IEEE, 2019.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 634–646, 2018.

Milad Nasr, Reza Shokri, and Amir Houmansadr. 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, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Shadi Rahimian, Tribhuvanesh Orekondy, and Mario Fritz. Sampling attacks: Amplification of membership inference attacks by repeated queries. *arXiv preprint arXiv:2009.00395*, 2020.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pp. 5558–5567. PMLR, 2019.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv: 1806.01246*, 2018.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Gregory K Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282. IEEE, 2018.

Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. How does data augmentation affect privacy in machine learning? 2021.

# Appendices

## A Experiment environment

Experiments are implemented with Python 3.7.5/Pytorch 1.6.0/CUDA 10.1 and run on two servers with (1) Intel(R) Xeon(R) Gold 6230 CPU @ 2.10GHz CPU and NVIDIA Quadro RTX 8000 GPU and (2) Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz CPU and NVIDIA TITAN RTX GPU, respectively. Codes will be open-sourced once the paper is published.

## B Comparison with MIAs on data augmented models

Yu et al. (2021) consider the scenario of data augmentation in model training. Specifically, the attacker has the original data, but only some augmented (transformed) versions are in the model training. On the other hand, we consider the scenario where the data that the attacker collects are transformed.

Suppose $x$ is the original data, and $x' = g(x)$ is transformed data with transformation $g$. If $g$ is invertible, and $x = h(x')$ where $h$ is the mathematical inverse of $g$, we can switch the role of $x$ and $x'$, that is, $x'$ is now regarded as the original example and $x$ is the transformed version of $x'$ via transformation $h$. In such cases, our scenario is equivalent to the scenario where the model is trained with single-time data augmentation. However, the assumption is not applicable for most transformations as they lose information, especially for those strictly lossy transformations (e.g., JPEG, down-scaling, feature-missing). In addition, for data with bounded values such as images, even invertible transformations can result in information loss due to value clipping. As a result, it is hard to apply Yu et al.'s attack in our scenario since they must know the exact form of the (inverse) transformation $h$.

On the contrary, both of our attacks can easily apply to any transformation. More importantly, the reverse transformation attack applies to all situations where the attacker gets slightly different examples from those in model training. In other words, the reverse transformation attack can also apply to data augmentation scenarios or even the combination of both scenarios discussed above. We will leave this topic in our future work.

## C Details of current MIAs

Here we provide detailed information for the five current attacks we evaluated in Section 4. Four of them are metric-based attacks, and the other one is an NN-based attack. Suppose $f$ is the target model, $x$ is the target example, $\tau$ is a real number represents the threshold, then the membership inference functions $\mathcal{M}$ for the five attacks are:

**Classification correctness (CC)** Classification correctness attack (Yeom et al., 2018) simply treats all correctly classified examples as training members.

$$\mathcal{M}_{\mathrm{CC}}(x, f) = \mathbf{1}(\mathrm{argmax} f(x) = y)$$

**Loss thresholding (LT)** Loss thresholding attack (Yeom et al., 2018) treats all examples with losses lower than a preset threshold as training members.

$$\mathcal{M}_{\mathrm{LT}}(x, f) = \mathbf{1}(\mathcal{L}(f(x), y) < \tau_{\mathrm{LT}})$$

where $\mathcal{L}$ is the loss function of the target model.

**Confidence score thresholding (ST)** Confidence score thresholding attack (Salem et al., 2018) treats all examples whose confidence score of the true label (the original attack selects the maximum confidence score) is larger than a preset threshold as training members.

$$\mathcal{M}_{\mathrm{ST}}(x, f) = \mathbf{1}(f(x)_y > \tau_{\mathrm{ST}})$$

**Entropy thresholding (ET)** Entropy thresholding attack (Song & Mittal, 2021) treats all examples whose output's entropy is smaller than a preset threshold as training members.

$$\mathcal{M}_{\mathrm{ET}}(x, f) = \mathbf{1}(\mathcal{E}(f(x), y) < \tau_{\mathrm{ET}})$$

where $\mathcal{E}(f, y) = -(1 - f_y) \cdot \log(f_y) - \sum_{i \neq y} f_i \cdot \log(1 - f_i)$ is the entropy metric.

**NN-based attack (NN)** NN-based attack (Shokri et al., 2017) collects an attack dataset containing output logits from shadow models and then trains a binary classifier for membership inference.

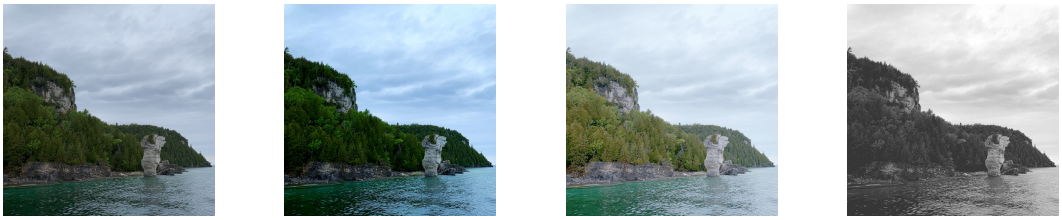
$$\mathcal{M}_{\mathrm{NN}}(x, f) = \mathrm{argmax}\mathcal{A}(f(x))$$

where $\mathcal{A}$ is the binary classifier.

## D Details of image filters

The three filters we applied in our experiments are Clarendon, Gingham, and Moon. They are the first three image filters implemented inside Instagram, a most popular photo-sharing platform with one billion active users (version 207.0 by the time of the current submission). The effects of the three filters are shown in Figure 3. The three filters are pretty different and represent various styles of image filtering. For example, Clarendon increases image saturation while Gingham reduces it; Clarendon and Gingham keep color while Moon goes grayscale. For experiment, we apply the implementation from *pilgram* library (Kamakura, 2019).

Our experiments assume that the attacker applies the filters as a black box, which means they do not need to know the detailed filter algorithm. Theoretically, for filters that only contain linear transformations, such as only increasing brightness, attackers can obtain the original image by applying the inverses of those linear transformations. However, in reality, image pixel values are strictly bounded. Even if many filters only contain linear transformations, most filtered images lose information during value clipping and cannot be reverted. On the other hand, we can apply our attacks to any black-box filters with a unified reverse transformation function.



(a) Original (b) Clarendon (c) Gingham (d) Moon

Figure 3: Image filters

## E Additional experimental results

### E.1 Full evaluation results on CIFAR-10

Table 5 and Table 6 are the full tables of Table 1 and Table 2, respectively.

Table 5: Accuracies of current attacks on CIFAR-10

| | Original | Gaussian($\sigma$) | | | Adversarial | | | JPEG($q$) | | | Scaling($r$) | | | Filtering | | | Rotation($\delta$) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | [0,0.1] | [0,0.2] | [0,0.3] | 1-step | 10-step | loss=10 | 50 | 10 | 1 | 0.5 | 1.5 | 10 | Clarendon | Gingham | Moon | $[0°, 10°]$ | $[0°, 20°]$ | $[0°, 30°]$ |
| CC | 65.64% | 66.21% | 65.53% | 63.48% | 78.89% | 50.22% | 50.22% | 66.68% | 66.07% | 56.15% | 58.65% | 68.02% | 67.97% | 66.72% | 63.60% | 64.41% | 66.10% | 63.11% | 60.09% |
| LT | 80.47% | 77.32% | 69.66% | 64.78% | 62.63% | 50.02% | 50.02% | 74.13% | 61.13% | 51.98% | 53.46% | 69.54% | 70.32% | 72.78% | 59.13% | 60.56% | 69.51% | 62.78% | 59.03% |
| ST | 78.32% | 75.84% | 69.19% | 64.57% | 64.86% | 50.10% | 50.10% | 73.40% | 61.42% | 52.06% | 53.64% | 69.61% | 70.38% | 72.04% | 59.29% | 60.69% | 68.99% | 62.63% | 58.97% |
| ET | 77.58% | 76.44% | 70.33% | 65.50% | 68.49% | 50.03% | 50.03% | 75.59% | 64.41% | 52.93% | 54.94% | 72.30% | 73.11% | 73.81% | 61.50% | 63.20% | 71.32% | 64.67% | 60.51% |
| NN | 79.46% | 76.33% | 68.87% | 63.14% | 54.23% | 48.71% | 53.25% | 73.10% | 59.48% | 51.37% | 51.90% | 67.56% | 68.33% | 72.18% | 57.56% | 59.75% | 67.94% | 61.63% | 58.14% |

Table 6: Accuracies and coverage difference rates $\omega$ of our attacks on CIFAR-10

(a) Gaussian noise, $\sigma$– range of standard deviation

| $\sigma$ | | $[0, 0.1]$ | | | $[0, 0.2]$ | | | $[0, 0.3]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 77.34% | | | 71.13% | | | 65.68% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 71.02% | 75.04% | 77.35% | 68.92% | 70.00% | 70.93% | 65.58% | 64.14% | 65.91% |
| $\omega$ | 8.37% | 6.96% | 1.35% | 9.27% | 4.14% | 3.81% | 7.63% | 2.52% | 8.51% |

(b) Adversarial noise

| PGD | | 1-step | | | 10-step | | | loss=10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 78.87% | | | 64.32% | | | 56.17% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 79.20% | 79.19% | 77.92% | 60.26% | 55.37% | 52.81% | 64.45% | 61.30% | 55.57% |
| $\omega$ | 2.49% | 5.62% | 10.33% | 38.28% | 42.76% | 44.85% | 42.47% | 42.90% | 47.45% |

(c) JPEG compression, $q$–compression quality

| $q$ | | 50 | | | 10 | | | 1 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 75.43% | | | 66.37% | | | 57.00% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 71.10% | 74.68% | 75.31% | 66.57% | 67.08% | 66.81% | 57.31% | 57.51% | 56.95% |
| $\omega$ | 9.02% | 5.62% | 2.17% | 1.66% | 7.47% | 12.46% | 10.97% | 18.88% | 23.11% |

(d) Scaling, $r$–scaling factor

| $r$ | | 0.5 | | | 1.5 | | | 10.0 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 58.87% | | | 73.27% | | | 73.78% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 61.09% | 61.46% | 61.09% | 71.81% | 73.40% | 73.84% | 71.98% | 73.88% | 74.28% |
| $\omega$ | 13.53% | 18.68% | 23.41% | 6.93% | 2.06% | 6.60% | 7.11% | 1.23% | 6.00% |

(e) Filtering

| Filter | | Clarendon | | | Gingham | | | Moon | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 74.49% | | | 64.05% | | | 65.49% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 71.00% | 74.03% | 74.15% | 64.11% | 64.00% | 63.47% | 65.51% | 65.56% | 64.85% |
| $\omega$ | 7.59% | 3.70% | 4.47% | 1.66% | 5.53% | 10.72% | 0.80% | 5.21% | 10.58% |

(f) Rotation, $\delta$–range of rotation degree

| $\delta$ | | $[0°, 10°]$ | | | $[0°, 20°]$ | | | $[0°, 30°]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 70.73% | | | 64.79% | | | 60.93% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 69.13% | 72.26% | 77.32% | 64.58% | 68.64% | 76.38% | 61.52% | 66.44% | 75.57% |
| $\omega$ | 8.47% | 9.84% | 14.2% | 6.26% | 13.73% | 29.68% | 13.69% | 18.72% | 39.47% |

### E.2 Evaluation on SVHN

SVHN (Netzer et al., 2011) is an image dataset for street house numbers. We use the cropped version of the original dataset that contains around 70k of $32 \times 32$ images of digits from 0 to 9.

For training SVHN models, we apply the same architecture and training settings as CIFAR-10 in Section 4. The final model of SVHN has a training accuracy of 99% and a testing accuracy of 89%. For MIAs, we also evaluated the six image transformations.

Table 7 and Table 8 show the results of current attacks and our attacks on SVHN with transformations. Similar to the results of CIFAR-10 and Purchase-100, we can see that our attacks can achieve better

accuracies with a properly selected $\epsilon$-robust area, and our two attacks discover different sets of transformed training examples.

Table 7: Accuracies of current attacks on SVHN

| | Original | Gaussian($\sigma$) | | | Adversarial | | | JPEG($q$) | | | Scaling($r$) | | | Filtering | | | Rotation($\delta$) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | [0,0.1] | [0,0.2] | [0,0.3] | 1-step | 10-step | loss=10 | 50 | 10 | 1 | 0.5 | 1.5 | 10 | Clarendon | Gingham | Moon | $[0°, 10°]$ | $[0°, 20°]$ | $[0°, 30°]$ |
| CC | 55.65% | 55.82% | 55.99% | 55.68% | 63.90% | 50.29% | 50.29% | 55.71% | 56.14% | 50.99% | 56.40% | 55.75% | 55.72% | 56.08% | 55.58% | 55.72% | 55.84% | 54.43% | 52.55% |
| LT | 65.21% | 63.24% | 60.02% | 58.38% | 52.96% | 49.99% | 49.99% | 63.39% | 56.27% | 50.47% | 59.39% | 64.11% | 64.23% | 61.77% | 56.85% | 59.74% | 59.30% | 55.10% | 53.05% |
| ST | 64.54% | 62.90% | 59.97% | 58.39% | 53.26% | 49.99% | 49.99% | 62.94% | 56.39% | 50.49% | 59.24% | 63.70% | 63.80% | 61.44% | 57.17% | 59.59% | 59.07% | 55.10% | 53.10% |
| ET | 65.39% | 63.43% | 60.14% | 58.44% | 53.01% | 49.99% | 49.99% | 63.53% | 56.33% | 50.47% | 59.47% | 64.25% | 64.38% | 61.90% | 56.87% | 59.86% | 59.37% | 55.13% | 53.12% |
| NN | 65.89% | 64.37% | 60.45% | 58.11% | 50.46% | 50.84% | 50.59% | 64.27% | 56.25% | 50.41% | 59.48% | 65.32% | 65.13% | 62.76% | 55.50% | 60.10% | 59.97% | 56.03% | 53.73% |

Table 8: Accuracies and coverage difference rates $\omega$ of our attacks on SVHN

(a) Gaussian noise, $\sigma$– range of standard deviation

| $\sigma$ | | $[0, 0.1]$ | | | $[0, 0.2]$ | | | $[0, 0.3]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 63.49% | | | 61.65% | | | 60.09% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 58.38% | 61.50% | 65.06% | 58.29% | 60.49% | 62.45% | 57.47% | 59.25% | 60.03% |
| $\omega$ | 21.33% | 21.35% | 19.31% | 22.93% | 19.67% | 14.82% | 20.52% | 15.31% | 7.32% |

(b) Adversarial noise

| PGD | | 1-step | | | 10-step | | | loss=10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 61.39% | | | 52.03% | | | 50.72% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 63.33% | 61.52% | 62.10% | 56.02% | 53.18% | 51.35% | 56.76% | 56.98% | 55.51% |
| $\omega$ | 11.86% | 1.29% | 8.15% | 25.55% | 32.16% | 39.61% | 40.71% | 43.60% | 45.09% |

(c) JPEG compression, $q$–compression quality

| $q$ | | 50 | | | 10 | | | 1 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 63.55% | | | 57.47% | | | 50.92% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 58.28% | 61.44% | 65.22% | 57.69% | 58.74% | 58.67% | 51.62% | 52.21% | 52.68% |
| $\omega$ | 21.52% | 21.42% | 19.89% | 27.25% | 21.69% | 11.78% | 6.09% | 9.94% | 13.56% |

(d) Scaling, $r$–scaling factor

| $\delta$ | | 0.5 | | | 1.5 | | | 10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 59.81% | | | 64.00% | | | 64.10% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 58.85% | 61.09% | 62.03% | 58.40% | 61.78% | 66.16% | 58.38% | 61.76% | 66.21% |
| $\omega$ | 28.61% | 25.91% | 18.50% | 21.87% | 21.86% | 21.38% | 21.68% | 21.67% | 21.29% |

(e) Filtering

| $\delta$ | | Clarendon | | | Gingham | | | Moon | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 63.04% | | | 57.45% | | | 60.16% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 58.79% | 62.13% | 65.00% | 57.25% | 58.38% | 58.46% | 58.17% | 60.83% | 62.55% |
| $\omega$ | 23.65% | 22.96% | 18.78% | 29.72% | 24.30% | 14.65% | 28.35% | 26.68% | 20.49% |

(f) Rotation, $\delta$–range of rotation degree

| $\delta$ | | $[0°, 10°]$ | | | $[0°, 20°]$ | | | $[0°, 30°]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 59.77% | | | 56.34% | | | 53.74% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 57.69% | 59.58% | 61.11% | 55.19% | 56.07% | 56.53% | 52.91% | 53.51% | 54.87% |
| $\omega$ | 26.24% | 23.06% | 16.88% | 23.33% | 15.91% | 5.67% | 17.82% | 11.84% | 6.29% |

### E.3 EVALUATION ON DEFENDED MODELS

We mainly evaluate our attacks on undefended models in Section 4. Here we provide evaluation results of CIFAR-10 models trained with two popular defense strategies: (1) large weight decay rate (Shokri et al., 2017), and (2) deferentially-private (DP) training (Abadi et al., 2016). We assume that the attacker knows the defense algorithms and the parameters so that the attacker can also train shadow models with the same defenses. Note that we only evaluate two defenses as our goal is not to evaluate the performance of state-of-the-art defenses but to show the effectiveness of current defenses against all MIAs.

For the first defense, we set the weight decay rate to 0.1 (target model trained with 5e-4) and trained the model until convergence. For DP, we applied the implementation from *Opacus* (Facebook, 2020) with an RMSprop optimizer for better convergence, a gradient clipping value of 1.2, a noise multiplier of 2.3, and trained for 100 epochs, which achieves (5, 1e-5)-differential privacy. All the other training settings are the same as in Section 4.1 in order to control variates. The final models trained with a large weight decay rate have an average of 85% training accuracy and 64% testing accuracy, and the final models trained with DP have an average of 40% training accuracy and 39% testing accuracy.

Table 9, Table 10 and Table 11 show the attack results of current attacks and our attacks on the defended models. We can see that both defenses are effective against all MIAs, including ours. DP with a small privacy budget provides the best privacy guarantee, reducing all attacks to random guesses, with a trade-off between utility as the model accuracy is considerably lower than undefended ones.

While loss-thresholding with transformations is still working better than the current attacks, we notice that the performance of reverse transformation attack is ineffective under the defenses when the $\epsilon$ settings are the same with attacking undefended models in Section 4. This is due to the defenses increasing the average loss so that both training and testing examples are far away from the robust area of a small $\epsilon$. However, an adaptive attacker can always select a better $\epsilon$ for the reverse transformation attack. In Figure 4, we compare reverse transformation attacks with different $\epsilon$-robust areas on the model with weight decay defense. We can see that a proper selection of $\epsilon = 0.5$ can still lead to a better attack accuracy than the random guessing baseline and the loss-thresholding attacks. How to optimally select $\epsilon$ is an interesting topic and is left as our future work.

Table 9: Accuracies of current attacks on defended models

(a) Weight decay

| | Original | Gaussian($\sigma$) | | | Adversarial | | | JPEG($q$) | | | Scaling($r$) | | | Filtering | | | Rotation($\delta$) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | [0,0.1] | [0,0.2] | [0,0.3] | 1-step | 10-step | loss=10 | 50 | 10 | 1 | 0.5 | 1.5 | 10 | Clarendon | Gingham | Moon | $[0°, 10°]$ | $[0°, 20°]$ | $[0°, 30°]$ |
| CC | 60.98% | 60.50% | 58.59% | 57.03% | 58.54% | 50.13% | 50.13% | 58.60% | 54.54% | 51.17% | 52.55% | 59.16% | 59.35% | 60.67% | 59.54% | 59.73% | 58.63% | 56.16% | 54.60% |
| LT | 61.33% | 60.53% | 58.19% | 56.34% | 56.01% | 50.11% | 50.11% | 57.94% | 53.52% | 50.78% | 51.86% | 58.61% | 58.90% | 61.11% | 59.31% | 59.26% | 58.32% | 55.81% | 54.23% |
| ST | 61.35% | 60.55% | 58.20% | 56.34% | 56.06% | 50.12% | 50.12% | 57.92% | 53.52% | 50.78% | 51.90% | 58.63% | 58.90% | 61.02% | 59.30% | 59.27% | 58.32% | 55.78% | 54.23% |
| ET | 61.33% | 60.55% | 58.19% | 56.32% | 56.02% | 50.11% | 50.11% | 57.95% | 53.54% | 50.79% | 51.86% | 58.63% | 58.89% | 61.11% | 59.33% | 59.24% | 58.33% | 55.81% | 54.23% |
| NN | 57.81% | 57.21% | 55.41% | 53.53% | 46.96% | 49.86% | 49.89% | 54.76% | 51.14% | 49.88% | 50.32% | 54.83% | 55.26% | 57.26% | 55.71% | 55.73% | 55.09% | 53.23% | 52.29% |

(b) DP

| | Original | Gaussian($\sigma$) | | | Adversarial | | | JPEG($q$) | | | Scaling($r$) | | | Filtering | | | Rotation($\delta$) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | [0,0.1] | [0,0.2] | [0,0.3] | 1-step | 10-step | loss=10 | 50 | 10 | 1 | 0.5 | 1.5 | 10 | Clarendon | Gingham | Moon | $[0°, 10°]$ | $[0°, 20°]$ | $[0°, 30°]$ |
| CC | 50.67% | 50.70% | 50.62% | 50.48% | 50.15% | 50.00% | 50.00% | 50.63% | 50.50% | 50.24% | 50.34% | 50.59% | 50.58% | 50.70% | 50.52% | 50.62% | 50.63% | 50.70% | 50.51% |
| LT | 50.79% | 50.72% | 50.79% | 50.53% | 50.18% | 50.01% | 50.01% | 50.72% | 50.46% | 50.30% | 50.27% | 50.70% | 50.74% | 50.88% | 50.68% | 50.68% | 50.70% | 50.66% | 50.53% |
| ST | 50.55% | 50.54% | 50.45% | 50.32% | 50.17% | 50.04% | 50.04% | 50.31% | 50.26% | 50.08% | 50.25% | 50.61% | 50.65% | 50.59% | 50.43% | 50.50% | 50.46% | 50.28% | 50.19% |
| ET | 50.86% | 50.73% | 50.82% | 50.53% | 50.23% | 50.00% | 50.00% | 50.72% | 50.49% | 50.28% | 50.35% | 50.79% | 50.81% | 50.94% | 50.69% | 50.69% | 50.74% | 50.68% | 50.57% |
| NN | 49.82% | 49.93% | 50.10% | 50.14% | 49.63% | 50.07% | 50.17% | 49.90% | 50.05% | 50.18% | 49.97% | 50.23% | 50.09% | 49.92% | 50.01% | 50.08% | 50.13% | 50.14% | 50.18% |

Table 10: Accuracies and coverage difference rates $\omega$ of our attacks on CIFAR-10 models trained with large weight decay rate

(a) Gaussian noise, $\sigma$– range of standard deviation

| $\sigma$ | | $[0, 0.1]$ | | | $[0, 0.2]$ | | | $[0, 0.3]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 60.57% | | | 58.50% | | | 56.88% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.08% | 50.07% | 50.09% | 49.91% | 50.02% | 50.08% | 50.27% | 50.06% | 50.07% |
| $\omega$ | 49.40% | 49.66% | 49.83% | 48.86% | 49.82% | 50.28% | 49.33% | 50.43% | 50.12% |

(b) Adversarial noise

| PGD | | 1-step | | | 10-step | | | loss=10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 60.33% | | | 50.75% | | | 50.75% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.06% | 49.82% | 50.04% | 50.05% | 50.06% | 50.10% | 50.04% | 49.77% | 49.74% |
| $\omega$ | 48.98% | 50.19% | 50.26% | 50.04% | 49.98% | 50.21% | 50.30% | 50.34% | 49.82% |

(c) JPEG compression, $q$–compression quality

| $q$ | | 50 | | | 10 | | | 1 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 58.50% | | | 54.65% | | | 51.39% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.26% | 50.27% | 49.74% | 49.90% | 50.17% | 49.95% | 50.27% | 50.11% | 50.07% |
| $\omega$ | 48.94% | 50.48% | 49.82% | 48.97% | 49.64% | 49.72% | 49.81% | 50.17% | 50.10% |

(d) Scaling, $r$–scaling factor

| $\delta$ | | 0.5 | | | 1.5 | | | 10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 52.98% | | | 59.13% | | | 59.38% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.14% | 49.89% | 50.23% | 50.33% | 50.25% | 50.09% | 50.13% | 50.20% | 49.92% |
| $\omega$ | 49.03% | 50.27% | 50.28% | 48.66% | 49.67% | 49.90% | 48.87% | 50.31% | 49.83% |

(e) Filtering

| $\delta$ | | Clarendon | | | Gingham | | | Moon | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 61.03% | | | 59.55% | | | 59.75% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 49.98% | 49.96% | 49.76% | 50.13% | 49.72% | 49.89% | 50.03% | 49.70% | 50.08% |
| $\omega$ | 49.13% | 50.01% | 49.95% | 49.08% | 50.51% | 49.83% | 48.79% | 50.09% | 49.99% |

(f) Rotation, $\delta$–range of rotation degree

| $\delta$ | | $[0°, 10°]$ | | | $[0°, 20°]$ | | | $[0°, 30°]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 58.64% | | | 56.28% | | | 54.63% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.17% | 50.01% | 49.89% | 49.88% | 50.01% | 49.81% | 49.86% | 49.95% | 49.97% |
| $\omega$ | 43.00% | 43.20% | 43.70% | 38.79% | 39.28% | 39.15% | 34.65% | 34.91% | 34.93% |

Table 11: Accuracies and coverage difference rates $\omega$ of our attacks on CIFAR-10 models trained with DP

(a) Gaussian noise, $\sigma$– range of standard deviation

| $\sigma$ | | $[0, 0.1]$ | | | $[0, 0.2]$ | | | $[0, 0.3]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 50.76% | | | 50.64% | | | 50.61% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.22% | 50.25% | 50.02% | 49.95% | 50.01% | 50.37% | 50.19% | 49.96% | 50.04% |
| $\omega$ | 47.32% | 50.06% | 49.92% | 47.61% | 50.18% | 49.62% | 48.22% | 50.05% | 49.99% |

(b) Adversarial noise

| PGD | | 1-step | | | 10-step | | | loss=10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 50.76% | | | 50.67% | | | 50.67% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.14% | 50.06% | 49.96% | 50.04% | 50.11% | 49.79% | 50.08% | 49.96% | 50.07% |
| $\omega$ | 47.98% | 49.57% | 49.79% | 49.47% | 49.93% | 50.07% | 49.05% | 49.86% | 49.76% |

(c) JPEG compression, $q$–compression quality

| $q$ | | 50 | | | 10 | | | 1 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 50.70% | | | 50.50% | | | 50.27% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 49.96% | 50.08% | 50.05% | 50.17% | 50.05% | 50.28% | 49.98% | 50.11% | 50.12% |
| $\omega$ | 46.91% | 49.38% | 50.12% | 47.48% | 50.05% | 49.52% | 47.81% | 50.25% | 49.71% |

(d) Scaling, $r$–scaling factor

| $\delta$ | | 0.5 | | | 1.5 | | | 10 | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 50.39% | | | 50.70% | | | 50.73% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 49.96% | 49.92% | 50.07% | 49.99% | 50.12% | 49.93% | 50.09% | 49.90% | 50.00% |
| $\omega$ | 48.04% | 50.21% | 50.02% | 47.00% | 49.86% | 49.89% | 47.33% | 49.70% | 50.00% |

(e) Filtering

| $\delta$ | | Clarendon | | | Gingham | | | Moon | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 50.86% | | | 50.68% | | | 50.70% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.25% | 49.71% | 50.04% | 49.94% | 50.28% | 49.78% | 49.90% | 50.16% | 49.89% |
| $\omega$ | 46.54% | 49.79% | 49.97% | 46.94% | 49.71% | 49.96% | 47.67% | 49.88% | 50.07% |

(f) Rotation, $\delta$–range of rotation degree

| $\delta$ | | $[0°, 10°]$ | | | $[0°, 20°]$ | | | $[0°, 30°]$ | |
|---|---|---|---|---|---|---|---|---|---|
| LTT | | 50.66% | | | 50.52% | | | 50.46% | |
| $\epsilon$ | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 | 0.1 | 0.01 | 0.001 |
| RT | 50.35% | 50.07% | 50.16% | 49.56% | 49.91% | 50.14% | 49.70% | 50.12% | 49.90% |
| $\omega$ | 48.28% | 49.37% | 49.63% | 47.58% | 48.77% | 48.86% | 46.64% | 48.10% | 48.34% |

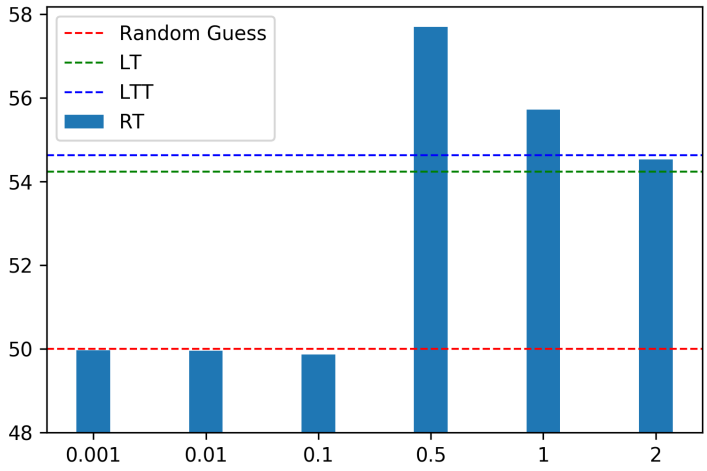



Figure 4: Comparison of different $\epsilon$-robust areas for RT attack (CIFAR-10 with weight-decay defense, Rotation of $[0°, 30°]$). The x-axis represents the $\epsilon$, and the y-axis represents the attack accuracy.