Adversarial Examples can be Effective Data Augmentation for Unsupervised Machine Learning

Chia-Yi Hsu\textsuperscript{1}, Pin-Yu Chen\textsuperscript{2}, Songtao Lu\textsuperscript{2}, Sijia Liu\textsuperscript{3}, Chia-Mu Yu\textsuperscript{1}
\textsuperscript{1}National Yang Ming Chiao Tung University
\textsuperscript{2}IBM Research
\textsuperscript{3}Michigan State University

\{chiayhsu8315, chiamuyu\}@gmail.com, \{pin-yu.chen, songtao\}@ibm.com, liusiji5@msu.edu

Abstract

Adversarial examples causing evasive predictions are widely used to evaluate and improve the robustness of machine learning models. However, current studies focus on supervised learning tasks, relying on the ground-truth data label, a targeted objective, or supervision from a trained classifier. In this paper, we propose a framework of generating adversarial examples for \textit{unsupervised} models and demonstrate novel applications to data augmentation. Our framework exploits a mutual information neural estimator as an information-theoretic similarity measure to generate adversarial examples without supervision. We propose a new MinMax algorithm with provable convergence guarantees for efficient generation of unsupervised adversarial examples. Our framework can also be extended to supervised adversarial examples. When using unsupervised adversarial examples as a simple plug-in data augmentation tool for model retraining, significant improvements are consistently observed across different unsupervised tasks and datasets, including data reconstruction, representation learning, and contrastive learning. Our results show novel methods and considerable advantages in studying and improving unsupervised machine learning via adversarial examples.

1 Introduction

Adversarial examples are known as prediction-evasive attacks on state-of-the-art machine learning models (e.g., deep neural networks), which are often generated by manipulating native data samples while maintaining high similarity measured by task-specific metrics such as $L_p$-norm bounded perturbations [Goodfellow, Shlens, and Szegedy 2015; Biggio and Roli 2018]. Due to the implications and consequences on mission-critical and security-centric machine learning tasks, adversarial examples are widely used for robustness evaluation of a trained model and for robustness enhancement during training (i.e., adversarial training).

Despite of a plethora of adversarial attacking algorithms, the design principle of existing methods is primarily for \textit{supervised} learning models — requiring either the true label or a targeted objective (e.g., a specific class label or a reference sample). Some recent works have extended to the \textit{semi-supervised} setting, by leveraging supervision from a classifier (trained on labeled data) and using the predicted labels on unlabeled data for generating (semi-supervised) adversarial examples [Miyato et al. 2018; Zhang et al. 2019; Stanforth et al. 2019; Carmon et al. 2019]. On the other hand, recent advances in unsupervised and few-shot machine learning techniques show that task-invariant representations can be learned and contribute to downstream tasks with limited or even without supervision [Ranzato et al. 2007; Zhu and Goldberg 2009; Zhai et al. 2019], which motivates this study regarding their robustness. Our goal is to provide efficient robustness evaluation and data augmentation techniques for unsupervised (and self-supervised) machine learning models through \textit{unsupervised} adversarial examples (UAEs). Table I summarizes the fundamental difference between conventional supervised adversarial examples and our UAES. Notably, our UAE generation is supervision-free because it solely uses an information-theoretic similarity measure and the associated unsupervised learning objective function. It does not use any supervision such as label information or prediction from other supervised models.

In this paper, we aim to formalize the notion of UAE, establish an efficient framework for UAE generation, and demonstrate the advantage of UAES for improving a variety of unsupervised machine learning tasks. We summarize our main contributions as follows.

\begin{itemize}
  \item We propose a new per-sample based mutual information neural estimator (MINE) between a pair of original and modified data samples as an information-theoretic similarity measure and a supervision-free approach for generating UAE. For
\end{itemize}
instance, see UAEs for data reconstruction in Figure 5 of Supplementary material. While our primary interest is generating adversarial examples for unsupervised learning models, we also demonstrate that our per-sample MINE can be used to generate adversarial examples for supervised learning models with improved visual quality.

- We formulate the generation of adversarial examples with MINE as a constrained optimization problem, which applies to both supervised and unsupervised machine learning tasks. We then develop an efficient MinMax optimization algorithm (Algorithm 1) and prove its convergence. We also demonstrate the advantage of our MinMax algorithm over the conventional penalty-based method.
- We show a novel application of UAEs as a simple plug-in data augmentation tool for several unsupervised machine learning tasks, including data reconstruction, representation learning, and contrastive learning on image and tabular datasets. Our extensive experimental results show outstanding performance gains (up to 73.5% performance improvement) by retraining the model with UAEs.

2 Related Work and Background

2.1 Adversarial Attack and Defense

For supervised adversarial examples, the attack success criterion can be either untargeted (i.e., model prediction differs from the true label of the corresponding native data sample) or targeted (i.e., model prediction targeting a particular label or a reference sample). In addition, a similarity metric such as $L_p$-norm bounded perturbation is often used when generating adversarial examples. The projected gradient descent (PGD) attack (Madry et al. 2018) is a widely used approach to find $L_p$-norm bounded supervised adversarial examples. Depending on the attack threat model, the attacks can be divided into white-box (Szegedy et al. 2013; Carlini and Wagner 2017b), black-box (Chen et al. 2017; Brendel, Rauber, and Bethge 2018; Liu et al. 2020), and transfer-based (Nitin Bhagoji et al. 2018; Papernot et al. 2017) approaches.

Although a plethora of defenses were proposed, many of them failed to withstand advanced attacks (Carlini and Wagner 2017a; Athalye, Carlini, and Wagner 2018). Adversarial training (Madry et al. 2018) and its variants aiming to generate worst-case adversarial examples during training are so far the most effective defenses. However, adversarial training on supervised adversarial examples can suffer from undesirable tradeoff between robustness and accuracy (Su et al. 2018; Tsipras et al. 2019). Following the formulation of untargeted supervised attacks, recent studies such as (Cemgil et al. 2020) generate adversarial examples for unsupervised tasks by finding an adversarial example within an $L_p$-norm perturbation constraint that maximizes the training loss. In contrast, our approach aims to find adversarial examples that have low training loss but are dissimilar to the native data (see Table 1), which plays a similar role to the category of “on-manifold” adversarial examples governing generalization errors (Stutz, Hein, and Schiele 2019). In supervised setting, (Stutz, Hein, and Schiele 2019) showed that adversarial training with $L_p$-norm constrained perturbations may find off-manifold adversarial examples and hurt generalization.

2.2 Mutual Information Neural Estimator

Mutual information (MI) measures the mutual dependence between two random variables $X$ and $Z$, defined as $I(X, Z) = H(X) - H(X|Z)$, where $H(X)$ denotes the (Shannon) entropy of $X$ and $H(X|Z)$ denotes the conditional entropy of $X$ given $Z$. Computing MI can be difficult without knowing the marginal and joint probability distributions ($P_X$, $P_Z$, and $P_{XZ}$). For efficient computation, the mutual information neural estimator (MINE) with consistency guarantees is proposed in (Belghazi et al. 2018). Specifically, MINE aims to maximize the lower bound of the exact MI using a model parameterized by a neural network $\theta$, defined as $I_\theta(X, Z) \leq I(X, Z)$, where $\Theta$ is the space of feasible parameters of a neural network, and $I_\theta(X, Z)$ is the neural information quantity defined as $I_\theta(X, Z) = \sup_{\theta \in \Theta} E_{P_{XZ}}[T_\theta] - \log(E_{P_X \otimes P_Z} e^{T_\theta})$. The function $T_\theta$ is parameterized by a neural network $\theta$ based on the Donsker-Varadhan representation theorem (Donsker and Varadhan 1983). MINE estimates the expectation of the quantities above by shuffling the samples from the joint distribution along the batch axis or using empirical samples ($x_i, z_i$) from $P_{XZ}$ and $P_X \otimes P_Z$ (the product of marginals).

MINE has been successfully applied to improve representation learning (Hjelm et al. 2019; Zhu, Zhang, and Evans 2020) given a dataset. However, for the purpose of generating an adversarial example for a given data sample, the vanilla MINE is not applicable because it only applies to a batch of data samples (so that empirical data distributions can be used for computing MI estimates) but not to single data sample. To bridge this gap, we will propose two MINE-based sampling methods for single data sample in Section 3.1.

3 Methodology

3.1 MINE of Single Data Sample

Given a data sample $x$ and its perturbed sample $x + \delta$, we construct an auxiliary distribution using their random samples or convolution outputs to compute MI via MINE as a similarity measure, which we denote as “per-sample MINE”.

**Random Sampling** Using compressive sampling (Candes and Wakin 2008), we perform independent Gaussian sampling of a given sample $x$ to obtain a batch of $K$ compressed samples $\{x_k, (x + \delta)_k\}_{k=1}^K$. For computing $I_\theta(x, x + \delta)$ via MINE. We refer the readers to the supplementary material (SuppMat 6.2) for more details. We also note that random sampling is agnostic to the underlying machine learning model since it directly applies to the data sample.

**Convolution Layer Output** When the underlying neural network model uses a convolution layer to process the input data (which is an almost granted setting for image data), we propose to use the output of the first convolution layer of a data input, denoted by $\text{conv}(\cdot)$, to obtain $K$ feature maps $\{\text{conv}(x), \text{conv}(x + \delta)\}_{K=1}^K$ for computing $I_\theta(x, x + \delta)$. We provide the detailed algorithm for convolution-based per-sample MINE in SuppMat 6.2.

**Evaluation** We use the CIFAR-10 dataset and the same neural network as in Section 4.2 to provide qualitative and quantitative evaluations on the two per-sample MINE methods for image classification. Figure 1 shows their visual com-
We formalize the objectives for supervised/unsupervised adversarial examples. We also tested the performance using the second convolution (or being a targeted class Bińkowski et al. 2018) between the generated adversarial examples using per-sample MINE. As summarized in Table 2, Frechet and kernel inception distances (FID/KID) [8] between the untargeted adversarial examples of 1000 test samples and the training data in CIFAR-10. Table 2 compares the Frechet inception distance (FID) and the kernel inception distance (KID) (Heusel et al. 2017) observed to have better visual quality as shown in Figure 1. The convolution-based approach attains lower KID score and is found degraded performance. In this paper we propose to use MINE to find the least similar perturbed data sample x + δ with respect to x while ensuring the reconstruction loss of Φ(x + δ) is no greater than Φ(x) (i.e., the criterion of successful attack for data reconstruction). The unsupervised attack formulation is as follows:

\[ \text{Minimize} \quad I_\delta(x, x + \delta) \]

such that \( x + \delta \in [0, 1]^d, \delta \in [-\epsilon, \epsilon]^d \) and \( f_\delta(x + \delta) \leq 0 \)

The first two constraints regulate the feasible data space and the perturbation range. For the \( L_2 \)-norm reconstruction loss, the unsupervised attack function is

\[ f^{\text{unsup}}_\delta(x + \delta) = ||x - \Phi(x + \delta)||_2 - ||x - \Phi(x)||_2 + \kappa \]

which means the attack is successful (i.e., \( f^{\text{unsup}}_\delta(x + \delta) \leq 0 \) if the reconstruction loss of \( x + \delta \) relative to the original sample \( x \) is smaller than the native reconstruction loss minus a non-negative margin \( \kappa \). That is, \( ||x - \Phi(x + \delta)||_2 \leq ||x - \Phi(x)||_2 - \kappa \). In other words, our unsupervised attack formulation aims to find that most dissimilar perturbed sample \( x + \delta \) to \( x \) measured by MINE while having smaller reconstruction loss (in reference to \( x \)) than \( x \). Such UAEs thus relates to generalization errors on low-loss samples because the model is biased toward these unseen samples.

### 3.2 MINE-based Attack Formulation

We formalize the objectives for supervised/unsupervised adversarial examples using per-sample MINE. As summarized in Table 1, the supervised setting aims to find most similar examples causing prediction evasion, leading to an MINE maximization problem. The unsupervised setting aims to find least similar examples but having smaller training loss, leading to an MINE minimization problem. Both problems can be solved efficiently using our unified MinMax algorithm.

**Supervised Adversarial Example** Let \((x, y)\) denote a pair of a data sample \( x \) and its ground-truth label \( y \). The objective of supervised adversarial example is to find a perturbation \( \delta \) to \( x \) such that the MI estimate \( I_\Theta(x, x + \delta) \) is maximized while the prediction of \( x + \delta \) is different from \( y \) (or being a targeted class \( y' \neq y \)), which is formulated as

\[ \text{Maximize} \quad I_\Theta(x, x + \delta) \]

such that \( x + \delta \in [0, 1]^d, \delta \in [-\epsilon, \epsilon]^d \) and \( f_\times(x + \delta) \leq 0 \).

We also tested the performance using the second convolution layer output but found degraded performance. In this paper we use convolution-based approach whenever applicable and otherwise use random sampling.

### 3.3 MINE-based Attack Algorithm

Here we propose a unified MinMax algorithm for solving the aforementioned supervised and unsupervised attack formulations, and provide its convergence proof in Section 3.4. For simplicity, we will use \( f_\times \) to denote the attack criterion for \( f^{\text{sup}}_\times \) or \( f^{\text{unsup}}_\times \). Without loss of generality, we will analyze

Figure 1: Visual comparison of MINE-based untargeted supervised adversarial examples (with \( \epsilon = 1 \)) on CIFAR-10.
the supervised attack objective of maximizing $I_{\Theta}$ with constraints. The analysis also holds for the unsupervised case since minimizing $I_{\Theta}$ is equivalent to maximizing $I_{\Theta}'$, where $I_{\Theta}' = -I_{\Theta}$. We will also discuss a penalty-based algorithm as a comparative method to our proposed approach.

**MinMax Algorithm (proposed)** We reformulate the attack generation via MINE as the following MinMax optimization problem with simple convex set constraints:

$$\min_{\delta: x + \delta \in [0, 1]^d, \delta \in [-\epsilon, \epsilon]^d} \max_{\epsilon \in [0, \epsilon]} F(\delta, \epsilon) \triangleq c f_+^T(x + \delta) - I_{\Theta}(x, x + \delta)$$

The outer minimization problem finds the best perturbation $\delta$ with data and perturbation feasibility constraints $x + \delta \in [0, 1]^d$ and $\delta \in [-\epsilon, \epsilon]^d$, which are both convex sets with known analytical projection functions. The inner maximization associates a variable $c \geq 0$ with the original attack function $f_+(x + \delta) \leq 0$, where $c$ is multiplied to the ReLU activation function of $f_+$, denoted as $f_+^T(x + \delta) = \text{ReLU}(f_+(x + \delta)) = \max(f_+(x + \delta), 0)$. The use of $f_+^T$ means when the attack criterion is not met (i.e., $f_+(x + \delta) > 0$), the loss term $c \cdot f_+(x + \delta)$ will appear in the objective function $F$. On the other hand, if the attack criterion is met (i.e., $f_+(x + \delta) \leq 0$), then $c \cdot f_+^T(x + \delta) = 0$ and the objective function $F$ only contains the similarity loss term $-I_{\Theta}(x, x + \delta)$. Therefore, the design of $f_+^T$ balances the tradeoff between the two loss terms associated with attack success and MINE-based similarity. We propose to use alternative projected gradient descent between the inner and outer steps to solve the MinMax attack problem, which is summarized in Algorithm 1. The parameters $\alpha$ and $\beta$ denote the step sizes of the minimization and maximization steps, respectively. The gradient $\nabla f_+^T(x + \delta)$ with respect to $\delta$ is set to be 0 when $f_+(x + \delta) \leq 0$. Our MinMax algorithm returns the successful adversarial example $x + \delta^*$ with the best MINE value $I_{\Theta}^*(x + \delta^*)$ over $T$ iterations.

**Algorithm 1: MinMax Attack Algorithm**

1. **Require**: data sample $x$, attack criterion $f_+(\cdot)$, step sizes $\alpha$ and $\beta$, perturbation bound $c$, # of iterations $T$
2. Initialize $\delta_0 = 0$, $c_0 = 0$, $\delta^* = \text{null}$, $I_{\Theta}^* = -\infty$, $t = 1$
3. for $t$ in $T$ iterations do
4. $\delta_{t+1} = \delta_t - \alpha \cdot (c \cdot \nabla f_+^T(x + \delta_t) - \nabla I_{\Theta}(x, x + \delta_t))$
5. Project $\delta_{t+1}$ to $[-\epsilon, \epsilon]$ via clipping
6. Project $x + \delta_{t+1}$ to $[0, 1]$ via clipping
7. Compute $I_{\Theta}(x, x + \delta_{t+1})$
8. $c_t = 1 - \frac{\beta}{T+1} \cdot c_t + \beta \cdot f_+^T(x + \delta_{t+1})$
9. Project $c_t$ to $[0, \infty]$
10. if $f_+(x + \delta_{t+1}) \leq 0$ and $I_{\Theta}(x, x + \delta_{t+1}) > I_{\Theta}^*$ then
11. update $\delta^* = \delta_{t+1}$ and $I_{\Theta}^* = I_{\Theta}(x, x + \delta_{t+1})$
12. Return $\delta^*$, $I_{\Theta}^*$

**Penalty-based Algorithm (baseline)** An alternative approach to solving the MINE-based attack formulation is the penalty-based method with the objective:

$$\min_{\delta: x + \delta \in [0, 1]^d, \delta \in [-\epsilon, \epsilon]^d} \frac{c}{2} \cdot f_+^T(x + \delta) - I_{\Theta}(x, x + \delta)$$

where $c$ is a fixed regularization coefficient instead of an optimization variable. Prior arts such as (Carlini and Wagner [2017b]) use a binary search strategy for tuning $c$ and report the best attack results among a set of $c$ values. In contrast, our MinMax attack algorithm dynamically adjusts the $c$ value in the inner maximization stage (step 8 in Algorithm 1). In Section 4.2, we will show that our MinMax algorithm is more efficient in finding MINE-based adversarial examples than the penalty-based algorithm. The details of the binary search process are given in SuppMat 5.4. Both methods have similar computation complexity involving $T$ iterations of gradient and MINE computations.

### 3.4 Convergence Proof of MinMax Attack

As a theoretical justification of our proposed MinMax attack algorithm (Algorithm 1), we provide a convergence proof with the following assumptions on the considered problem:

- **A.1**: The feasible set $\Delta$ for $\delta$ is compact, and $f_+^T(x + \delta)$ has (well-defined) gradients and Lipschitz continuity (with respect to $\delta$) with constants $L_f$ and $L_L$. That is, $|f_+(x + \delta) - f_+(x + \delta')| \leq L_f \|\delta - \delta'\|$ and $|\nabla f_+^T(x + \delta) - \nabla f_+^T(x + \delta')| \leq L_L \|\delta - \delta'\|$, $\forall \delta, \delta' \in \Delta$. Moreover, $I_{\Theta}(x, x + \delta)$ also has gradient Lipschitz continuity with constant $L_l$.

- **A.2**: The per-sample MINE is $\eta$-stable over iterations for the same input. $|I_{\Theta_{t+1}}(x, x + \delta_{t+1}) - I_{\Theta_{t}}(x, x + \delta_{t+1})| \leq \eta$. A.1 holds in general for neural networks since the numerical gradient of ReLU activation can be efficiently computed and the sensitivity (Lipschitz constant) against the input perturbation can be bounded (Weng et al. 2018). The feasible perturbation set $\Delta$ is compact when the data space is bounded. A.2 holds by following the consistent estimation proof of the native MINE in (Belghazi et al. 2018).

To state our main theoretical result, we first define the proximal gradient of the objective function as $L(\delta, c) := \delta - P_X[\delta - \nabla \Lambda(\delta, c)]$, $c \in [-\epsilon, \epsilon]$, and $\Lambda(\delta, c) := \frac{1}{2} \|f_+(x + \delta) - I(\Theta)\|^2 + c \|\nabla \Lambda(\delta, c)\|$. Theorem 1 states the rate of convergence of our proposed MinMax attack algorithm when provided with sufficient stability of MINE and proper selection of the step sizes. We also remark that under the assumptions and conditions of step-sizes, this convergence rate is standard in non-convex min-max saddle point problems (Lu et al. 2020).

**Theorem 1.** Suppose Assumptions A.1 and A.2 hold and the sequence $\{\delta_t, c_t, t \geq 1\}$ is generated by the MinMax attack algorithm. For a given small constant $\epsilon'$ and positive constant $\beta$, let $T(\epsilon')$ denote the first iteration index such that the following inequality is satisfied:

$T(\epsilon') := \min\{t \|L(\delta_t, c_t)\|^2 \leq \epsilon', t \geq 1\}$. Then, when the step-size and approximation error achieved by Algorithm 1 satisfy $s = \eta \leq \sqrt{T(\epsilon'/\epsilon)}$, there exists some constant $\mathcal{C}$ such that $\|L(\delta_{T(\epsilon')}, \epsilon_{T(\epsilon')})\| \leq \mathcal{C}/\sqrt{T(\epsilon')}$. Proof. Please see the supplemental material (SuppMat 6.3).

Theorem 1 states the rate of convergence of our proposed MinMax attack algorithm when provided with sufficient stability of MINE and proper selection of the step sizes. We also remark that under the assumptions and conditions of step-sizes, this convergence rate is standard in non-convex min-max saddle point problems (Lu et al. 2020).
3.5 Data Augmentation using UAE

With the proposed MinMax attack algorithm and per-sample MINE for similarity evaluation, we can generate MINE-based supervised and unsupervised adversarial examples (UAEs). Section 4 will show novel applications of MINE-based UAEs as a simple plug-in data augmentation tool to boost the model performance of several unsupervised machine learning tasks. We observe significant and consistent performance improvement in data reconstruction (up to 73.5% improvement), representation learning (up to 1.39% increase in accuracy), and contrastive learning (1.58% increase in accuracy). The observed performance gain can be attributed to the fact that our UAEs correspond to “on-manifold” data samples having low training loss but are dissimilar to the training data, causing generalization errors. Therefore, data augmentation and retraining with UAEs can improve generalization (Stutz, Hein, and Schiele 2019).

4 Performance Evaluation

In this section, we conduct extensive experiments on a variety of datasets and neural network models to demonstrate the performance of our proposed MINE-based MinMax adversarial attack algorithm and the utility of its generated UAEs for data augmentation, where a high attack success rate using UAEs suggests rich space for data augmentation to improve model performance. Codes are available at https://github.com/IBM/UAE.

4.1 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 color 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. • Islet 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 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.

Supervised Adversarial Example Setting Both data samples and their labels are used in the supervised setting. We select 1000 test images classified correctly by the pretrained MNIST and CIFAR-10 deep neural network classifiers used in (Carlini and Wagner 2017b) and set the confidence gap parameter $κ = 0$ for the designed attack function $f_{x}^{sup}$ defined in Section 3.2. The attack success rate (ASR) is the fraction of the final perturbed samples leading to misclassification.

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 $κ = 0$. 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. The training performance is provided in SuppMat 6.10.

MinMax Algorithm Parameters We use consistent parameters by setting $α = 0.01$, $β = 0.1$, and $T = 40$ as the default values. The vanilla MINE model (Belghazi et al. 2018) is used in our per-sample MINE implementation. The parameter sensitivity analysis is reported in SuppMat 6.13.

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 SuppMat.

4.2 MinMax v.s. Penalty-based Algorithms

We use the same untargeted supervised attack formulation and a total of $T = 9000$ iterations to compare our proposed MinMax algorithm with the penalty-based algorithm using 9 binary search steps on MNIST and CIFAR-10. Table 3 shows that while both methods can achieve 100% ASR, MinMax algorithm attains much higher MI values than penalty-based algorithm. The results show that the MinMax approach is more efficient in finding MINE-based adversarial examples, which can be explained by the dynamic update of the coefficient $c$ in Algorithm 1.

Figure 2 compares the statistics of MI values over attack iterations. One can find that as iteration count increases, MinMax algorithm can continue improving the MI value, whereas penalty-based algorithm saturates at a lower MI value due to the use of fixed coefficient $c$ in the attack process. In the remaining experiments, we will report the results using MinMax algorithm due to its efficiency.

4.3 Qualitative Visual Comparison

Table 3: Comparison between MinMax and penalty-based algorithms on MNIST and CIFAR-10 datasets in terms of attack success rate (ASR) and mutual information (MI) value averaged over 1000 adversarial examples.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>MNIST</th>
<th>CIFAR-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASR</td>
<td>MI</td>
<td>ASR</td>
</tr>
<tr>
<td>Penalty-based</td>
<td>100%</td>
<td>28.28</td>
</tr>
<tr>
<td>MinMax</td>
<td>100%</td>
<td>51.29</td>
</tr>
</tbody>
</table>

Table 3: Comparison between MinMax and penalty-based algorithms on MNIST and CIFAR-10 datasets in terms of attack success rate (ASR) and mutual information (MI) value averaged over 1000 adversarial examples.

figure presents a visual comparison of MNIST supervised adversarial examples crafted by MinMax attack and the PGD attack with 100 iterations (Madry et al. 2018) given different $ε$ values governing the $L_{∞}$ perturbation bound. The main difference is that MinMax attack uses MINE as an additional similarity regulation while PGD attack only uses $L_{∞}$ norm.
Next, we study three different unsupervised learning tasks. Here we use the default implementation of the following four autoencoders to generate UAEs based on the training data samples of MNIST and SVHN for data augmentation, retrain the model from scratch on the augmented dataset, and report the resulting reconstruction error on the original test set. The results of larger-scale datasets (CIFAR-10 and Tiny-ImageNet) are reported in SuppMat 6.11. All autoencoders use the $L_2$ reconstruction loss defined as $||x - \Phi(x)||_2$. We provide more details about the model retraining in SuppMat 6.10.

- **Dense Autoencoder** (Cavallari, Ribeiro, and Ponti 2018): The encoder and decoder have 1 dense layer separately and the latent dimension is 128/256 for MNIST/SVHN.
- **Sparse Autoencoder**: It has a sparsity enforcer ($L_1$ penalty on the training loss) that directs a network with a single hidden layer to learn the latent representations minimizing the error in reproducing the input while limiting the number of code words for reconstruction. We use the same architecture as Dense Autoencoder for MNIST and SVHN.
- **Convolutional Autoencoder (Makhzani et al. 2016)**: It is composed of an encoder, a decoder and a discriminator. The rationale is to force the distribution of the encoded values to be similar to the prior data distribution.

We also compare the performance of our proposed MINE-based UAE (MINE-UAE) with two baselines: (i) $L_2$-UAE that replaces the objective of minimizing $I_\Theta(x, x + \delta)$ with maximizing the $L_2$ reconstruction loss $||x - \Phi(x + \delta)||_2$ in the MinMax attack algorithm while keeping the same attack success criterion; (ii) **Gaussian augmentation** (GA) that adds zero-mean Gaussian noise with a diagonal covariance matrix of the same constant $\sigma^2$ to the training data.

Table 5 shows the reconstruction loss and the ASR. The improvement of reconstruction error is measured with respect to the reconstruction loss of the original model (i.e., without data augmentation). We find that MINE-UAE can attain much higher ASR than $L_2$-UAE and GA in most cases. More importantly, data augmentation using MINE-UAE achieves consistent and significant reconstruction performance improvement across all models and datasets (up to 56.7% on MNIST and up to 73.5% on SVHN), validating the effectiveness of MINE-UAE for data augmentation. On the other hand, in several cases $L_2$-UAE and GA lead to notable performance degradation. The results suggest that MINE-UAE can be an effective plug-in data augmentation tool for boosting the performance of unsupervised machine learning models.

Table 6 demonstrates UAEs can further improve data reconstruction when the original model already involves conventional augmented training data such as flip, rotation, and Gaussian noise. The augmentation setup is given in SuppMat 6.12. We also show the run time analysis of different augmentation methods in SuppMat 6.7.

## 4.4 UAE Improves Data Reconstruction

Data reconstruction using an autoencoder $\Phi(\cdot)$ that learns to encode and decode the raw data through latent representations is a standard unsupervised learning task. Here we use the default implementation of the following four autoencoders to generate UAEs based on the training data samples of MNIST and SVHN for data augmentation, retrain the model from scratch on the augmented dataset, and report the resulting reconstruction error on the original test set. The results of larger-scale datasets (CIFAR-10 and Tiny-ImageNet) are reported in SuppMat 6.11. All autoencoders use the $L_2$ reconstruction loss defined as $||x - \Phi(x)||_2$. We provide more details about the model retraining in SuppMat 6.10.

- **Dense Autoencoder** (Cavallari, Ribeiro, and Ponti 2018): The encoder and decoder have 1 dense layer separately and the latent dimension is 128/256 for MNIST/SVHN.
- **Sparse Autoencoder**: It has a sparsity enforcer ($L_1$ penalty on the training loss) that directs a network with a single hidden layer to learn the latent representations minimizing the error in reproducing the input while limiting the number of code words for reconstruction. We use the same architecture as Dense Autoencoder for MNIST and SVHN.
- **Convolutional Autoencoder (Makhzani et al. 2016)**: It is composed of an encoder, a decoder and a discriminator. The rationale is to force the distribution of the encoded values to be similar to the prior data distribution.

We also compare the performance of our proposed MINE-based UAE (MINE-UAE) with two baselines: (i) $L_2$-UAE that replaces the objective of minimizing $I_\Theta(x, x + \delta)$ with maximizing the $L_2$ reconstruction loss $||x - \Phi(x + \delta)||_2$ in the MinMax attack algorithm while keeping the same attack success criterion; (ii) **Gaussian augmentation** (GA) that adds zero-mean Gaussian noise with a diagonal covariance matrix of the same constant $\sigma^2$ to the training data.

Table 4 shows the reconstruction loss and the ASR. The improvement of reconstruction error is measured with respect to the reconstruction loss of the original model (i.e., without data augmentation). We find that MINE-UAE can attain much higher ASR than $L_2$-UAE and GA in most cases. More importantly, data augmentation using MINE-UAE achieves consistent and significant reconstruction performance improvement across all models and datasets (up to 56.7% on MNIST and up to 73.5% on SVHN), validating the effectiveness of MINE-UAE for data augmentation. On the other hand, in several cases $L_2$-UAE and GA lead to notable performance degradation. The results suggest that MINE-UAE can be an effective plug-in data augmentation tool for boosting the performance of unsupervised machine learning models. Table 5 demonstrates UAEs can further improve data reconstruction when the original model already involves conventional augmented training data such as flip, rotation, and Gaussian noise. The augmentation setup is given in SuppMat 6.12. We also show the run time analysis of different augmentation methods in SuppMat 6.7.

### 4.5 UAE Improves Representation Learning

The concrete autoencoder (Balin, Abid, and Zou 2019) is an unsupervised feature selection method which recognizes a subset of the most informative features through an additional

We use [https://github.com/shibuiwilliam/Keras_Autoencoder](https://github.com/shibuiwilliam/Keras_Autoencoder)
Table 4: Comparison of data reconstruction by retraining the autoencoder on UAE-augmented data. The error is the average reconstruction loss of the test set. The improvement (in green/red) is relative to the original model. The attack success rate (ASR) is the fraction of augmented training data having smaller reconstruction loss than the original loss (see Table 1 for definition).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Augmentation</th>
<th>Original</th>
<th>MINE-UAE</th>
<th>L2-UAE</th>
<th>GA ($\sigma = 0.01$)</th>
<th>GA ($\sigma = 10^{-2}$)</th>
<th>MINE-UAE</th>
<th>L2-UAE</th>
<th>GA ($\sigma = 0.01$)</th>
<th>GA ($\sigma = 10^{-2}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>Flip + Rotation</td>
<td>0.00235</td>
<td>0.00315</td>
<td>0.00301 ± 0.00137</td>
<td>0.00293 ± 0.0078</td>
<td>100% 72.16% 72.42% 79.92%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVHN</td>
<td>Flip + Rotation</td>
<td>0.00235</td>
<td>0.00315</td>
<td>0.00301 ± 0.00137</td>
<td>0.00293 ± 0.0078</td>
<td>100% 72.16% 72.42% 79.92%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>L2-UAE GA(3)</td>
<td>0.26755</td>
<td>0.26755</td>
<td>1.58% 91.36% 91.36% 91.36%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Performance of data reconstruction when retraining with MINE-UAE and additional augmented training data.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Original</th>
<th>MINE-UAE</th>
<th>L2-UAE</th>
<th>GA ($\sigma = 0.01$)</th>
<th>GA ($\sigma = 10^{-2}$)</th>
<th>MINE-UAE</th>
<th>L2-UAE</th>
<th>GA ($\sigma = 0.01$)</th>
<th>GA ($\sigma = 10^{-2}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.01370</td>
<td>0.01442</td>
<td>2.4%</td>
<td>94.97% 95.41% 95.41% 95.41%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fashion MMIST</td>
<td>0.01307</td>
<td>0.01254</td>
<td>4.1%</td>
<td>84.92% 85.24% 85.24% 85.24%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Islet</td>
<td>0.01200</td>
<td>0.01159</td>
<td>3.4%</td>
<td>81.98% 82.93% 82.93% 82.93%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coil-20</td>
<td>0.00693</td>
<td>0.00734</td>
<td>4.3%</td>
<td>98.96% 98.96% 98.96% 98.96%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mice Protein</td>
<td>0.00651</td>
<td>0.00661</td>
<td>6.1%</td>
<td>89.81% 89.81% 89.81% 89.81%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Activity</td>
<td>0.00337</td>
<td>0.00300</td>
<td>11.0%</td>
<td>83.38% 84.45% 84.45% 84.45%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Comparison of contrastive loss and the resulting accuracy on CIFAR-10 using SimCLR (Chen et al. 2018) (ResNet-18 with batch size = 512). The attack success rate (ASR) is the fraction of augmented training data having smaller contrastive loss than original loss. For CLAE (Ho and Vasconcelos 2020), we use the reported accuracy improvement (it shows negative gain in our implementation), though its base SimCLR model only has 83.27% test accuracy.

5 Conclusion

In this paper, we propose a novel framework for studying adversarial examples in unsupervised learning tasks, based on our developed per-sample mutual information neural estimator as an information-theoretic similarity measure. We also propose a new MinMax algorithm for efficient generation of MINE-based supervised and unsupervised adversarial examples and establish its convergence guarantees. As a novel application, we show that MINE-based UAs can be used as a simple yet effective plug-in data augmentation tool and achieve significant performance gains in data reconstruction, representation learning, and contrasting learning.
Acknowledgments

Chia-Yi Hsu and Chia-Mu Yu were supported by MOST 110-2636-E-009-018, and we also thank National Center for High-performance Computing (NCHC) of National Applied Research Laboratories (NARLabs) in Taiwan for providing computational and storage resources.

References


