A APPENDIX

B SUPPLEMENTARY MATERIAL

B.1 QUALITATIVE AND QUANTITATIVE EVALUATIONS ON RANDOM SAMPLING AND CONVOLUTION LAYER OUTPUT

Figure 1 shows their visual comparisons, with the objective of finding the most similar perturbed sample (measured by MINE with the maximal scaled L_∞ perturbation bound $\epsilon = 1$) leading to misclassification. Both random sampling and convolution-based approaches can generate high-similarity prediction-evasive adversarial examples despite of large L_∞ perturbation.

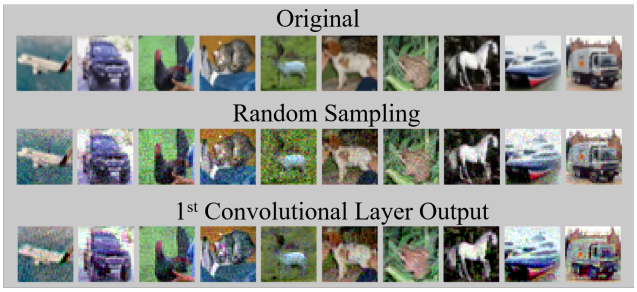


Figure 1: Visual comparison of MINE-based supervised adversarial examples (untargeted attack with $\epsilon = 1$) on CIFAR-10. Both random sampling and convolution output can be used to craft adversarial examples with high similarity.

Table 5 compares the Frechet inception distance (FID) Heusel et al. (2017) and the kernel inception distance (KID) Bińkowski et al. (2018) between the generated adversarial examples versus the training data (lower value is better). Both per-sample MINE methods have comparable scores. The convolution-based approach attains lower KID score and is observed to have better visual quality as shown in Figure 1.

Table 5: Frechet and kernel inception distances (FID/KID) between the untargeted adversarial examples of 1000 test samples and the training data in CIFAR-10 for the proposed per-sample MINEs.

	Random Sampling (10 runs, $K = 96$)	1st Convolution Layer Output ($K = 96$)
FID	339.47 ± 8.07	344.231
KID	14.86 ± 1.45	10.78

B.2 MORE DETAILS ON PER-SAMPLE MINE

B.2.1 RANDOM SAMPLING

We reshape an input data sample as a vector $x \in \mathbb{R}^d$ and independently generate K Gaussian random matrices $\{M_k\}_{k=1}^K$, where $M_k \in \mathbb{R}^{d' \times d}$. Each entry in M_k is an i.i.d zero-mean Gaussian random variable with standard deviation $1/d'$. The compressed samples $\{x_k\}_{k=1}^K$ of x is defined as $x_k = M_k x$. Similarly, the same random sampling procedure is used on $x + \delta$ to obtain its compressed samples $\{(x+\delta)_k\}_{k=1}^K$. In our implementation, we set $d' = 128$ and $K = 500$.

B.2.2 CONVOLUTION LAYER OUTPUT

Given a data sample x, we fetch its output of the 1^{st} convolutional layer, denoted by $conv(x)$. The data dimension is $d' \times K$, where K is the number of filters (feature maps) and d' is the (flattend) dimension of the feature map. Each filter is regarded as a compressed sample denoted by $conv(x)_k$. Algorithm 1 summarizes the proposed approach, where the function T_θ is parameterized by a neural

network θ based on the Donsker-Varadhan representation theorem Donsker & Varadhan (1983), and T_I is the number of iterations for training the MI neural estimator $I(\theta)$.

Algorithm 1 Per-sample MINE via Convolution

Require: input sample x, perturbed sample $x+\delta$, 1st convolution layer output $conv(\cdot)$, MI neural estimator $I(\theta)$
Initialize neural network parameters θ
Get $\{conv(x)_k\}_{k=1}^K$ and $\{conv(x+\delta)_k\}_{k=1}^K$ via 1^{st} convolution layer
for t in T_I iterations **do**
  Take K samples from the joint distribution: $\{conv(x)_k, conv(x+\delta)_k\}_{k=1}^K$
  Shuffle K samples from $conv(x+\delta)$ marginal distribution: $\{conv(x+\delta)_{(k)}\}_{k=1}^K$
  Evaluate $I(\theta) \leftarrow \frac{1}{K}\sum_{k=1}^K T_\theta(conv(x)_k, conv(x+\delta)_k) - \log\left(\frac{1}{K}\sum_{k=1}^K \exp[T_\theta(conv(x)_k, conv(x+\delta)_{(k)})]\right)$
  $\theta \leftarrow \theta + \nabla_\theta I(\theta)$
end for
Return $I(\theta)$

B.3 MinMax Attack Algorithm

The parameters α and β denote the step sizes of the minimization and maximization steps, respectively. The gradient $\nabla f_x^+(x+\delta)$ with respect to δ is set to be 0 when $f_x(x+\delta) \leq 0$. Our MinMax algorithm returns the successful adversarial example $x+\delta^*$ with the best MINE value $I_\Theta^*(x, x+\delta^*)$ over T iterations.

Algorithm 2 MinMax Attack Algorithm

Require: data sample x, attack criterion $f_x(\cdot)$, step sizes α and β, perturbation bound ϵ, # of iterations T
Initialize $\delta_0 = 0$, $c_0 = 0$, $\delta^* =$ null, $I_\Theta^* = -\infty$, $t = 1$
for t in T iterations **do**
  $\delta_{t+1} = \delta_t - \alpha \cdot (c \cdot \nabla f_x^+(x+\delta_t) - \nabla I_\Theta(x, x+\delta_t))$
  Project δ_{t+1} to $[\epsilon, -\epsilon]$ via clipping
  Project $x+\delta_{t+1}$ to $[0,1]$ via clipping
  Compute $I_\Theta(x, x+\delta_{t+1})$
  Perform $c_{t+1} = (1 - \frac{\beta}{t^{1/4}}) \cdot c_t + \beta \cdot f_x^+(x+\delta_{t+1})$
  Project c_{t+1} to $[0, \infty]$
  if $f_x(x+\delta_{t+1}) \leq 0$ and $I_\Theta(x, x+\delta_{t+1}) > I_\Theta^*$ **then**
    update $\delta^* = \delta_{t+1}$ and $I_\Theta^* = I_\Theta(x, x+\delta_{t+1})$
  end if
end for
Return δ^*, I_Θ^*

B.4 Experiment Setup and Datasets

Datasets We provide a brief summary of the datasets:
• **MNIST** consists of grayscale images of hand-written digits. The number of training/test samples are 60K/10K.
• **SVHN** is a colar image dataset set of house numbers extracted from Google Street View images. The number of training/test samples are 73257/26302.
• **Fashion MNIST** contains grayscale images of 10 clothing items. The number of training/test samples are 60K/10K.
• **Isolet** consists of preprocessed speech data of people speaking the name of each letter of the English alphabet. The number of training/test samples are 6238/1559.
• **Coil-20** contains grayscale images of 20 multi-viewed objects. The number of training/test samples are 1152/288.

• **Mice Protein** consists of the expression levels (features) of 77 protein modifications in the nuclear fraction of cortex. The number of training/test samples are 864/216.
• **Human Activity Recognition** consists of sensor data collected from a smartphone for various human activities. The number of training/test samples are 4252/1492.

Unsupervised Adversarial Example Setting Only the training data samples are used in the unsupervised setting. Their true labels are used in the post-hoc analysis for evaluating the quality of the associated unsupervised learning tasks. All training data are used for generating UAEs individually by setting $\kappa = 0$. A perturbed data sample is considered as a successful attack if its loss (relative to the original sample) is no greater than the original training loss (see Table 1). For data augmentation, if a training sample fails to find a successful attack, we will replicate itself to maintain data balance. The ASR is measured on the training data, whereas the reported model performance is evaluated on the test data. For completeness, the training performance is provided in the supplementary material.

MinMax Algorithm Parameters We use consistent parameters by setting $\alpha = 0.01$, $\beta = 0.1$, and $T = 40$ as the default values. The vanilla MINE model Belghazi et al. (2018) is used in our per-sample MINE implementation. We also study the sensitivity analysis of the parameters and report the results in the supplementary material.

Computing Resource All experiments are conducted using an Intel Xeon E5-2620v4 CPU, 125 GB RAM and a NVIDIA TITAN Xp GPU with 12 GB RAM.

Models and Codes We defer the summary of the considered machine learning models to the corresponding sections. Our codes are provided in the supplementary material.

B.5 Compared to other data augmentation methods

Table 6 further demonstrates UAEs can improve data reconstruction when the original model involves augmented training data such as flip, rotation, and Gaussian noise. The augmentation setup is given in the supplementary material.

Table 6: Performance evaluation of data reconstruction when retraining with UAEs from augmented training data.

SVNH - Convolutional AE		
Augmentation	Original+Aug. (test set)	+MINE-UAE (test set)
Flip + Rotation	0.00285	**0.00107** ($\uparrow$ 62.46%)
Gaussian noise ($\sigma = 0.01$)	0.00107	**0.00095** ($\uparrow$ 11.21%)
Flip + Rotation + Gaussian noise	0.00307	**0.00099** ($\uparrow$ 67.75%)

B.6 Additional Visual Comparisons

B.6.1 Visual Comparison of Unsupervised Adversarial Examples

Figure 2 shows the generated MINE-UAEs with $\epsilon = 1$ on SVHN using the convolutional autoencoder. We pick the 10 images such that their reconstruction loss is no greater than that of the original image, while they have the top-10 perturbation level measured by the L_2 norm on the perturbation $\|\delta^*\|_2$.

B.7 More Details on Data Augmentation and Model Retraining

Table 7 summarizes the training epochs and training loss of the models and datasets used in Section 3.

B.8 More Details on Table 6

For all convolutional autoencoders, we use 100 epochs and early stopping if training loss converges at early stage. For data augmentation of the training data, we set the rotation angle $= 10$ and use both horizontal and vertical flip. For Gaussian noise, we use zero-mean and set $\sigma = 0.001$.

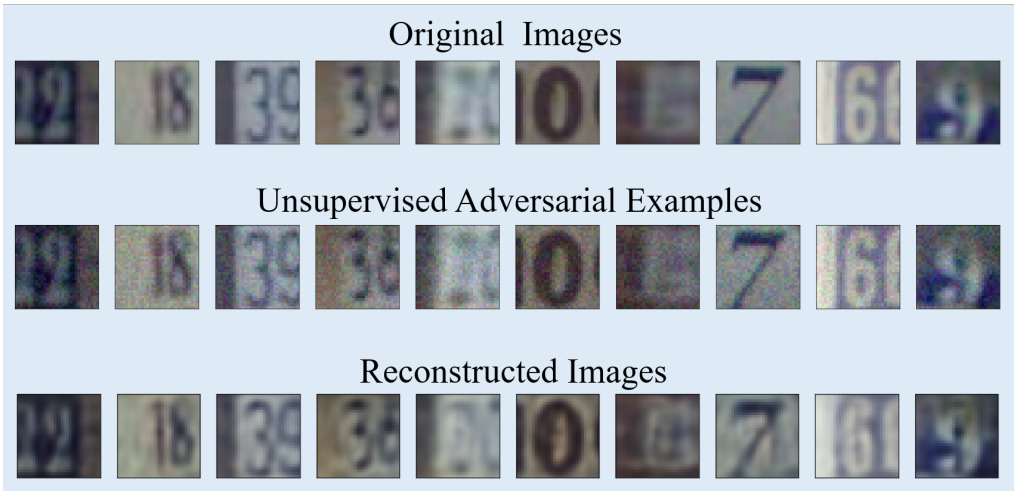


Figure 2: Visual comparison of MINE-based unsupervised adversarial examples on SVHN using convolutional autoencoder.

Table 7: Details on the training epochs and losses for the datasets and models used in Section 3. The reconstruction error is the average L_2 reconstruction loss of the training set.

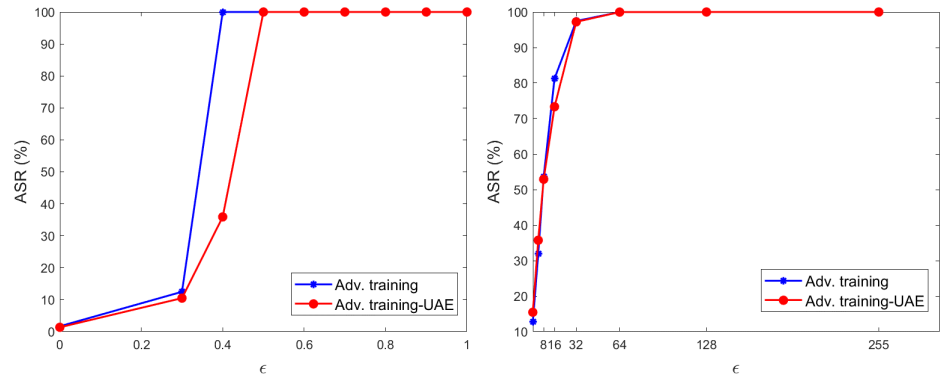
MNIST									
	Training Epochs				Reconstruction Error (training set)				
Autoencoder	Original	MINE-UAE	L_2-UAE	GA ($\sigma = 10^{-2}/10^{-3}$)	Original	MINE-UAE	L_2-UAE	GA ($\sigma = 10^{-2}$)	GA ($\sigma = 10^{-3}$)
Sparse	50	80	80	80	0.00563	0.00233	0.00345	0.00267±2.93e-05	0.00265±3.60e-5
Dense	20	30	30	30	0.00249	0.00218	0.00275	0.00231±0.00013	0.0023±0.00011
Convolutional	20	30	30	30	0.00301	0.00260	0.00371	0.00309±0.00013	0.00310±0.00015
Adversarial	50	80	80	80	0.044762	0.04612	0.06063	0.058711±0.00659	0.05551±0.00642
SVHN									
Sparse	50	80	80	80	0.00729	0.00221	0.00290	0.00283±0.00150	0.00275 ± 0.00081
Dense	30	50	50	50	0.00585	0.00419	0.00503	0.00781±0.00223	0.00781±0.00187
Convolutional	We set 100 epochs, but the training loss converges after 5 epochs				0.00140	0.00104	0.00131	0.00108±3.83e-05	0.00113±6.76e-05
Adversarial	We set 200 epochs and use the model with the lowest training loss				0.00169	0.00124	0.02729	0.00158±0.00059	0.00130±0.00036

B.9 Hyperparameter Sensitivity Analysis

All of the aforementioned experiments were using $\beta = 0.1$ and $\alpha = 0.01$. Here we show the results on MNIST with convolution autoencoder (Section 3) and the concrete autoencoder (Section **??**) using different combinations of α and β values. The results are comparable, suggesting that our MINE-based data augmentation is robust to a wide range of hyperpameter values.

Table 8: Comparison of reconstruction error (test set) of convolution and concrete autoencoders with different combinations of α and β values on MNIST.

MNIST						
	Convolution AE			Concrete AE		
α \ β	0.05	0.1	0.5	0.05	0.1	0.5
0.01	0.00330	0.00256	0.00283	0.01126	0.01142	0.01134
0.05	0.00296	0.00278	0.00285	0.01129	0.01133	0.01138



(a) MNIST (b) CIFAR-10

Figure 3: Performance evaluation of ASR for adv. training and adv. training-UAE against PGD attack with different ϵ value. Adv. training-UAE consistently shows lower or comparable ASR than adv. training, suggesting that data augmentation using UAE can improve adversarial training.

B.10 More Data Augmentation Runs

While the first fun of UAE-based data augmentation is shown to improve model performance, here we explore the utility of more data augmentation runs. We conducted two data augmentation runs for sparse autoencoder on SVHN. We re-train the model with 1^{st}-run UAEs, 2^{nd}-run UAEs (generated by 1^{st}-run augmented model) and original training data. The reconstruction error on the test set of 2^{nd} data augmentation is 0.00199, which slightly improves the 1^{st}-run result (0.00235). In general, we find that 1^{st}-run UAE data augmentation has a much more significant performance gain comparing to the 1^{st}-run results.

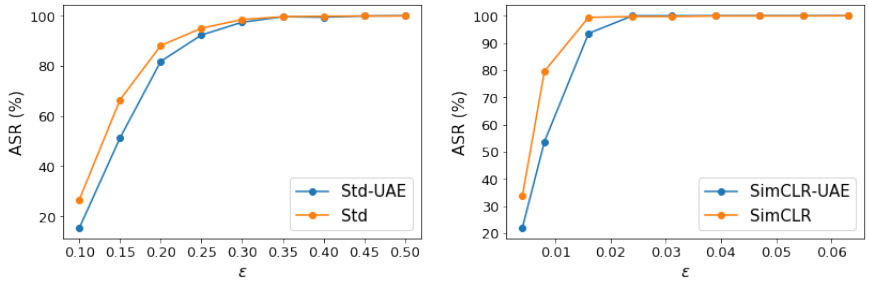
Adversarial Training with MINE-based Unsupervised Adversarial Examples We use the MNIST and CIFAR-10 models to compare the performances of standalone adversarial training (Adv. training) Madry et al. (2018) and adversarial training plus data augmentation by MINE-based unsupervised adversarial examples (Adv. training-UAE) generated from convolutional Autoencoder. Figure 3 shows the attack success rate (ASR) of Adv. training model and Adv training-UAE against PGD attack. For all PGD attacks, we use 100 steps and set step size $= 2.5 * \frac{\epsilon}{\text{number of steps}}$. When $\epsilon = 0.4$, Adv. training-UAE model can still resist more than 60% of adversarial examples on MNIST. By contrast, ASR is 100% for Adv. training model. For CIFAR-10, ASR of Adv. training-UAE model is about 8% lower than Adv. training model when $\epsilon = 16$. We therefore conclude that data augmentation using UAE can improve adversarial training.

B.11 IMPROVED ADVERSARIAL ROBUSTNESS AFTER DATA AUGMENTATION WITH MINE-UAES

To evaluate the adversarial robustness after data augmentation with MINE-UAEs, we use the MNIST and CIFAR-10 models in Section 3. We randomly select 1000 classified correctly images (test set) to generate adversarial examples. For all PGD attacks, We set step size$= 0.01$ and use 100 steps.

In our first experiment (Figure 4 (a)), we train the convolutional classifier (Std) and Std with UAE (Std-UAE) generated from the convolutional autoencoder on MNIST. The attack success rate (ASR) of the Std-UAE is consistently lower than the Std for each ϵ value.

In the second experiment (Figure 4 (b)), the ASR of SimCLR-UAE is significantly lower than that of the original SimCLR model, especially for $\epsilon \leq 0.02$. When $\epsilon = 0.01$, SimCLR-UAE on CIFAR-10 can still resist more than 40% of adversarial examples, which is significantly better than the original SimCLR model. Based on the empirical results, we therefore conclude that data augmentation using UAE can improve adversarial robustness of unsupervised machine learning tasks.



(a) MNIST (convolutional-autoencoder) (b) CIFAR-10 (SimCLR)

Figure 4: Robustness evaluation of attack success rate (ASR) for the original model (Std/SimCLR) and the models trained with UAE augmentation (Std-UAE/ SimCLR-UAE) against PGD attack with different ϵ values. Models trained with UAE shows better robustness (lower ASR) than original models, implying that data augmentation with UAE can strengthen the adversarial robustness.