Journal of Information Security and Applications 75 (2023) 103502

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Dual adversarial attacks: Fooling humans and classifiers

Johannes Schneider [*], Giovanni Apruzzese

Institute of Information Systems, University of Liechtenstein, Liechtenstein

ARTICLE INFO

Keywords:
Adversarial attacks
Dual attacks
Computer vision
Deep learning

ABSTRACT

Adversarial samples mostly aim at fooling machine learning (ML) models. They often involve minor pixel-based perturbations that are imperceptible to human observers. In this work, adversarial samples should fool both humans and ML models, which is important in two-stage decision processes. We perform changes on a higher abstraction level so that a target sample exhibits properties of a desired sample. Technically, we contribute by deriving a regularization scheme for autoencoders incorporating a classifier loss for smoothly interpolating between wildly different samples. The realism and effectiveness of generated samples are confirmed with a user study and other evaluations. Our experiments consider neural networks of four architectures, assessed on MNIST, FashionMNIST, QuickDraw and CIFAR-10. Results show that our scheme leads to superior performance compared to existing interpolation techniques: on average, other methods have an 11% higher failure rate when producing a sample that is of any of two interpolated classes. Furthermore, our attacks work in both white- and black-box settings.

1. Introduction

Attacks based on adversarial samples (AS) are a key security threat for machine learning (ML) models. The most common goal of "classic" AS is to mislead the ML model while being invisible to a human observer [3]. Such samples are commonly produced through minimal changes of an ordinary, non-adversarial sample. Many methods to achieve this goal have been proposed (for surveys, see [4–6]). The scope of our paper is different: we aim to mislead the human as well (or only the human) by applying larger (and "visible") changes to our AS. This scenario has received much less attention [7,8]. It is highly relevant in a two-stage decision process that involves both a human and an ML classifier. For instance, an online merchant might upload an item for sale to an online platform. The platform might employ artificial intelligence (AI) methods, such as ML models, to derive information from the image of such an item; e.g., for cars, it might compute a price estimate, or assign the car to a specific category.[1] In turn, a human (i.e., a potential customer) might search the online platform – thereby looking at the same input image provided to the ML model – and make a purchasing decision that might at least partially be based on the image.[2] A malicious merchant – i.e., an attacker – might aim to simultaneously mislead (a) the ML model so that the item for sale is assigned a more favorable category, and (b) the human to make her buy the item. The attacker might walk a thin line between legally permissible nudging (common in marketing [9]) and illegal forms of deception. However, central to our work is that the attacker produces samples analyzed by humans and AI and should mislead both. For example, when selling a piece of clothing, an attacker might aim to make it appear more fashionable and newer than it actually is so that an AI classifies it as "looks new" and a human also judges it to be in good condition. Similar examples are also plausible in other contexts, such as sales of artwork or online dating, where people aim to attract the interest of others by "enhancing their samples" or adjusting to current trends, or appearing more like another artwork or a particular person, i.e., slightly change their photo to look like their favorite movie star.

In this work, we aim to shift (by applying an adversarial perturbation) a given original sample towards a target sample chosen by the attacker: the generated AS should be similar to the original sample but exhibit properties of the target sample. In Table 1, we show the four options that result from fooling: a human, an AI, both, or none (in a classification context). Generally, we aim for an aligned decision process between humans and AI. That is, if a human shows some uncertainty on the class of a sample, so should the classifier (and vice-versa). In other words, either the classifier and the human are confused, or neither. Practically, we might still aim that only the classifier predicts the wrong class, while the human at least shows uncertainty about the correct class (and vice-versa).

To generate AS, we use an autoencoder (AE) with regularized latent space. While regularization of latent spaces is common, we are among

* Corresponding author.
E-mail addresses: johannes.schneider@uni.li (J. Schneider), giovanni.apruzzese@uni.li (G. Apruzzese).

[1] E.g., for Mercedes Benz see https://group.mercedes-benz.com/careers/about-us/artificial-intelligence/for-nerds/pricing.html.

[2] Of course, the user can use also other data sources to make her decision, but images tend to be the most relevant one [1,2].

https://doi.org/10.1016/j.jisa.2023.103502

Table 1
Cases in which AS aim to fool humans and/or AI.

Should be fooled?		Human	
		Yes	No
AI	Yes	This work	Classic AS
	No	This work	No fooling

the first (to the best of our knowledge) to explicitly aim at ensuring a gradual interpolation between two samples that aims to be "class-preserving": an interpolated sample (i.e., the AS) should appear to be of the class of either (a) the original or of (b) the target sample. As we will experimentally show, such a condition is commonly violated for plain AE and other methods [10]. That is, interpolating between two classes might yield samples of a third class that is undesirable by the attacker. In short, we make the following **contributions**:

- We tackle an understudied problem: Generating AS that is perceived similarly by a human and a classifier, so that the AS will fool the human and potentially also the classifier.
- Proposing multiple regularization schemes for generating our AS—supported by theoretical analyses. Our proposed schemes outperform existing ones for regularizing latent spaces.
- Evaluating all of the above quantitatively, qualitatively, and with a user study.

Furthermore, we also publicly release our source code.[3]

2. Problem statement

We assume an attacker with full knowledge[4] of the targeted classifier C. For a sample X, the classifier's logit layer L outputs a value $L(Y|X)$ for each class Y. These are turned into a probability $p(Y|X)$ (typically with a Softmax layer). We assume that a human H can also classify the sample by producing a probability $p_H(Y|X)$, which can reflect the human's "confidence" on sample X. The attacker also has a dataset $\mathcal{D} = \{(X,Y)\}$, containing samples and labels[5] that stem from the data distribution used to train C.

Our goal is to create an adversarial sample X_A (having class Y_A) that is constrained to be similar to an original sample X_O (having class Y_O) and a target sample X_T (having class Y_T); formally, we want that $Y_A \in \{Y_O, Y_T\}$. Additionally, we pursue the following objectives that an adversarial sample X_A should fulfill:

(1) *Similar probabilities for sample X_A by both human and classifier*: $p(Y|X_A) \approx p_H(Y|X_A)$. That is, both the classifier and the human should have roughly aligned judgment of the adversarial sample X_A. However, they might disagree when forced to make a precise inference, i.e., choose a class. That is, it is possible that $\arg\max_Y p(Y|X_A) \neq \arg\max_Y p_H(Y|X_A)$.

(2) *Being similar to both X_O and X_T, controllable by a parameter b*: The adversarial sample $X_A(b)$ should be similar to X_O and X_T, e.g., as measured by the $L1$-norm or human judgment or classifier outputs, and, classified as either the original or target sample class, i.e., $\arg\max_Y p(Y|X_A(b)) \in \{Y_O, Y_T\}$. The attacker should be able to control the similarity by varying a parameter $b \in [0,1]$, i.e. $X_A(b)$, where $b = 0$ means $X_A(0) = X_O$ (i.e., the AS is equal to the original sample) and $b = 1$ means $X_A(1) = X_T$ (i.e., the AS is equal to the target sample).

Thus, we aim at fulfilling objectives (1) and (2) with a function f given the above inputs and objectives that produce an adversarial sample X_A, i.e. $X_A = X_A(b) := f(X_O, X_T, Y_O, Y_T, \mathcal{D}, b)$. Our attack occurs at inference time, i.e., we do not tamper with the classifier C.

3. Background and related work

Our paper falls at the intersection of *ML security* and *generative methods*—both of which belong to the broad research domain of "adversarial machine learning". In what follows, we summarize both of these areas and highlight the major differences of our paper with respect to prior art.

3.1. Attacks against ML

The security of ML has been extensively scrutinized by recent literature (for surveys, we refer the reader to [4–6,14]). Among the most discussed threats to ML are the so-called "adversarial attacks" [15], which entail leveraging (small) perturbations that, when applied to a given sample, yield an adversarial sample (AS) that induces an incorrect response by the targeted ML model. In this paper, we focus on attacks that occur at the inference-stage of the ML model (i.e., "evasion" attacks). Prior work showed that such a vulnerability affects both "traditional" ML classifiers, as well as those based on deep neural networks [16,17]. There is increasing evidence of AS successfully bypassing even operational ML systems [18,19].

In this context, several works aimed at understanding or comparing differences in human and machine perception [8,20–22]. The authors of [8] also aimed to fool humans. They investigated whether AS transfer to humans by matching the human visual system: humans were shown (for 3 s) the image and binary random masks and had to infer the class of a sample. Our work has no time constraints and presents samples in a typical decision setting rather than an artificially constrained one. Harrington et al. [22] identified biologically plausible robust features through carefully crafted user studies, while Zhou and Firestone [21] investigated whether humans could recognize AS generated from scratch. Such samples typically look very awkward for a human but still present similarities to samples of the class, as confirmed by the study.

Harding et al. [23] investigated whether humans could better identify a targeted or non-targeted AS (focused only on fooling ML models). Salamati et al. [24] created AS to fool an ML model through a black-box optimization process, asking humans to select the most indistinguishable samples. Mirsky et al. [25] aimed to fool humans and classifiers for medical imagery. Their focus is on injecting or removing cancer from images, which effectively boils down to adding or removing a gray-scale circular shape. Thus, there is no notion of a gradual change between classes or shift from one sample (original) to another (target): their AS are crafted so that any resemblance to the original sample is lost.

3.2. Generative methods

Techniques that can create samples such as generative adversarial networks [26] as well as (variational) AE have received a lot of attention. Both of these techniques can yield high-quality images, e.g., as shown by NVAE [27] and StyleGAN-XL [28].

Moving within the latent space between two samples is a known procedure often employed in the context of eXplainable Artificial Intelligence (XAI) [29] to understand classifiers [30] or the latent space of AE. Some works aim to shape the latent space in a particular way. For instance, [10,31,32] focused on generating a convex latent distribution. The motivation is that – in a convex space – any interpolation between

[3] Available at this anonymous archive: https://drive.google.com/file/d/1OHg8iNJmtycKk2DoG3-9LUXfy9oX4JPq/view?usp=sharing.

[4] Such an assumption is becoming increasingly more realistic because multiple ML models are now open-source (e.g. [11,12]). However, we also (successfully) assess black-box scenarios by exploiting the transferability property of AS [13].

[5] Labels are non-essential since the attacker can also use the classifier C to obtain less accurate labels.

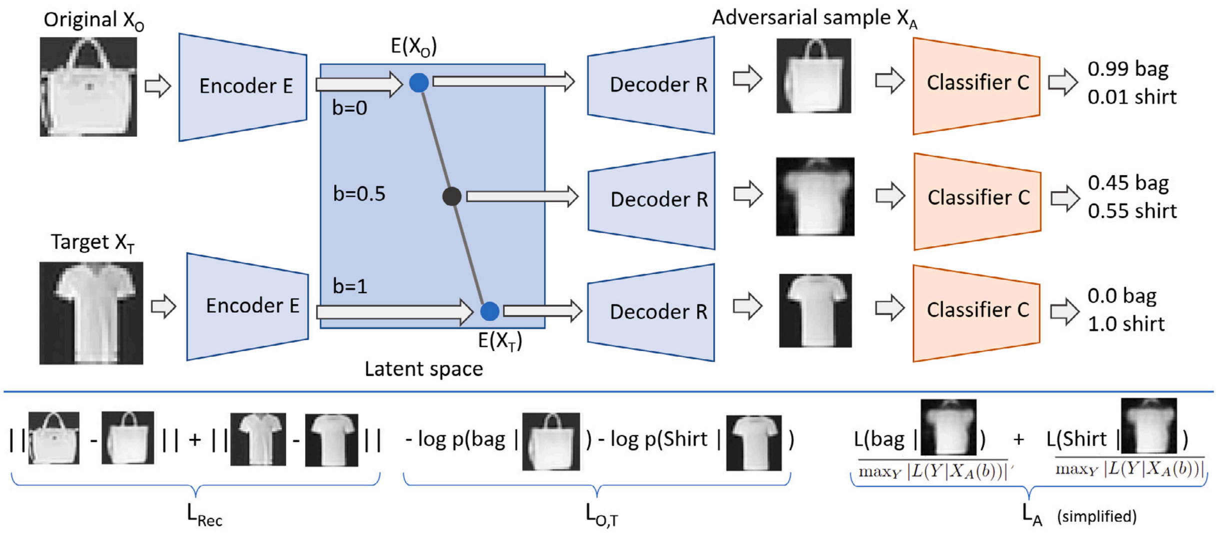

Fig. 1. Outline of our approach. We employ a classifier loss for samples $X_A(0) = X_O$, $X_A(1) = X_T$ and, possibly, $X_A(b)$ for some b, e.g. $b = 0.5$. The classifier helps to regularize the AE.

two points of the convex space must be itself in the convex space. Thus, interpolations should not cross very sparse areas that yield poor reconstructions, i.e., their goal is that the interpolation between any two samples yields realistic samples. In our work, we have the additional constraint (aside from achieving high-quality reconstructions) that the interpolation between two samples of classes Y_O, Y_T should yield only samples that are a mix of characteristics of these classes. In other words, contrarily to us, [10,31,32] allow non-"class preserving" transitions from samples of class Y_O towards a class $Y' \notin \{Y_O, Y_T\}$ and then towards class Y_T, e.g., for digits with Y_O = "1" and Y_T = "3", this could be from "1" to "2" to "3", while we aim primarily for moving directly from "1" to "3". Specifically, the authors of [31] used ideas of a VAE-GAN and a BEGAN, i.e., they added a discriminator in the form of a GAN to an AE architecture. The work in [32] relied on generative models based on normalizing flows, proposing a special norm to ensure smooth linear interpolation. Finally, [10] used an adversarial regularizer for their adversarial constrained AE Interpolation (ACAI): here, the AE is trained to fool the critic so that it outputs that no interpolation took place. In contrast to [10,31,32], our proposed scheme enforces that the interpolated samples (i.e., the AS) exhibit characteristics of both original and target samples since we aim to fool both the classifier and the human. Adding a classifier loss to regularize an AE has been done before (e.g., to optimize human-AI interaction [33]), but neither to craft "class-preserving" interpolations nor to generate AS.

Similarly to our work, the authors of [7] aimed at potentially fooling humans and classifiers. They propose to either encode a sample directly using an encoder of an AE or to encode a sample with a classifier using activations of some upper layer, reconstruct it from the upper layer and embed it again using an encoder (from an AE), yielding a latent representation. The motivation is to strip away information irrelevant to a classifier. In their work, they alter the original sample towards the target sample up to some predefined level of distortion. The AS might or might not fool a classifier and/or human. In our work, we apply a regularized AE. We distort the original to the maximum extent to not make the classifier mispredict or just as much to make it mispredict. Practically, [7,10,31] face the problem that they might not lead to samples that appear to be from a distinct class—as we show and discuss in our evaluation and user study (the latter not being included in either of [7,10,31]).

Neural style transfer (see [28,34,35]) allows separating content and style, so that the style of one image can be transferred to that of another, e.g., an image of a camera can be altered to appear like a drawing of an artist. Thus, in typical scenarios of style transfer, images appear very different from their original In contrast, we generally aim not only at changing style but also at changing content and focus on more subtle changes. However, there are certainly scenarios, where changing either only style or only content might be the goal when creating adversarial samples.

4. Architecture and method

Our goal is to create an adversarial sample $X_A(b)$ in between an original sample X_O of class Y_O and a target sample X_T of class Y_T. To do so, we interpolate between both samples in a latent space stemming from a regularized autoencoder $AE = (E, R)$ with encoder E and decoder R trained on the given dataset $\mathcal{D}$. That is, for a given interpolation coefficient $b \in [0, 1]$, we obtain the adversarial sample $X_A(b)$ as:

$$X_A(b) := R(b \cdot E(X_O) + (1 - b) \cdot E(X_T)) \tag{1}$$

Generating high-quality samples X_A with linear interpolation comes with multiple challenges. First, samples might not be visually appealing because the latent space is non-convex. Second, the interpolation between samples of the two classes Y_O, Y_T might yield a sample more similar to another class $Y' \notin \{Y_O, Y_T\}$, which is against our needs and problem statement. We address these problems by enforcing that, for a given adversarial sample X_A, a classifier assigns non-zero outputs for both classes Y_O, Y_T, i.e., the class of the original sample X_O and the class of the target sample X_T. In the next section, we discuss variants of our regularized AE in more detail, and then we elaborate on setting the interpolation coefficient b. Fig. 1 shows the conceptual outline.

4.1. Regularizing the interpolation

We aim for a smooth transition between the original sample X_O and the target sample X_T both visually and in terms of classification, i.e., from Y_O to Y_T. Thus, we add a loss term to the training objective of the AE. We use a weighted loss L_{Tot} consisting of up to three loss terms: (i) a reconstruction loss L_{Rec}; (ii) the cross-entropy loss of the

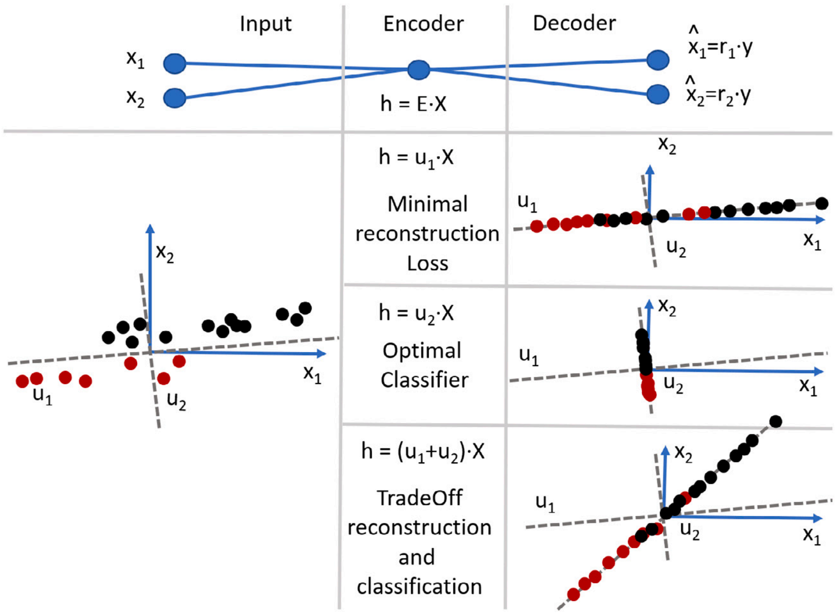

Fig. 2. An AE with optimal encoder $h = u_1 \cdot x$ (and decoder) has lowest reconstruction loss, but it might be impossible or hard for a classifier to classify (autoencoded) samples optimally. Adding a classifier loss of an optimal classifier, e.g., $h = u_2 \cdot x$, allows trading reconstruction loss for better classification of autoencoded samples.

classifier $L_{O,T}$ for X_O and X_T; and (iii) a loss for an AS $X_A(b)$ resulting from some interpolation coefficient b. Using a classifier loss seems reasonable since we are concerned with how a classifier judges the AS. The reconstruction loss L_{Rec} is the standard loss as used for any AE, i.e., the difference between the generated sample and the original sample as measured by the L1-norm.

$$L_{Rec} := \|R(E(X_O)) - X_O\|_1 + \|R(E(X_T)) - X_T\|_1$$

The classifier loss $L_{O,T}$ is given by the cross-entropy loss, i.e., the logarithm of the predicted probability for the correct class:

$$L_{O,T} := -\log p(Y_O|X_O) - \log p(Y_T|X_T)$$

The loss L_A for $X_A(b)$ is more intricate:

$$n(Y') := \min(m_{Y'}, \frac{L(Y'|X_A(b))}{\max_Y |L(Y|X_A(b))|})$$
$$L_A := n(Y_O) + n(Y_T) \tag{2}$$

The term $n(Y')$ resembles normalized logit-values. We found this to work better than using the cross-entropy loss based on $p(Y|X)$. The cross-entropy loss is relatively large for completely wrong predictions, i.e. $|\log p(Y|X)|$ becomes large for small $p(Y|X)$. With the normalized loss, the change is more gradual. We also perform class-dependent loss clipping $m_{Y'}$ for X_A. If $X_A(b)$ should be closer to X_O, i.e. for $b < 0.5$, the value for m_{Y_O} should be larger than that for m_{Y_T} to encourage X_A to be classified as Y_O and vice versa for $b > 0.5$. For simplicity, we assess a variant with only $b = 0.5$ and one where $b \in \{0.35, 0.5, 0.65\}$. The weights m_{Y_O}, m_{Y_A} for some $f, m > 1$ are defined as follows:

$$(m_{Y_O}, m_{Y_A}) := \begin{cases} (m/f, m \cdot f), & \text{for } b = 0.35 \\ (m, m), & \text{for } b = 0.5 \\ (m \cdot f, m/f), & \text{for } b = 0.65 \end{cases}$$

Thus, m specifies the "base weight" of samples X_A and f states how much the classifier should focus on either class when $b \neq 0.5$. Finally, the total loss L_{Tot} is:

$$L_{Tot} := (1 - \alpha - \beta)L_{Rec} + \alpha L_{C(O,X)} + \beta L_A \tag{3}$$

4.2. *Theoretical motivation for classifier loss*

We provide rationale using a linear AE for employing a loss term based on the classifier. In particular, we illustrate that a trade-off between reconstruction loss and classifier loss can be unavoidable.

AE perform a transformation of inputs to a latent space and then back to the original input space. This comes with information loss on the original inputs because reconstructions are not identical to their originals.

An AE aims to find an encoding vector E and a reconstruction vector R, so that the reconstruction $\hat{X} = R \cdot h$ of the encoding $h = E \cdot X$ is minimal using the L2-loss:

$$\min_{E,R} \|X - R \cdot E \cdot X\|^2$$

The optimal solution which minimizes the reconstruction loss stems from projecting onto the eigenvector space (as given by a Principal Component Analysis—see [36]). Given that there is just a single latent variable, the optimal solution for $W = R \cdot E$ is the first eigenvector u_1. This is illustrated in Fig. 2 in the upper part with $h = u_1 \cdot x$. However, this does not allow separating classes perfectly using a linear classifier. In contrast, the classifier loss is minimal if we use $h = u_2 \cdot x$, since then we can perfectly separate all samples, i.e., using $h > 0$ to separate the 'black' dots/class from the 'red' ones, as depicted in the middle part in Fig. 2 with $h = u_2 \cdot x$ Thus, training the AE with a classifier loss for a sample X of class Y, i.e.,

$$\min_{R,E} \|X - R \cdot E \cdot X\|^2 - w_C \log(p(Y|R \cdot E \cdot X))$$

leads to an encoder $E = w_1u_1 + w_2u_2$ for some weights w_1, w_2 that depends on both u_1 and u_2 depending on the given weight w_C of the classifier loss. This implies a tradeoff between reconstruction error and classification accuracy, as indicated in the lower part of Fig. 2.

4.3. *Determining the interpolation coefficient*

For a trained AE, an attacker must still choose an appropriate b to construct samples to his liking (see Eq. (1)). Such an operation

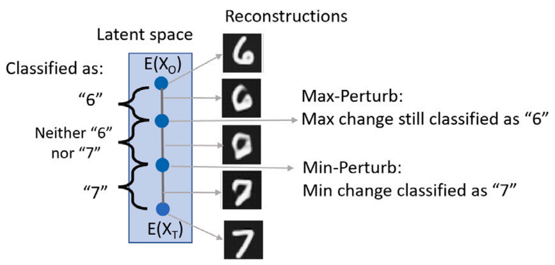

Fig. 3. Illustration of interpolation outcomes and method Max/Min-Perturb. In between two samples X_O, X_T, a reconstructed sample looks very different, potentially being of a class different from Y_O, Y_T.

is necessary to allow an attacker her preferred similarity of X_A with X_O and X_T. We propose three ways to determine the interpolation coefficient b that also cover the options from Table 1, where the goal is to fool a human and a classifier. In other words, both a human and a classifier should always believe that the sample X_A is of either class Y_T or Y_O. However, in general, this might not hold since samples might be poorly recognizable. This issue and two proposed methods are illustrated in Fig. 3.

Let us explain our methods in detail (which depend on b).

- *Max-perturb* (MaP): representing the scenario wherein the classifier should give a correct prediction, but it should still be uncertain about the true class. To this end, we maximize the coefficient b (large b means being closer to the target sample), while ensuring that the classifier recognizes the sample as being of the original class, i.e., Y_O. Formally:

$$b_{MaP} := \max_b \left((\arg\max_Y p(Y|X_A(b))) = Y_O \right) \tag{4}$$

  We solve for b_{MaP} using simple binary search up to a pre-specified precision of $\epsilon = 0.001$ (see Algorithm 1).
- *Min-perturb*: representing the scenario wherein the classifier should give a wrong prediction (with minimal changes). We choose the minimal coefficient b that causes the classifier to classify a sample as Y_T. Note that Min-perturb is symmetrical to the Max-perturb with the original and the target switched.
- *Fixed*: We use a fixed coefficient $b \in [0, 1]$.

We stress that, in all cases, the human should always be confused—i.e., she should be uncertain of the true class of the generated AS (which should be either Y_O or Y_T).

Algorithm 1 Max-Perturb
Input: Target sample X_T, original sample X_O, encoder E, decoder R, classifier Cl
Output: Coefficient b_{MaP}.

```
1: $b_0 := 0; b_1 = 1$ {min/max value for linear interpolation coefficient}
2: $\epsilon := 0.001$ {Precision of interpolation coefficient}
3: while $b_1 - b_0 > \epsilon$ do
4:     $b_{MaP} := \frac{b_0+b_1}{2}$
5:     $E(X_A) := b_{MaP} \cdot E(X_O) + (1 - b_{MaP}) \cdot E(X_T)$ {encoding of adversarial sample}
6:     $X_A := R(E(X_A))$ {(decoded) adversarial sample}
7:     if $(\arg\max_Y p(Y|X_A)) = Y_O$ then $b_0 := b_{MaP}$
8:     else $b_1 := b_{MaP}$
9: end while
10: return $b_{MaP}$
```

5. Evaluation

We now empirically evaluate our proposed method, and corresponding attack. We focus on image classification using convolutional neural networks (CNN). To determine the interpolation coefficient, we use Max-perturb (if we do not explicitly state a fixed coefficient). We first describe the goal (Section 5.1) of our experiments, and then describe the testbed (Section 5.2). We then present our main results (in Section 5.3, which also includes an ablation study), and conclude by assessing the transferability of our AS (5.4). We also conduct a user study, presented in the next section (Section 6).

5.1. Goal

Our goal is to interpolate (smoothly) between samples X_O and X_T of potentially different classes $Y_O \neq Y_T$. In particular, we want to ensure that an AS X_A is of class Y_O or Y_T. While it is relatively easy to create samples that show small differences to either X_O or X_A, it is more challenging to generate a more uniform "mix" of both samples, i.e., samples that could be either of the two classes Y_O, Y_T. For this reason, we aim to evaluate an AS $X_A(b)$ for an interpolation coefficient b at the transition between the two classes, e.g., shortly before switching from Y_O to Y_T or shortly after. The coefficient b_{MaP} (Eq. (4)) from method Max-perturb fits this purpose. Both humans and AI judge AS. Any existing metrics cannot perfectly capture human judgment. Metrics used for generative models focus primarily on the question of whether a sample stems from the training data distribution. For example, the well-known Fréchet inception distance (FID) [37] uses the 2-Wasserstein distance, which in many practical situations boils down to comparing the mean and standard deviation of the deepest layer of a neural network such as Inceptionv3. Such metrics deliberately refrain from computing distances directly on raw pixels, since generated samples being very different from training samples (but still belonging to the training distribution) can be seen as a positive aspect indicating that the model can generate truly novel samples. Novelty is a necessary condition for creativity [38]. In contrast, in our case, the generated sample $X_A(b_{MaP})$ is constrained to be similar to both X_O and X_T for both a human observer and a classifier. As such, differences in pixels play a more profound role. Therefore, we use the $L2$-norm and measure the distance from $X_A(b_{MaP})$ to X_O as well as to X_T.

$$d_A := \|X_A - X_O\| + \|X_A - X_T\| \tag{5}$$

The $L2$-norm can be treated as a proxy for measuring the visual similarity for both humans and AI. Nonetheless, as a supplementary form of validation, we also perform a user study to better assess the human perception of the AS generated with our method.

To measure the 'confusion' of the classifier, we compute the output probabilities for the two desired classes Y_O, Y_T as given by the SoftMax layer.

$$p(Y_O, Y_T) := p(Y_O|X_A(b_{MaP})) + p(Y_T|X_A(b_{MaP}))$$

Thus, by definition the measure $p(Y_O, Y_T)$ is in $[0, 1]$. Ideally, it is 1, indicating that only the two interpolated classes are relevant for any interpolated samples. We are also interested in whether the AS yield a class distinct from Y_O, Y_T for some $b \in [0, 1]$. Technically, this can be done by checking all $b \in [0, 1]$. However, this is tedious. We check only $X_A(b)$, i.e., one b, that is likely of neither of the two classes Y_O, Y_T. We use the observation illustrated in Fig. 3 that the transition is smooth and continuously changing between X_O and X_T, i.e., $X_A(b)$ is classified as Y_O for b close to 0 and as Y_T for b close to 1. For b neither small nor large, there is largest uncertainty. This is also indicated in Fig. 3. Max-perturb gives b_{MaP}, i.e. the largest b, which yields a sample $X_A(b_{MaP})$ classified as Y_O. Thus, if we investigate the class of $X_A(b_{MaP} + \epsilon)$ for a small ϵ, i.e., we use $\epsilon = 0.0025$, then we are likely observing a class different from Y_O and Y_T. By definition of b_{MaP} (Eq. (4)), $X_A(b_{MaP} + \epsilon)$ cannot be of class Y_O, but only of Y_T or any other class distinct from Y_O. If we observe Y_T, then a change to another class $Y' \notin \{Y_O, Y_T\}$ for larger b is less likely, since X_A gets closer to X_T in a gradual manner.

$$1_{\notin\{Y_O,Y_T\}} := \begin{cases} 1, & \text{if } \arg\max_Y p(Y|X_A(b)) \neq Y_T \\ 0, & \text{otherwise} \end{cases}$$

Table 2
Ablation Study. Results for distance to the X_O, X_T are mixed, but for other measures regularizing with a classifier yields significant improvements. **Bold** results are best per dataset, *italic* is second best.

Data	Method	$1_{\notin\{Y_O,Y_T\}}$	$p(Y_O, Y_T)$	d_A
Qu.D.	Vanilla AE	0.3±0.01	0.64±0.0	53.06±0.7
Qu.D.	ACAI	0.37±0.0	0.62±0.0	**50.69±0.0**
Qu.D.	AE w. O,T	0.21±0.0	0.71±0.0	*52.0±0.0*
Qu.D.	AE w. O,T,M	*0.18±0.0*	*0.75±0.0*	52.93±0.0
Qu.D.	AE w. O,T,M_3	**0.15±0.0**	**0.78±0.0**	53.07±0.0
MNI.	Vanilla AE	0.23±0.01	0.71±0.0	**48.11±0.65**
MNI.	ACAI	0.17±0.01	*0.78±0.01*	51.72±0.76
MNI.	AE w. O,T	0.16±0.01	0.77±0.01	*48.56±0.8*
MNI.	AE w. O,T,M	*0.15±0.01*	*0.78±0.01*	48.61±0.73
MNI.	AE w. O,T,M_3	**0.14±0.0**	**0.8±0.0**	49.26±1.23
Fash.	Vanilla AE	0.28±0.01	0.66±0.01	*42.36±0.12*
Fash.	ACAI	0.31±0.03	0.61±0.04	49.12±5.65
Fash.	AE w. O,T	0.17±0.0	0.75±0.0	**42.31±0.39**
Fash.	AE w. O,T,M	**0.15±0.0**	*0.78±0.0*	42.77±0.35
Fash.	AE w. O,T,M_3	**0.15±0.01**	**0.79±0.0**	42.42±0.36
CIFAR-10	Vanilla AE	0.39±0.0	0.49±0.01	**99.57±0.01**
CIFAR-10	ACAI	0.4±0.0	0.45±0.01	102.93±0.34
CIFAR-10	AE w. O,T	***0.29±0.01***	**0.55±0.01**	101.78±0.2
CIFAR-10	AE w. O,T,M	*0.31±0.0*	**0.55±0.0**	*99.97±0.34*
CIFAR-10	AE w. O,T,M_3	0.34±0.02	0.48±0.01	109.52±0.1

5.2. Testbed

Classifiers: We employ four standard classifiers that capture a diverse set of design options to assess the generation of AS themselves, i.e., a ResNet-10 (Res-10) [39], a MobileNet [40] variant with 11 layers called MOB-11, and two variants of VGG [41], i.e., VGG-8 and VGG-11. We used a VGG-11 for the model C to attack. For the encoder, we used a VGG-8 and for the decoder, a standard architecture with deconv layers [42]. For details on networks, we refer the reader to our source code. We use the trained AE to construct AS using Max-Perturb on classifiers {VGG-8, VGG-11, Res-10, MOB-11} to investigate transferability.

Datasets: We employ four datasets: Fashion-MNIST (Fashion) [43], MNIST [44], CIFAR-10 [45], and a subset of 10 classes of Google's QuickDraw dataset.[6] The use of fashion items as found in the Fashion dataset and artwork, i.e., human sketches as captured by QuickDraw has been motivated in the introduction. MNIST supports comparison since it is commonly used in related works, e.g., [10]. For Quickdraw, we used the first ten classes beginning with A, and for each, the first 9k samples for training and 1k for testing, giving a total of 100k samples. Fashion consists of 70k 28 × 28 images of clothing stemming from 10 classes. MNIST of 60k digits objects; CIFAR-10 of 60k colored objects of size 32 × 32 of 10 classes; The Quickdraw dataset consists of 100k samples of human sketches of 10 object classes. For all datasets, 10k samples are used for testing.

Preprocessing: We scaled all images to 32 × 32, performed standardization, and autoencoded all images using a separately trained (Vanilla) plain AE to "smoothen" outliers, i.e., isolated data points in areas of low density that differ strongly from other samples. Generative models fail in areas with low sample density. Our work is no exception to this rule. Thus, we acknowledge that transforming very rare samples is difficult for our method as we also discuss later. For MNIST, autoencoding makes very little difference since AE tend to reconstruct all samples almost perfectly. For Fashion it has little impact for somewhat common samples, but it helps for outliers, e.g., clothes with a seemingly random dotted pattern cannot be reconstructed well and are transformed into more common samples through the separate AE. For such outliers, reconstructions tend to be poor. Thus, they would add significant noise to our evaluation, which we aim to avoid. For CIFAR-10 the impact is most profound, since reconstructions for diverse colored objects are difficult given only 50k images. For testing, we permuted the 10k test data of each dataset, and used 5k samples as X_O and the other 5k as targets X_T, which resulted in 5k samples X_A obtaining b for each pair (X_O, X_T) using method Max-perturb.

Training: We trained on NVIDIA RTX 2080Ti on Ubuntu 20.04 using Pytorch 1.11 and Python 3.9. AE are trained using the Adam optimizer for 160 epochs with batchsize of 128. The classifiers use SGD for 64 epochs starting from a learning rate of 0.1 that was decayed twice by 0.1. We did 3 runs for each reported number, e.g., we trained all networks (classifiers, encoders, decoders) 3 times. We show both averages and standard deviations. The baseline performance on each dataset matches roughly the state-of-the-art of these classifiers without data augmentation, e.g., for VGG-8, we achieved a mean accuracy of about 99% on MNIST and for Fashion above 90% and for Quickdraw about 83%. For CIFAR-10, it was lower due to the autoencoding, i.e., about 50% rather than around 80% for VGG-8 without data augmentation. For ACAI [10], we used the same AE, and as the discriminator the same encoder as in the AE.[7] *Trained AE:* We trained a *Vanilla AE* using only the reconstruction loss, i.e. $\alpha = \beta = 0$ in Eq. (3). We trained an AE *AE w. O,T* using a classifier loss only for $X_A(0) = X_O$ and $X_A(1) = b$, i.e., $\alpha = 0.2, \beta = 0$ in Eq. (3), an *AE w. O,T,M*, i.e. $\alpha = \beta = 0.2$, where M stands for also using a classifier loss for interpolation $b = .5$ and, finally, *AE w. O,T,M_3*, i.e. $\alpha = \beta = 0.2$, where M_3 stands for choosing uniformly at random a $b \in \{0.35, 0.5, 0.65\}$ for each batch and applied then for each pair (X_O, X_A). For f, we used 1.5, and for $m = 0.2$. We elaborate more on the choice of parameters in the discussion.

5.3. Results: Ablation study and comparison

In our ablation study, we compare against a Vanilla AE and ACAI [10]. ACAI is similar to our work in that it regularizes an AE using interpolated samples using a GAN, while we regularize with a classifier. We propose and assess multiple variants that might also be beneficial in other setting such as that of ACAI.

Quantitative Results. Table 2 shows quantitative results of our ablation study.

First, with respect to distance to the original and target sample, i.e. d_A, no method clearly outperforms. The poor performance of ACAI on Fashion can likely be attributed to a failure in GAN training, i.e., the standard deviation for ACAI for the Fashion dataset is fairly large. However, even if we remove benchmark results for failed GAN training of ACAI, the average is not the best for the distance. Regularizing with a classifier (*AE w. O,T (M/M_3)*) clearly outperforms other methods in terms of producing samples of one of the two considered classes Y_O and Y_T, i.e., the metrics $1_{\notin\{Y_O,Y_T\}}$ and $p(Y_O, Y_T)$ are better compared to ACAI and Vanilla AE. This is evident from the low standard deviations (mostly < 0.05), and it is also confirmed by a T-test yielding p-values $< 1e{-}3$. This holds irrespective of the exact regularization, which confirms the effectiveness of our method. For example, other methods fail on average across all four datasets 30% more often to produce a sample that is classified as any of two interpolated classes ($1_{\notin\{Y_O,Y_T\}}$). One can also observe a trade-off for classifier regulation-based methods. Performing better on the distance d_A yields worse performance on the other two metrics and vice-versa. This phenomenon raises the question of whether regularizing only the original and the target sample (i.e., the simplest approach) is the best option. Furthermore, for CIFAR-10 the distance d_A for *AE w. O,T,M_3* is large, i.e., images appear distorted. This is because the classifier giving feedback to the reconstruction model has lower accuracy, i.e., a large loss (compared to the other datasets). Thus, the classifier weight α seems very influential. To better understand the

[6] https://github.com/googlecreativelab/quickdraw-dataset.

[7] Our implementation is based on https://gist.github.com/kylemcdonald/e8ca989584b3b0e6526c0a737ed412f0.

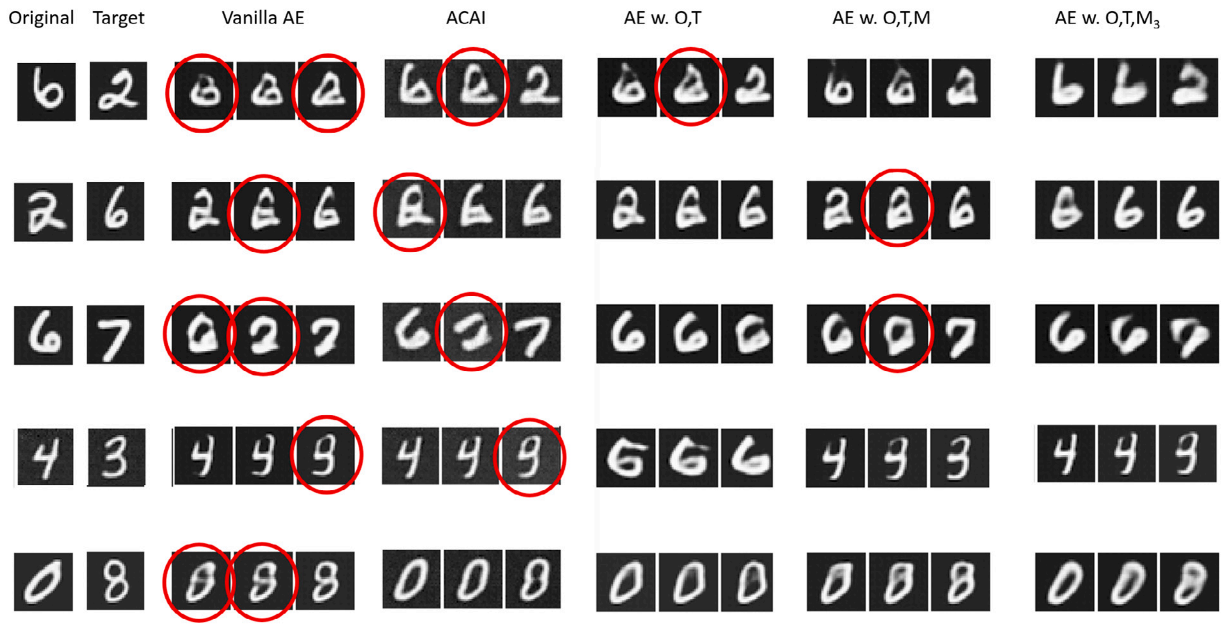

Fig. 4. AS for MNIST for fixed interpolation coefficients $b \in \{0.35, 0.5, 0.65\}$. Red encircled digits highlight difficult to recognize digits and digits appearing from a class different from the original and target.

Table 3
Sensitivity. Strength of regularization (α) is more important than method used.

Dataset	Method	$1_{\notin\{Y_O,Y_T\}}$	$p(Y_O,Y_T)$	d_A
FashionM.	AE w. CL(O,T), $\alpha = 0.2$	0.23±0.01	0.7±0.01	**42.1±0.32**
FashionM.	AE w. CL(O,T), $\alpha = 0.4$	0.17±0.0	0.75±0.0	*42.31±0.39*
FashionM.	AE w. CL(O,T), $\alpha = 0.8$	**0.13±0.0**	**0.79±0.01**	43.88±0.37
FashionM.	AE w. CL(O,T,M_3)	*0.15±0.01*	**0.79±0.0**	42.42±0.36
QuickDraw	AE w. CL(O,T), $\alpha = 0.2$	0.24±0.0	0.68±0.01	*52.79±0.83*
QuickDraw	AE w. CL(O,T), $\alpha = 0.4$	0.22±0.0	0.69±0.01	**52.71±0.71**
QuickDraw	AE w. CL(O,T), $\alpha = 0.8$	**0.14±0.01**	*0.76±0.02*	53.13±0.13
QuickDraw	AE w. CL(O,T,M_3)	*0.15±0.0*	**0.78±0.0**	53.07±0.0
MNIST	AE w. CL(O,T), $\alpha = 0.2$	0.19±0.0	0.74±0.0	**48.44±0.73**
MNIST	AE w. CL(O,T), $\alpha = 0.4$	0.16±0.01	0.77±0.01	*48.56±0.8*
MNIST	AE w. CL(O,T), $\alpha = 0.8$	**0.07±0.01**	**0.85±0.01**	49.76±1.01
MNIST	AE w. CL(O,T,M_3)	*0.14±0.0*	*0.8±0.0*	49.26±1.23
CIFAR-10	AE w. CL(O,T), $\alpha = 0.2$	0.34±0.0	0.52±0.0	**99.07±0.13**
CIFAR-10	AE w. CL(O,T), $\alpha = 0.4$	*0.29±0.01*	*0.55±0.01*	101.78±0.03
CIFAR-10	AE w. CL(O,T), $\alpha = 0.8$	**0.28±0.0**	**0.57±0.0**	108.53±0.16
CIFAR-10	AE w. O,T,M_3	0.34±0.02	0.48±0.01	109.52±0.1

relevance of α, we varied α for the simplest method, i.e., *AE w. O,T*, and found that the achieved trade-offs are comparable to those using classifier loss for AS. The quantitative results of such analysis are in Table 3.

From Table 3 it becomes apparent that the exact regularization scheme for the classifier is not relevant, but the strength α has a profound impact on the trade-off between distance d_A and other metrics. Thus, the simplest scheme using only a classifier loss for the $X_A(0) = X_O$ and $X_A(1) = X_T$ suffices. Regularizing samples in the middle is not clearly beneficial. One reason could be that a fixed b is a too restrictive constraint, i.e., the transition between two classes does not always occur at $b = 0.5$ (it might occur for significantly larger or smaller values). For CIFAR-10 we see that a large $\alpha = 0.8$ can have an adverse effect leading to a large distance d_A, while a small $\alpha = 0.2$ is beneficial, i.e., leading to the lowest distance d_A (even lower than for the Vanilla AE in Table 2). The issue with large α for CIFAR-10 is that since the classifier performance is not as good on CIFAR-10 compared to other datasets (i.e., it has large losses for samples), the classifier loss has a very strong impact on the total loss. This can lead to artifacts. Put differently, the parameter α is sensitive to classifier performance. Artifacts due to classifier loss in combination with AE have also been observed in other contexts, e.g., in the field of explainability [42].

Qualitative Results. Figs. 4 and 5 show qualitative outcomes for MNIST and Fashion (QuickDraw and CIFAR-10 samples are in Appendix).

We discuss MNIST first (Fig. 4). Vanilla AE tends to create difficult to recognize digits or wrong classes. For example, the leftmost sample for Vanilla AE in the third row is difficult to recognize (i.e., a 0, 2, and 6 all seem possible). The sample in the middle in the last row appears more like a 5 than either a 0 or 8, i.e., it is of the wrong class. ACAI tends to create more recognizable figures but interpolation between two classes might still yield non-desirable classes. Using a classifier loss generally yields better results. However, between the classifier loss-based methods there is no clear winner. The outcome appears to be more dependent on the total regularization strength as discussed in the (quantitative) ablation study. Generally, for small weights (for the classifier loss), our method might also create samples that are hard to recognize or appear to be of a different class, but this happens significantly less often as also indicated by the quantitative evaluation. For large weights for the classifier loss, AS rarely appear to be of an incorrect class, but might still be poorly recognizable. Furthermore, for large weights, they appear fairly different from the original.

For Fashion (Fig. 5) reconstructions between classes is more demanding. We highlighted samples that are well-recognizable and contain characteristics of both the original and the target. For example, the shoe in the first row contains the handle (of the handbag) but is still well-recognizable as a shoe. Similarly, the combination of shirts and T-shirts highlights that the sleeves are still visible though small, and the upper part (covering shoulders) appear more pronounced. Most methods struggle when combining very diverse objects such as a boot and a pant (fourth row). Such a task is very challenging. The highlighted sample illustrates that key elements of the boot are preserved, while

Table 4
Transferability. Using Max-perturb on a trained AE w. O,T,M_3 yields effective AS.

Attacked model	Metric	VGG-8(trained w. AE)	VGG-11	MOB-11	Res-10
Fashion	$1_{\notin\{Y_O,Y_T\}}$	0.15±0.01	0.16±0.0	0.22±0.0	0.21±0.01
	$p(Y_O,Y_T)$	0.79±0.0	0.77±0.01	0.71±0.0	0.78±0.01
	d_A	42.42±0.36	42.46±0.36	42.46±0.44	42.43±0.45
QuickDraw	$1_{\notin\{Y_O,Y_T\}}$	0.15±0.0	0.21±0.0	0.28±0.01	0.22±0.0
	$p(Y_O,Y_T)$	0.78±0.0	0.71±0.0	0.63±0.01	0.78±0.01
	d_A	53.07±0.0	53.46±0.78	52.92±0.79	53.31±0.72
MNIST	$1_{\notin\{Y_O,Y_T\}}$	0.14±0.0	0.14±0.0	0.15±0.0	0.16±0.0
	$p(Y_O,Y_T)$	0.8±0.0	0.79±0.0	0.79±0.01	0.81±0.0
	d_A	49.26±1.23	48.77±0.53	48.71±0.53	48.68±0.49
CIFAR-10	$1_{\notin\{Y_O,Y_T\}}$	0.34±0.02	0.34±0.0	0.39±0.01	0.4±0.0
	$p(Y_O,Y_T)$	0.48±0.01	0.48±0.01	0.43±0.01	0.59±0.01
	d_A	109.52±0.1	109.83±0.32	110.32±0.07	109.55±0.14

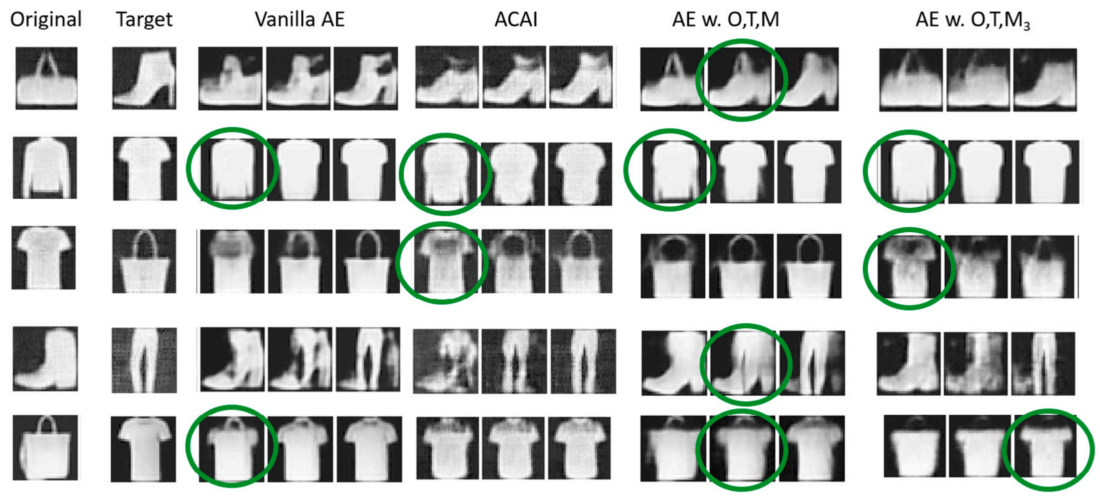

Fig. 5. AS for fashion for fixed interpolation coefficient $b \in \{0.35, 0.5, 0.65\}$. Green encircled samples have aspects of both the original and target and are well-recognizable.

relevant attributes such as the separation between legs are also visible for the pants. Overall, we find that our methods do not only lead to quantitatively better results but also qualitatively.

5.4. Results: Transferability

The construction of AS is a two-step process: Training an AE, then generating an AS $X_A(b)$ by choosing an appropriate b. For transferability, we assume that the generator AE regularized using a classifier C is fixed. When creating an AS to fool a classifier C', we allow searching for b using black-box access to C', i.e., we only obtain the predicted class by C', but no other information such as gradients or output probabilities. This is motivated by our exemplary scenario, where a user uploads information to a webpage. The information is first analyzed by an AI. The user might view the outcome and decide to cancel the process, e.g., for online sales one might preview a sales ad and decide to edit or delete it before it is judged by a human. Table 4 shows that our attack is effective across multiple architectures. For MNIST models appear to behave similarly, however, note the small standard deviation, i.e., there are significant differences between the models. The largest challenges for transfer arose for the QuickDraw dataset. Here the differences between the model used for training (white-box access) and others are most profound, especially, when looking at metric $1_{\notin\{Y_O,Y_T\}}$ (0.15 vs. 0.21–0.28), i.e., that a sample X_A differs form the two preferred classes. We explain this with the increased complexity of the dataset that also yields fairly low classifier accuracy compared to the other dataset and, possibly, larger differences between models.

6. User study

To assess the effectiveness of our proposal on humans, we conduct a user study on Amazon Mechanical Turk (AMT). The purpose of this study is to answer two research questions (RQ):

RQ1: Are real humans *fooled/confused* when analyzing the AS crafted with our method?
RQ2: Can real humans identify the *original and target* samples that were used to yield a given AS?

6.1. Survey

To answer our RQ, we create a survey containing 46 sets of 3 questions, in which we ask AMT workers to provide their opinion on the images generated through our method on the MNIST dataset. Specifically, each set revolves around a specific adversarial sample X_A generated using Max-Perturb from samples X_O, X_T. An exemplary representation of a set of questions is in Fig. 6.

In the first question of each set, we show *only* X_A (i.e., S in Fig. 6) and ask whether the digit shown in X_A represents the number of class Y_O, or the number of class Y_T; the user can respond on a scale between 1 and 7 (where 1 means that they are confident that it is class Y_O, and 7 means that it is class Y_T, whereas 4 means undecided). In the second and third questions, we show the user an image with the adversarial sample X_A, alongside two samples from class Y_O (i.e., A1 and A2 in

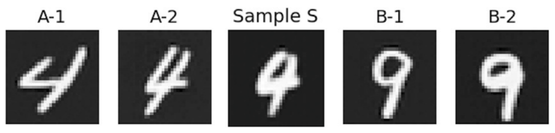

Fig. 6. Question in user study. Here, the AS X_A (called S) was generated via Max-Perturb by interpolating A2 (original) and B2 (target).

Fig. 6) and two samples from class Y_T (i.e., B1 and B2 in Fig. 6); only one between A1 and A2, and between B1 and B2, represents the true original/target sample used to generate X_A. Then, we ask whether X_A is more similar to A1 or to A2 (second question); and whether X_A is more similar to B1 or B2 (third question). For both of these questions, the user can answer on a scale between 1 and 4 (where 1 means that X_A is more similar to A/B1, and 4 that it is more similar to A/B2).[8] The 46 AS were chosen randomly from all the AS generated via the *Max_perturb* technique. Our survey was filled by 32 AMT workers with a hit rate of over 95% to ensure better reliability; on average, a single worker took 15 min to finish the survey.

6.2. Findings

We report the results of our user study in Fig. 7. Here, the left plot shows the distribution of the means for the first question (reflecting RQ1); whereas the right plot shows the distribution of the means for the second and third questions (reflecting RQ2).

From these results, we derive the following conclusions.

RQ1: *humans appear to be very confused.* The average was 4.4 (std: 1.275). Thus, users judged AS neither as being more similar to the original or target but saw elements of both in them. That is, a T-Test yielded that with $p < 1e{-}5$ the mean is larger than 2 and smaller than 6. There is a slight preference for the original, but it is not significant. This is somewhat expected since samples due to max-perturb should still be classified as the original but are right at the boundary.

RQ2: *humans inferred the correct original/target.* The average is 2.84 for A (std: 0.51) and 2.86 for B (std: 0.43). Although there was some uncertainty, a T-test shows that users picked the wrong original/target sample better than guessing, i.e., the mean is different from 2.5 ($p < 1e{-}5$).

7. Discussion and future work

We investigated regularizing an AE with a classifier using multiple variants. Our evaluation yielded that while regularizing with a classifier is highly beneficial, none of the investigated variants clearly outperforms (see Table 2). Overall, using a classifier loss for AS somewhere in between X_O, X_T, e.g. for $b = 0.5$, yielded no clear gain, but added significant complexity, i.e., hyperparameters β, f, m in addition to α. In the course of writing the paper, we also investigated various other hyperparameter settings, e.g., increasing α and $\beta \in [0.05, 0.8]$ (the weight of the classifier losses), $m \in [0.05, 1]$ (the maximal loss per sample) and $f \in [1, 3]$ (the weight of the classifier loss for $b \in \{0.35, 0.65\}$). All yielded qualitatively similar effects, i.e., they increase reconstruction loss but yield better classifiable AS—such a trade-off can be unavoidable as shown in our theoretical treatment. Even with significant tuning, the benefits of the more complex schemes across all datasets were not consistently very large. Thus, the simplest scheme with a single hyperparameter (α) specifying the weight of the classifier loss, i.e., controlling the trade-off between reconstruction loss and classifiability of samples, is likely preferable in practice or at least as a starting point. We also found that α is sensitive to classifier

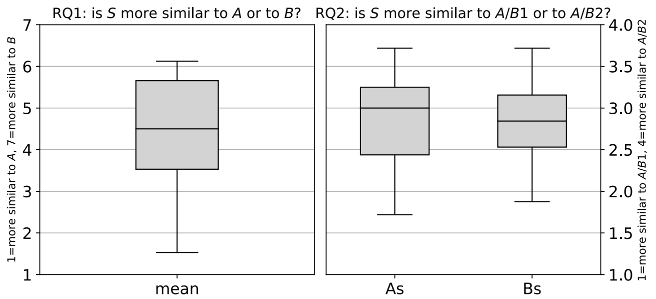

Fig. 7. Results of our user-study. The left plot reflects RQ1, whereas the right plot reflects RQ2. Participants could identify original and target samples significantly better than chance. They were also successfully misled by the AS generated by *Max-Perturb*.

performance, i.e., for poor classifiers with high loss α should be rather small, since otherwise the influence of the classifier on the generated samples is too large. Aside from this, future work might build on our ideas. For example, we argued that fixing $\beta \in \{0.35, 0.5, 0.65\}$ might be too restrictive. Potentially, selecting b in a more flexible way, i.e., using the method Max-perturb and using b over a continuous range might yield better outcomes (although our preliminary tests did not point in this direction).

High resolution images: We thoroughly evaluated our approach on multiple datasets. Still, one might wonder how well it generalizes to other scenarios, e.g., high resolution images. Our approach primarily relies on altering the objective function and is not depending on the architecture of the AE. Thus, we might use a more complex and data-hungry architecture like NVAE [27] and train it with our objective function on ImageNet. However, even in this NIPS paper [27] the authors only trained on 32 × 32 colored images, which took 1700 GPU hours.[9] Training on full ImageNet (256 × 256 × 3) would require more than 100.000 GPU hours. To the best of our knowledge, most architectures are scalable, e.g., a ResNet works well on MNIST and ImageNet.[10] We propose altering the objective, which is in general less sensitive than architectural modifications, e.g., L2-regularization works on essentially any CNN architecture and dataset. Given the current energy crises and European governments calling to save energy, it is debatable to what extent an evaluation that very likely does not yield new insights but would cost a lot of energy should be undertaken. While we have evaluated on multiple datasets and multiple classifiers, we acknowledge that for applications requiring high resolution images the amount of computation can be very large due to the training (and regularization) of generative models like NVAE. Note that in practice, an attacker must also get "data" either for training from scratch or fine-tuning a model. Thus, working with smaller datasets is appealing for an attacker as well.

Limiting interpolations: Put simply, some interpolations might be irrelevant to an attacker. Reducing regularization to AS of a few selected classes might yield better results. One might prioritize the original and target samples, e.g., by choosing them more frequently during training of the AE.

Traditional attacks: Our approach seeks to generate samples that potentially fool both a classifier and a human in a joint optimization process. Alternatively, samples are produced that fool humans and then altered so that a classifier is fooled as well based on traditional adversarial attacks. Using traditional adversarial attacks in combination with other image manipulation techniques has strengths and weaknesses. For example, we acknowledge that our method struggles if the original or target sample stem from low density areas, i.e., few similar points

[8] We randomized: the original/target is shown randomly on the left or right; the correct answer is randomly placed at the first or second position, i.e., $A1$ or $A2$ (and $B1$ or $B2$, respectively).

[9] 70 h on 24 16-GB V100 GPUs as stated in the official repo https://github.com/NVlabs/NVAE.

[10] Also the reviewers at NIPS came to this conclusion when judging and accepting the NVAE paper.

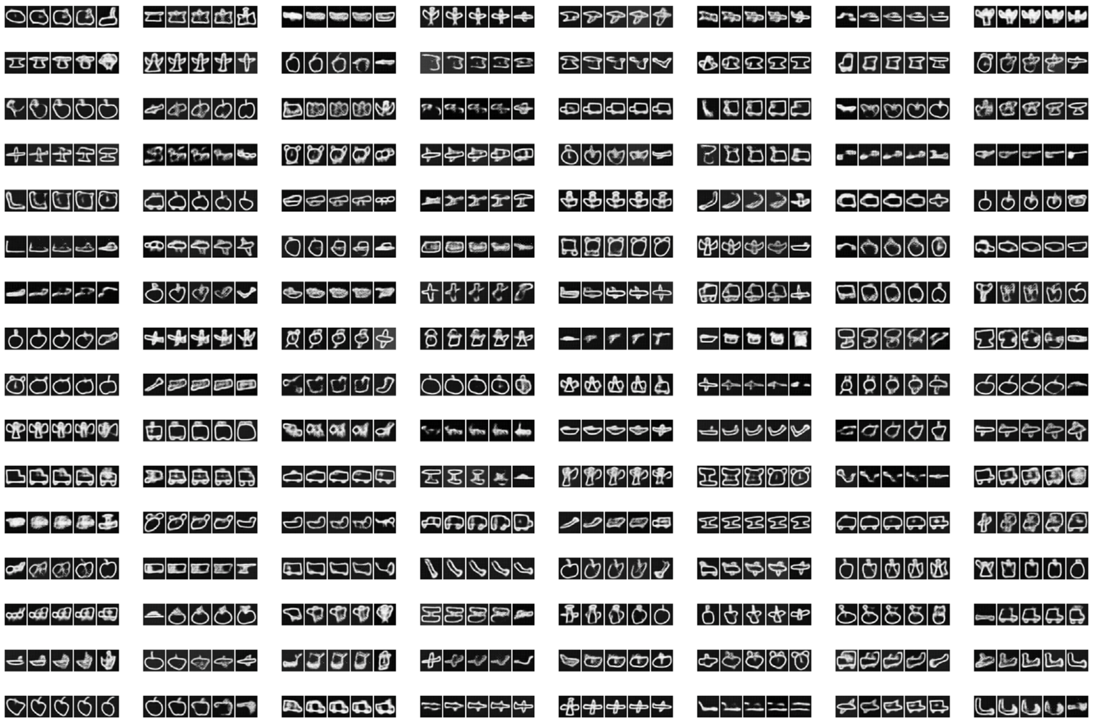
Fig. 8. QuickDraw samples for AE w. O,T,M_3.

exist in the dataset. In such a case an AE is likely unable to produce good samples. A disadvantage of classical attacks is that an image with imperceptible perturbation might be disguised as being unrealistic, i.e., a clear fraud attempt. Our motivation is to ensure samples that might occur in reality or are at least deemed possible. A fraud attempt is a crime (intent to deceive) and puts an attacker at risk. We allow an attacker to control the level of deceit. That is, an attacker can aim for less pronounced changes as in marketing, where an image is altered ranging from minor polishing (i.e., the object might look like X_A under some conditions in reality) to extreme manipulation. As such, the impact is also different ranging from legally ok (though morally unacceptable) to fraud. The attacker can control manipulation strength by adjusting parameter β. The exact choice depends on the attacker's risk appetite. A small b means staying similar to the original, and thus, the adversarial sample has a low risk of being judged as a "criminal" manipulation. Choosing a large b means being close to the target (but more dissimilar to the original) and also having a higher risk of being accused of illegal changes. Finally, our approach is also convenient for an attacker: He/she only has to provide a target image rather than manual image editing. Furthermore, an end-to-end process targeting both the human and the classifier in one joint optimization procedure might lead to better outcomes. To this end, our work is only a first step.

Other security concerns: We assess transferability assuming black-box access to the attacked model. Black-box access allows generating AS, e.g., using surrogate models [13]. Thus, it is a known security risk, and naive, frequent access can be risky for an attacker since every access might leave a digital trace. In future work, one can minimize the number of black box accesses. From a security standpoint, any *offense* can be turned into a *defense*; in our case, by using the well-known *adversarial training* technique [46,47]. Many papers propose to first attack an ML model, and then use the successful attacks to harden such an ML model (e.g., [48]). Such property also allows our method to be applicable as a defense: in such cases, the 'white-box' assumption is relaxed because the defender has full control of their own ML system. Finally, we observe that the underlying principles of our method are agnostic of the data-type, and hence applicable also to DL applications for other domains (e.g., malware detection [49]).

8. Conclusions

Secure AI is a critical factor for AI's successful and beneficial deployment. We analyzed potential security issues that can arise when decision-making is a collaborative process involving humans and deep learning systems. Such hybrid processes are likely to flourish as AI adoption increases. Our adversarial samples stemming from carefully crafted interpolation of regularized latent spaces often exhibit more or less recognizable differences to a given original sample while simultaneously showing similarities to a target sample—as confirmed by our user study and other evaluation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Appendix. Quickdraw (qualitative results)

We report in Figs. 8–13 the images generated on QuickDraw and CIFAR-10. For CIFAR-10 results are less visually appealing across all methods which could be overcome with a much larger computation budget paired with a much larger dataset as elaborated in our discussion. For CIFAR-10 ACAI sometimes fails to reconstruct meaningful samples due to the GAN architecture, which might not always successfully train. AE w. O,T,M_3 tends to lead to somewhat crispier images than Vanilla AE on CIFAR-10.

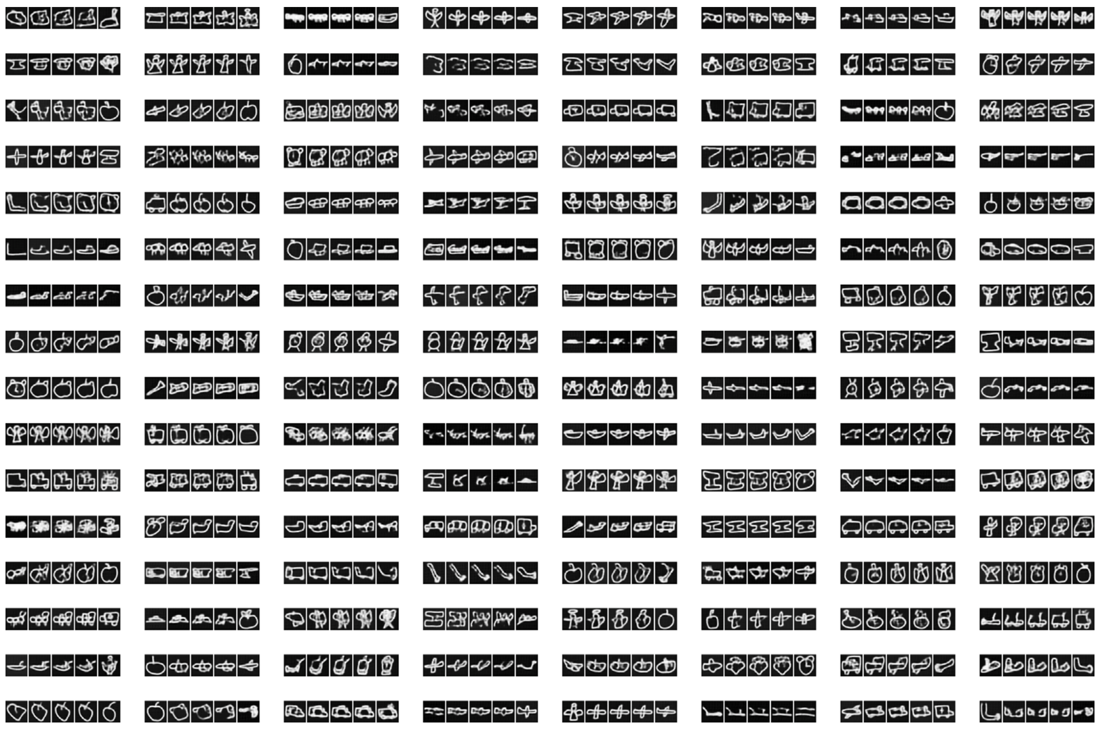
Fig. 9. QuickDraw samples for Vanilla AE.

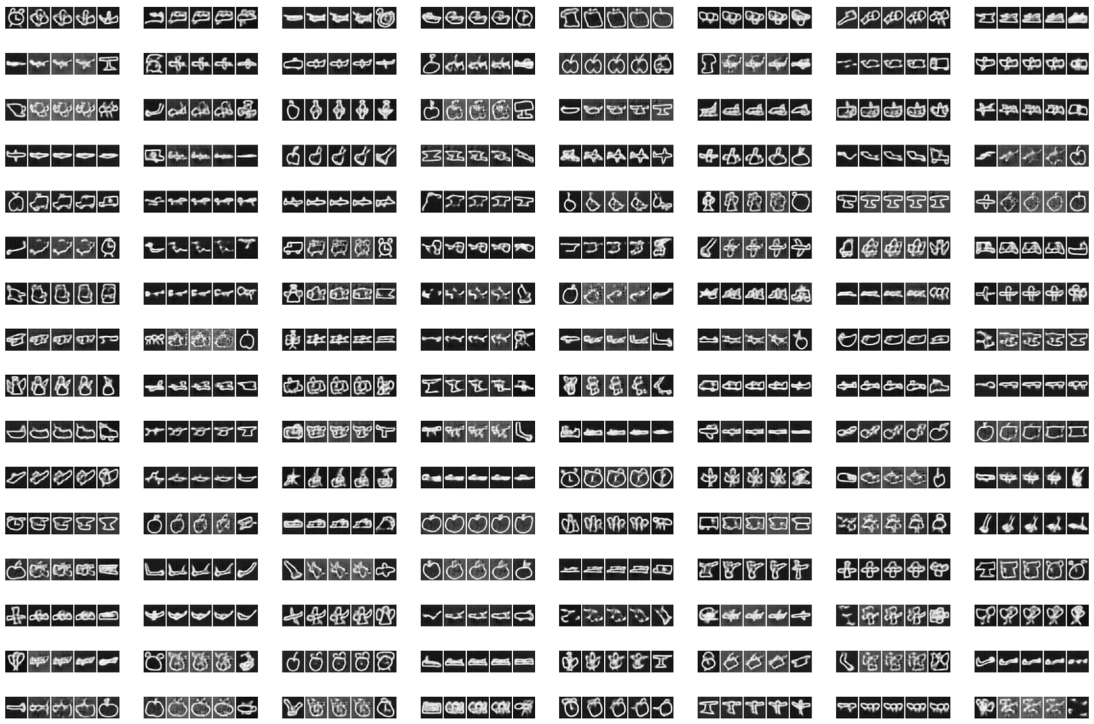
Fig. 10. QuickDraw samples for ACAI.

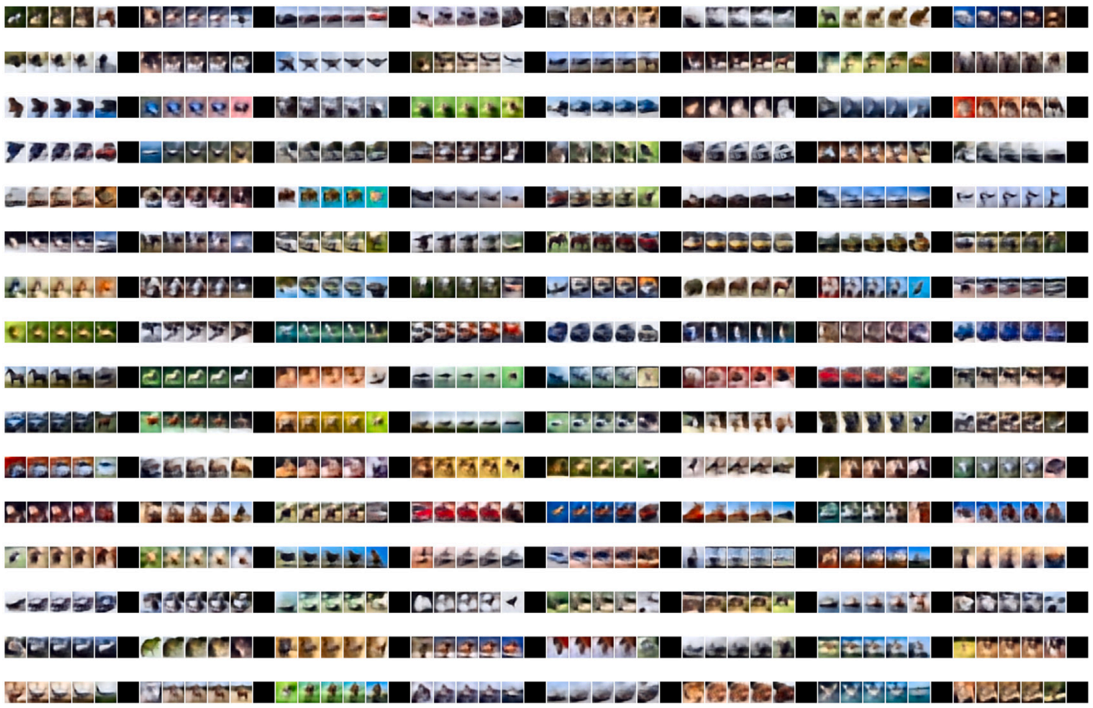

Fig. 11. CIFAR-10 samples for AE w. O,T,M_3.

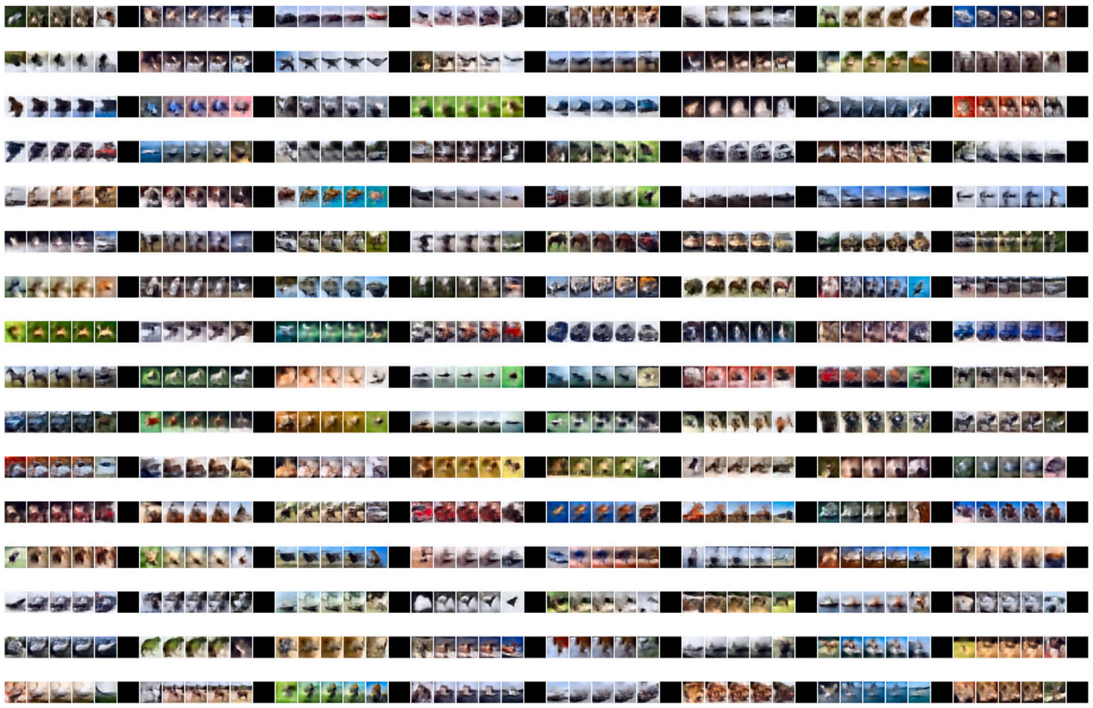

Fig. 12. CIFAR-10 samples for Vanilla AE.

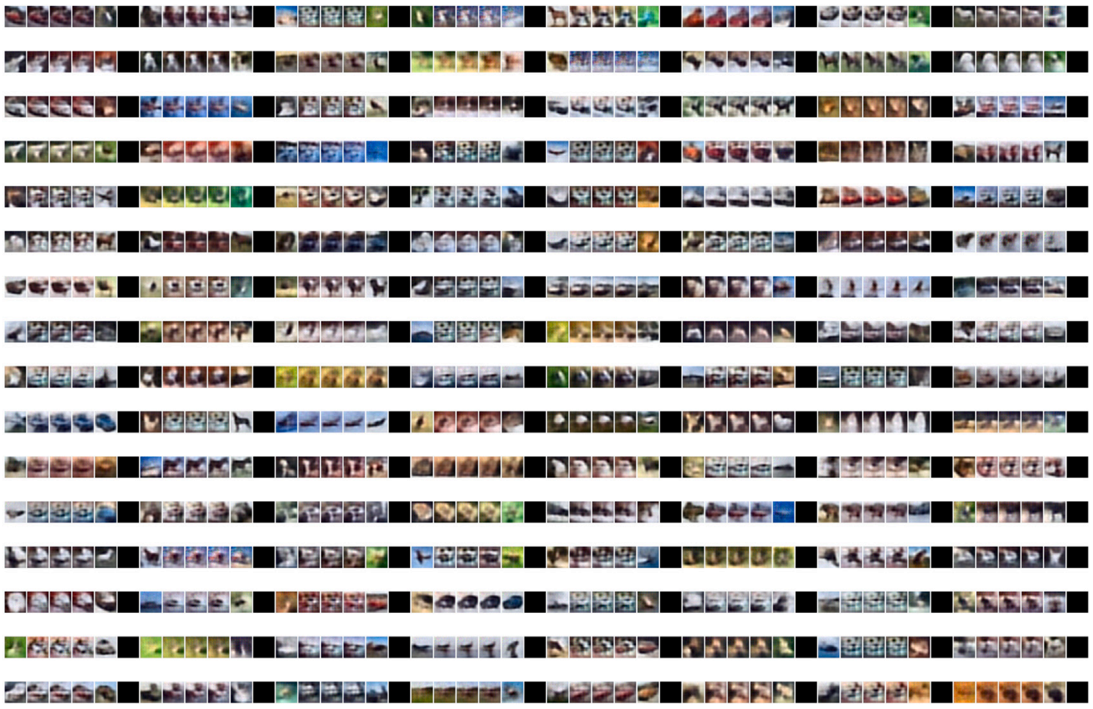
Fig. 13. CIFAR-10 samples for ACAI.

References

[1] Di W, Sundaresan N, Piramuthu R, Bhardwaj A. Is a picture really worth a thousand words? -on the role of images in e-commerce. In: Proceedings of the 7th ACM International conference on web search and data mining. 2014, p. 633–42.

[2] How Listing Photos Impact Marketplace Performance. 2023, https://claid.ai/blog/article/photos-impact-conversions/.

[3] Papernot N, McDaniel P, Sinha A, Wellman MP. SoK: Security and Privacy in Machine Learning. In: Proc. IEEE European symposium on security and privacy. 2018.

[4] Xu H, Ma Y, Liu H-C, Deb D, Liu H, Tang J-L, Jain AK. Adversarial attacks and defenses in images, graphs and text: A review. Int J Autom Comput 2020;17(2):151–78.

[5] Akhtar N, Mian A, Kardan N, Shah M. Advances in adversarial attacks and defenses in computer vision: A survey. IEEE Access 2021;9:155161–96.

[6] Serban A, Poll E, Visser J. Adversarial examples on object recognition: A comprehensive survey. ACM Comput Surv 2020;53(3):1–38.

[7] Schneider J, Apruzzese G. Concept-based adversarial attacks: Tricking humans and classifiers alike. In: 5th Deep learning and security workshop @ security and privacy (S&P). 2022.

[8] Elsayed G, Shankar S, Cheung B, Papernot N, Kurakin A, Goodfellow I, Sohl-Dickstein J. Adversarial examples that fool both computer vision and time-limited humans. Adv Neural Inf Process Syst 2018.

[9] Alemanno A, Spina A. Nudging legally: On the checks and balances of behavioral regulation. Int J Const Law 2014;12(2):429–56.

[10] Berthelot D, Raffel C, Roy A, Goodfellow I. Understanding and improving interpolation in autoencoders via an adversarial regularizer. 2018, arXiv preprint arXiv:1807.07543.

[11] YOLOv3: Real-time object detection algorithm (guide). 2022, URL https://viso.ai/deep-learning/yolov3-overview/.

[12] OpenPilot. 2022, https://github.com/commaai/openpilot.

[13] Demontis A, Melis M, Pintor M, Jagielski M, Biggio B, Oprea A, Nita-Rotaru C, Roli F. Why do adversarial attacks transfer? Explaining transferability of evasion and poisoning attacks. In: Proc. USENIX Security symposium. 2019, p. 321–38.

[14] Biggio B, Roli F. Wild patterns: Ten years after the rise of adversarial machine learning. Elsevier Pattern Recognit 2018;84:317–31.

[15] Apruzzese G, Anderson H, Dambra S, Freeman D, Pierazzi F, Roundy K. Position:“real attackers don’t compute gradients”: Bridging the gap between adversarial ML research and practice. In: IEEE Conference on secure and trustworthy machine learning. IEEE; 2023.

[16] Pierazzi F, Pendlebury F, Cortellazzi J, Cavallaro L. Intriguing properties of adversarial ml attacks in the problem space. In: Proc. IEEE Symposium on security and privacy. SP, 2020, p. 1332–49.

[17] Su J, Vargas DV, Sakurai K, et al. One pixel attack for fooling deep neural networks. IEEE Trans Evol Comput 2019;23(5):828–41.

[18] Wu Z, Lim S-N, Davis LS, Goldstein T. Making an invisibility cloak: Real world adversarial attacks on object detectors. In: European Conf. on computer vision. 2020.

[19] Pajola L, Conti M. Fall of giants: How popular text-based mlaas fall against a simple evasion attack. In: 2021 IEEE European symposium on security and privacy (EuroS&P). IEEE; 2021, p. 198–211.

[20] Funke CM, Borowski J, Stosio K, Brendel W, Wallis TS, Bethge M. Five points to check when comparing visual perception in humans and machines. J Vis 2021;21(3):16.

[21] Zhou Z, Firestone C. Humans can decipher adversarial images. Nature Commun 2019;10(1):1–9.

[22] Harrington A, Deza A. Finding biological plausibility for adversarially robust features via metameric tasks. In: SVRHM 2021 Workshop@ NeurIPS. 2021.

[23] Harding S, Rajivan P, Bertenthal B, Gonzalez C. Human decisions on targeted and non-targeted adversarial samples. In: Annual conference of the cognitive science society. 2018.

[24] Salamati M, Soudjani S, Majumdar R. Perception-in-the-loop adversarial examples. 2019, arXiv preprint arXiv:1901.06834.

[25] Mirsky Y, Mahler T, Shelef I, Elovici Y. {Ct-gAN}: Malicious tampering of 3D medical imagery using deep learning. In: USENIX Security symposium. 2019.

[26] Wang Z, She Q, Ward TE. Generative adversarial networks in computer vision: A survey and taxonomy. ACM Comput Surv 2021;54(2):1–38.

[27] Vahdat A, Kautz J. NVAE: A deep hierarchical variational autoencoder. Adv Neural Inf Process Syst 2020;33:19667–79.

[28] Sauer A, Schwarz K, Geiger A. Stylegan-xl: Scaling stylegan to large diverse datasets. In: Special interest group on computer graphics and interactive techniques conference proceedings. 2022, p. 1–10.

[29] Meske C, Bunde E, Schneider J, Gersch M. Explainable artificial intelligence: objectives, stakeholders, and future research opportunities. Inf Syst Manage 2022;39(1):53–63.

[30] van Doorenmalen J, Menkovski V. Evaluation of CNN performance in semantically relevant latent spaces. In: Int. symposium on intelligent data analysis. 2020.

[31] Sainburg T, Thielk M, Theilman B, Migliori B, Gentner T. Generative adversarial interpolative autoencoding: adversarial training on latent space interpolations encourage convex latent distributions. 2018, arXiv preprint arXiv:1807.06650.

[32] Fadel SG, Mair S, S Torres Rd, Brefeld U. Principled interpolation in normalizing flows. In: Joint European conference on machine learning and knowledge discovery in databases. 2021, p. 116–31.

[33] Schneider J. Human-to-ai coach: Improving human inputs to ai systems. In: International symposium on intelligent data analysis. 2020.

[34] Gatys LA, Ecker AS, Bethge M. A neural algorithm of artistic style. 2015, arXiv preprint arXiv:1508.06576.

[35] Jing Y, Yang Y, Feng Z, Ye J, Yu Y, Song M. Neural style transfer: A review. IEEE Trans Vis Comput Graphics 2019;26(11):3365–85.

[36] Baldi P, Hornik K. Neural networks and principal component analysis: Learning from examples without local minima. Neural Netw 1989;2(1):53–8.
[37] Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. Adv Neural Inf Process Syst 2017;30.
[38] Basalla M, Schneider J, vom Brocke J. Creativity of deep learning: Conceptualization and assessment. In: 14th Int. conf. on agents and artificial intelligence. 2022.
[39] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on computer vision and pattern recognition. 2016.
[40] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 2017, arXiv preprint arXiv:1704.04861.
[41] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In: Int. conference on learning representations (ICLR). 2014.
[42] Schneider J, Vlachos M. Explaining classifiers by constructing familiar concepts. Mach Learn 2022.
[43] Xiao H, Rasul K, Vollgraf R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. 2017, arXiv preprint arXiv:1708.07747.
[44] Deng L. The mnist database of handwritten digit images for machine learning research. IEEE Signal Process Mag 2012;29(6).
[45] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images. Tech. Rep., 2009.
[46] Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, Lempitsky V. Domain-adversarial training of neural networks. J Mach Learn Res 2016;17(1).
[47] Shafahi A, Najibi M, Ghiasi A, Xu Z, Dickerson J, Studer C, Davis LS, Taylor G, Goldstein T. Adversarial training for free!. In: Proc. international conference on neural information processing systems. 2019, p. 3358–69.
[48] Apruzzese G, Andreolini M, Marchetti M, Venturi A, Colajanni M. Deep reinforcement adversarial learning against botnet evasion attacks. IEEE Trans Netw Serv Manag 2020;17(4):1975–87.
[49] Kolosnjaji B, Demontis A, Biggio B, Maiorca D, Giacinto G, Eckert C, Roli F. Adversarial malware binaries: Evading deep learning for malware detection in executables. In: European signal processing conference. EUSIPCO, 2018, p. 533–7.