# Self-Supervised Adversarial Example Detection by Disentangled Representation

Zhaoxi Zhang[1], Leo Yu Zhang[2], Xufei Zheng[1]✉, Jinyu Tian[3], Jiantao Zhou[3]
[1]School of Computer and Information Science, Southwest University, China
[2]School of Information Technology, Deakin University, Australia
[3]Department of Computer and Information Science, University of Macau, Macau
✉ Corresponding Author: X. Zheng (zxufei@swu.edu.cn)

***Abstract*—Deep learning models are known to be vulnerable to adversarial examples that are elaborately designed for malicious purposes and are imperceptible to the human perceptual system. Autoencoder, when trained solely over benign examples, has been widely used for (self-supervised) adversarial detection based on the assumption that adversarial examples yield larger reconstruction errors. However, because lacking adversarial examples in its training and the too strong generalization ability of autoencoder, this assumption does not always hold true in practice. To alleviate this problem, we explore how to detect adversarial examples with disentangled label/semantic features under the autoencoder structure. Specifically, we propose Disentangled Representation-based Reconstruction (DRR). In DRR, we train an autoencoder over both correctly paired label/semantic features and incorrectly paired label/semantic features to reconstruct benign and counterexamples. This mimics the behavior of adversarial examples and can reduce the unnecessary generalization ability of autoencoder. We compare our method with the state-of-the-art self-supervised detection methods under different adversarial attacks and different victim models, and it exhibits better performance in various metrics (area under the ROC curve, true positive rate, and true negative rate) for most attack settings. Though DRR is initially designed for visual tasks only, we demonstrate that it can be easily extended for natural language tasks as well. Notably, different from other autoencoder-based detectors, our method can provide resistance to the adaptive adversary.**

## I. Introduction

In 2013, the seminal work [1] reported that, during model test time, deep neural networks can be easily fooled by adversarial attacks that add tiny perturbations to inputs. Since then, adversarial attacks and defenses have drawn significant research attention [2]–[11]. On the one hand, attackers are persistently developing new strategies to construct adversarial examples; on the other hand, defenders are struggling to cope with all existing and forthcoming attacks [12].

Most of the existing defense methods [6]–[8], [13], [14] are trained with supervision, and these methods work well when defending against adversarial attacks they were originally trained for. However, it is widely regarded that supervised methods cannot generalize well to adversarial examples from (existing) unseen attacks, let alone examples from new attacks.

Self-supervised based defense, in comparison with supervised defense, requires only benign examples for its training. As a typical example, the works in [15], [16] utilize the encoder of an autoencoder (AE) to draw the manifold of benign examples and then the decoder network for reconstruction, as shown in Fig. 1(a). Since the manifold is learnt from benign examples only and the AE is trained to minimize Reconstruction Errors (REs) for benign examples, thus the encoding of adversarial examples will likely be out-of-distribution and the associated REs will be larger.

It is soon realized that this is not always true because AE has very strong generalization ability [17]: examples with various kinds of small perturbations can be reconstructed with small RE. This is desirable if the perturbed examples are benign. However, adversarial examples are just specific perturbed versions of benign examples, and the malicious perturbations can also be made very small in many attacks (i.e., now their encodings will reside in the light gray area of Fig. 1(a)). When this happens, all REs are mixed, and it leads to a high false negative or false positive rate (FNR/FPR) during detection. To refine the volume of the manifold drawn by AE and reduce its unnecessary generalization ability on adversarial examples, there exist a number of variants [18]–[22], as will be reviewed in detail in Sec. II-B.

As a better solution, we propose a self-supervised disentangled representation-based reconstruction (DRR) method to detect adversarial examples. DRR possesses the advantage of supervised defense, even though it does not have access to any adversarial examples in training. This is achieved through mimicking the behavior of adversarial examples by encoding and decoding a special class of examples (counterexamples in this work), which is the reconstruction of the correct semantic feature and the incorrect label feature from one example. The rationale is based on the very fact of adversarial examples: **they cause misclassification (i.e., wrong label) without changing semantics (contained in its benign counterparts).**

With this observation, we make use of the victim model and an auxiliary encoder to obtain disentangled representations: class-dependent and class-independent features (i.e., label feature and semantic feature), of images. After disentanglement, we then train a decoder to reconstruct benign examples by combining label/semantic features from the same benign image, as well as counterexamples by combining the original semantic feature and the post-processed incorrect label feature. For detection, as shown in Fig. 1(b), DRR will faithfully disentangle any benign image into paired label/semantic features and its adversarial counterpart into unpaired label/semantic features, whose associated REs are significantly different as desired.

This paper makes the following contributions:

- We use disentangled representation for adversarial detection, which makes it possible for the detector to mimic the behavior of adversarial examples in the self-supervised framework.
- We design DRR via an AE structure, but it reduces the unnecessary generalization capability of AE on adversarial examples.
- We achieve state-of-the-art adversarial detection performance on MNIST, SVHN, CIFAR-10 and CIFAR-100 in most cases. Specifically, DRR is the first of its kind that can offer protection against adaptive adversarial attacks. And we also demonstrate that DRR is effective for defending against adversarial textual attacks.

## II. Background and Related Works

### A. Constructing Adversarial Examples

The existence of adversarial examples in deep neural networks is first pointed out by [1], who find maliciously designed imperceptible perturbations can fool deep models to misclassify. Let $\mathsf{F}(\cdot)$ be a general neural model and $\mathsf{F}_z(\cdot)$ be the layers before $\mathrm{softmax}$ of the model, then evaluating the test example $x$ is simply a $\mathrm{softmax}$ classification over the logits $z = \mathsf{F}_z(x)$, i.e., $y = \mathsf{F}(x) = \mathrm{softmax}(z)$. The (untargeted) adversarial example $x_{\text{adv}}$ derived from $x$ satisfies

$$\begin{aligned} &\mathsf{F}(x_{\text{adv}}) \neq \mathsf{F}(x), \\ &x_{\text{adv}} = x + \delta_{\text{adv}} \text{ and } \|\delta_{\text{adv}}\| < \epsilon, \end{aligned} \tag{1}$$

where $\delta_{\text{adv}}$ is the adversarial perturbation, $\epsilon$ is the maximum magnitude of $\delta_{\text{adv}}$ under certain norm compliance. Commonly used norms include $L_2$ and $L_{\text{inf}}$, and we also focus on detecting adversarial examples bounded by them.

In the current literature, the Fast Gradient Sign Method (FGSM) is the first widely used method in generating adversarial examples [2]. Projected Gradient Descent (PGD) [5] improves FGSM by iterating the building block of it according to different criteria to find the optimized perturbation. DeepFool [3] and CW [4] do not directly rely on gradient but instead optimize (the norm of) $\delta_{\text{adv}}$, which is usually more stealthy than FGSM and its variants.

Another line of research for constructing adversarial examples is called adaptive attacks [23]. Under an adaptive attack, the attacker also has white-box access to possible defense mechanisms, and his goal is to find an optimal $\delta_{\text{adv}}$ that solves Eq. (1) and circumvents the defense mechanisms simultaneously. All the attacks above can have their adaptive versions by considering different defenses.

### B. Detecting Adversarial Examples

Adversarial detection is an effective way to prevent adversarial examples, and it can be classified as supervised and unsupervised methods, considering whether adversarial examples are needed to build a detector. For supervised

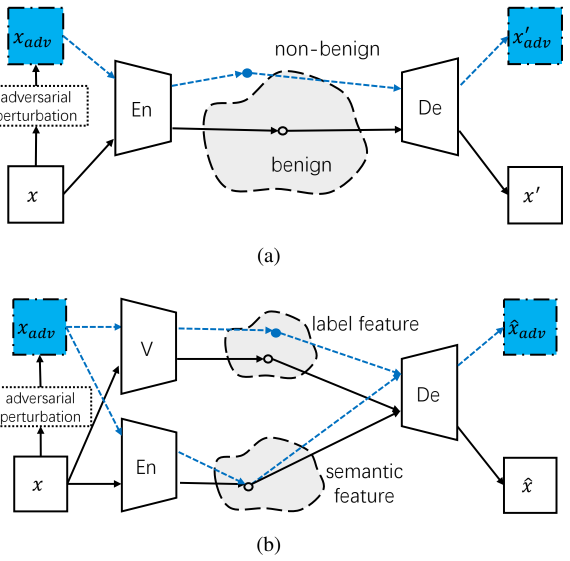


Fig. 1: Different detectors. Here, En stands for the encoder, De the decoder, V the victim model. (a) AE based method draws a manifold (the light gray area) of benign examples; (b) DRR employs two disentangled features: label features which represent images' labels (vulnerable to adversarial perturbation) and semantic features (robust to adversarial perturbation). When decoding label/semantic features, benign examples can be reconstructed faithfully but adversarial examples cannot.

detectors, their detecting capability depends on how to capture the differences between adversarial and benign examples. Techniques range from studying statistical properties [6], [7], [14], training traditional machine learning classifiers [24], [25] and deep classifiers [13], [19], [26]. It is widely accepted that supervised methods cannot generalize well to adversarial examples produced by unseen attacks. For example, a supervised detector trained with PGD adversarial examples will likely fail to detect examples produced by DeepFool. However, it is also known that supervised detectors are robust to adaptive attacks, i.e., a detector trained with PGD can detect examples produced by the adaptive PGD attack [13], [26].

For unsupervised detectors, their detecting capability depends on how to embed and represent benign examples to a different manifold (other than the natural spatiotemporal domain) such that adversarial perturbations will be magnified with embedding (without even seeing them at all). The seminal work MagNet [15] uses an autoencoder to draw the manifold for benign examples, which implicitly assumes that the distance between the adversarial and benign examples is large in this embedding manifold.

However, the embedding manifold induced by AE is not always desirable for detection since the autoencoder has a too strong generalization capability [17]. Further to MagNet, the work [18] trains a variant of the autoencoder by adding logits of the victim model into the loss function to refine the volume of the embedded manifold. The work [19] proposes to directly use parameters fixed victim model as the encoder, and the victim models' logits as high-level representations/embedding.

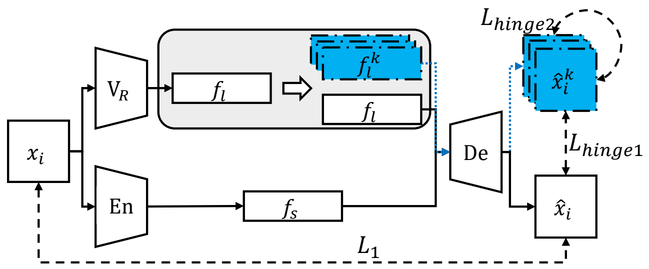


Fig. 2: Overview of DRR.

Different from the works above, which treat the manifold of all benign examples as a whole, [20]–[22] propose a class-conditional model to embed and reconstruct benign examples for even better refining the (embedding) manifold. However, it is reported in [20] that this method still only reacts to relatively large (adversarial) perturbations over simple datasets. Compared with supervised detectors, self-supervised detectors may generalize better to unseen attacks but they are vulnerable to adaptive attacks [15], [17], [19]. For example, it is shown in [27] that the detector of [15] fails to resist adaptive attacks at all.

### *C. Disentangled Representation*

Disentangled representation is firstly advocated by InfoGAN in [28], which encourages the learning of interpretable and meaningful representations of inputs for manipulating specific features. The work [29] disentangles speaker-related representation to synthesis voices of different speakers and the work [30] uses disentangled representation to address the pose discrepancy problem among face images.

From the view of information, the training of a neural model is to minimize the mutual information of input and output [31], so semantic (contained in the input) and label (contained in the output) features of an image sample are made independent via training and they are disentangled by nature in any neural model. From the view of adversarial learning, it is acknowledged that there are many features in an image, and not all features are equally easy to be manipulated in adversarial attack [32]. Specifically, the class an image belongs to (i.e., the label), which does not depend on semantic feature it has, is a concrete example. In this concern, we employ the disentangled semantic feature and label feature to build DRR. Specifically, semantic feature can be easily manipulated by the attackers (through adversarial perturbation) but the perturbed semantic feature is similar to the original version (even after embedding if the perturbation is tiny). In contrast, label feature cannot be directly manipulated by attackers, but they are fragile to adversarial perturbation and robust to natural perturbation of the semantic feature.

## III. DRR for Adversarial Detection

### *A. Training of DRR*

From a high-level-of-view, as depicted in Fig. 2, assisted by the parameter-fixed victim model $\mathsf{V}$, DRR first extracts label and semantic features from benign examples $x_i$ to train the encoder $\mathsf{En}$ and decoder $\mathsf{De}$. Then DRR permutes the label feature of $x_i$ and combines it with the semantic feature of $x_i$ to decode counterexamples. The details of how to design the loss functions of $\mathsf{En}$ and $\mathsf{De}$ of DRR are presented in detail below.

First and foremost, we examine how to obtain the desired semantic feature $f_s$ and label feature $f_l$ of $x_i$ for detection purpose. For $f_l$, we choose to rescale the logits the victim $\mathsf{V}$, i.e.,

$$y = \mathsf{V}(x_i) = \mathrm{softmax}(\mathsf{V}_z(x_i)), \tag{2}$$
$$f_l = \mathsf{V}_R(x_i) = R_S(\mathsf{V}_z(x_i)), \tag{3}$$

where $\mathsf{V}_R(x_i)$ denotes the rescaled logits and the rescaling function is given by

$$R_S(x) = \mathrm{softmax}(x/T), \tag{4}$$

where $T \geq 1$ is a temperature parameter. The choice of the label feature $f_l$ is two-folded: 1) logits is very close to the final one-hot encoding of $\mathsf{V}(x_i)$ and resacling with a temperature $T$ is commonly used for information distillation [33]; 2) no matter how the concrete adversarial example $x_{\text{adv}}$ is constructed, $\mathsf{V}_R(x_{\text{adv}})$ must be different enough from $\mathsf{V}_R(x)$ to induce the final erroneous classification.

For the semantic feature $f_s$, we use a general encoder network $\mathsf{En}$ to derive it, i.e., $f_s = \mathsf{En}(x_i)$. This is because it is widely accepted that high-dimensional input $x_i$ resides in a low-dimensional manifold, and it is a commonly used method in the area of representation learning.

With $f_l$ and $f_s$ available, the decoder network $\mathsf{De}$ of DRR reconstructs $x_i$ as $\hat{x}_i = \mathsf{De}(f_s, f_l)$. The natural requirement for $\mathsf{En}$ and $\mathsf{De}$ is that, for a generic image $x_i$ from the benign dataset, the reconstructed version $\hat{x}_i$ should be similar to $x_i$. Thus, the associated loss is:

$$\mathsf{L}_1 = \mathbb{E}_{x_i}\mathsf{MAE}(x_i, \hat{x}_i), \tag{5}$$

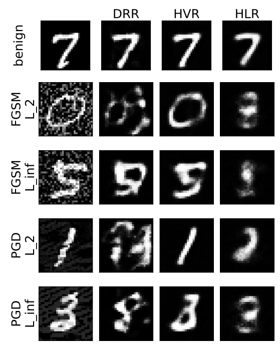
where $\mathsf{MAE}$ is the mean absolute error and $\mathbb{E}$ the expectation.

We then move to the training of DRR with counterexamples by using unpaired semantic and label features, as depicted by the dashed blue box in Fig. 2. As emphasized in Sec. II-B, the drawback of AE based detection is AE generalizes too well and the refinement of the manifold drawn by AE is not always effective, especially on attacks that directly optimize the norm of the adversarial perturbation. The solution is now straightforward since we can mimic the behavior of adversarial examples by constructing counterexamples from unpaired semantic feature and label feature to better refine the manifold drawn by $\mathsf{En}$ and $\mathsf{De}$.

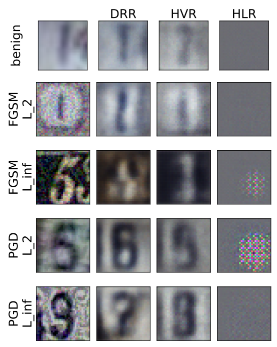
To obtain counterexamples, we first randomly relocate the largest element in $f_l$ from index $i$ to $k$, i.e.,

$$f_{l_i}^k = \mathsf{R}_L(f_{l_i}, k), k \neq i. \tag{6}$$

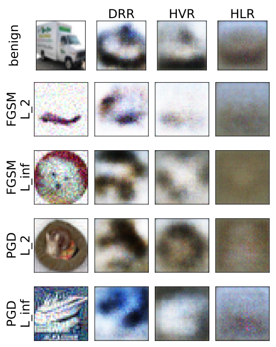
We then decode the permuted label feature $f_{l_i}^k$ and the original semantic feature $f_{s_i}$ to obtain counterexamples by $\hat{x}_i^k = \mathsf{De}\left(f_{s_i}, f_{l_i}^k\right)$.



(a) MNIST



(b) SVHN



(c) CIFAR10

Fig. 3: Reconstructed images of each AE-based detector.

Recall that our original purpose is to refine the manifold drawn by the encoder and decoder, so the loss function here should satisfy the following two requirements:

- The decoded $\hat{x}_i^k$ should not converge to $\hat{x}_i$ (and thus $x_i$) because convergence indicates that counterexamples and benign examples are mixed, which is contrary to our original detection purpose;
- The decoded $\hat{x}_i^k$ should not be far away from $\hat{x}_i$, since too-far-away implies out-of-distribution and will cause overfitting.

Note that $\hat{x}_i^{k_1}$ and $\hat{x}_i^{k_2}$ ($k_1 \neq k_2$) are different counterexamples of $x_i$ and they should also obey the above requirements. For these reasons, we propose to use a soft hinge function as the loss, i.e.,

$$\mathrm{L}_{hinge1} = \mathbb{E}_{x_i}\left[\max\left(0, d - \min_{\substack{k_1,k_2\in\Sigma,\\ k_1\neq k_2}}\left(\mathsf{MAE}(\hat{x}_i^{k_1}, \hat{x}_i^{k_2})\right)\right)\right], \tag{7}$$

$$\mathrm{L}_{hinge2} = \mathbb{E}_{x_i}\left[\max\left(0, d - \min_{\substack{k\in\Sigma,\\ k\neq i}}\left(\mathsf{MAE}(\hat{x}_i^{k}, x_i)\right)\right)\right], \tag{8}$$

$$\mathrm{L}_{hinge} = \mathrm{L}_{hinge1} + \mathrm{L}_{hinge2}, \tag{9}$$

where $d$ is a hyperparameter used to control the farthest allowable distance between counter and benign examples, the set $\Sigma$ determines the number of counterexamples used for $x_i$. This soft hinge function (passively) meets the second requirements list above: its gradient equals $0$ when $\mathsf{MAE}(\hat{x}_i^k, \hat{x}_i) > d$, so when the loss meets its upper bound, it will not contribute gradient to the training process. The size of the set $\Sigma$ is also an important hyperparameter because a hard counterexample contributes more to the final performance than an easy counterexample. When the size of $\Sigma$ is too small, the chance of sampling hard counterexamples is low each time. But when the size of $\Sigma$ is too large, it will cause loss function very hard to converge.

In summary, the final loss function to train the encoder $\mathsf{En}$, and the decoder $\mathsf{De}$ of DRR is

$$\mathrm{Loss} = \lambda \mathrm{L}_1 + \mathrm{L}_{hinge}, \tag{10}$$

where $\lambda$ is used to control the relative importance.

## B. Detecting with DRR

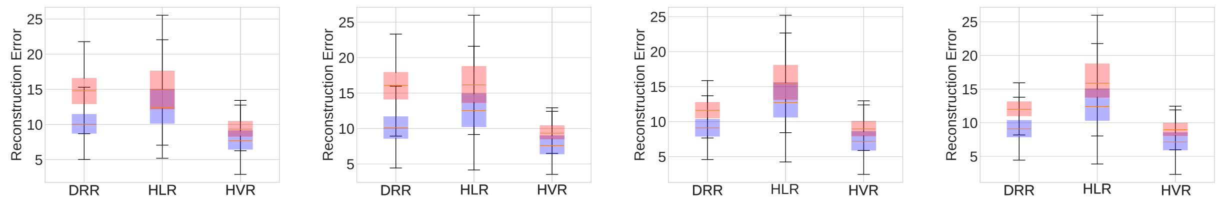
For detection, as discussed earlier and shown in Fig. 1(b), an incoming test example $x$ will be encoded and decoded as $\hat{x}$, then the RE $\|\hat{x} - x\|_2$ is compared to a threshold value. If the RE is larger, then it is considered as adversarial; otherwise, it is not. Ideally, this threshold should be universal (on a given dataset) for all possible attacks (even for new attacks). However, for existing detectors, the threshold value is related to the used attack methods. This makes it hard to determine a universal threshold for existing detectors and to compare with them. For this reason, we use the metrics area under the ROC curve (AUC), true positive rate (TPR) and true negative rate (TNR) for evaluation in Sec. IV. But it is worth mentioning that it is easy to set a universal threshold for DRR for all the considered attacks.

## IV. Experimental Results

In this section, we assess the performance of DRR on 4 datasets, MNIST, SVHN and CIFAR-10/CIFAR-100, by comparing it with other state-of-the-art self-supervised detectors [15], [18], [19], [22] against the adversarial attacks FGSM, PGD, DeepFool (taken from Foolbox [34]) and the adaptive PGD under diffent norms ($L_2$ and $L_{\inf}$). As a proof-of-concept, two representative networks, an 8-layer CNN and VGG-16 [35], are used as the victim models.

## A. Settings

**DRR**: DRR consists of the encoder $\mathsf{En}$ and the decoder $\mathsf{De}$. It is suffice to say that $\mathsf{En}$ has the same architecture as the victim model except that its last layer is fully connected and contains $64$ neurons. The architecture of the decoder $\mathsf{De}$ is simply a mirror of $\mathsf{En}$. Following the literature [13], [26], the hyperparameter $\lambda$ of the Loss are determined by a binary search in $[10^{-2}, 10^{2}]$ (3, 5, 4, 3 for MNIST, SVHN, CIFAR-10, CIFAR-100). The hyperparameter $d$ is determined empirically as $0.16, 0.16, 0.2, 0.14$ for MNIST, SVHN, CIFAR-10, CIFAR-100, respectively. The size of the set $\Sigma$ is set to $20$ for CIFAR100, and $8$ for the other datasets.



(a) FGSM $L_{\text{inf}}$ (b) PGD $L_{\text{inf}}$ (c) FGSM $L_{\text{inf}}$ (d) PGD $L_{\text{inf}}$

Fig. 4: Box-plot of the reconstruction errors: (a, b) CIFAR-10, and (c, d) CIFAR-100 (red box is for adversarial and blue box for benign).

**Methods for comparison**: As mentioned above, three kinds of self-supervised detectors [15], [18], [19], [22] are used for comparison. The structure of the encoder, the decoder for all these three methods are the same as of DRR. These methods can be classified as hidden vector based reconstruction (HVR) and high-level representation based reconstruction (HLR). For case of HVR, the works in [15], [18], [22] are based on optimizing the pixel-level RE from the hidden vector outputted by the encoder. For case of HLR, the work in [19] takes the logits from the parameter-fixed victim model as a high-level representation of examples for decoding.

## *B. Results Analyses*

**The baseline results**: The victim VGG-16 is trained on SVHN, CIFAR-10, CIFAR-100 and the victim 8-layer CNN is trained on MNIST, these 4 victim models achieve 0.961, 0.869, 0.682, 0.993 accuracy over benign test examples. For each test example, we generate its corresponding adversarial versions with 3 attacks (FGSM, PGD, DeepFool) under $L_2$ and $L_{\text{inf}}$ norm compliance ($\epsilon$ in Eq. (1)). Following the literature works [15], [18], [19], for FGSM and PGD attacks on MNIST, we set the budget for $L_{\text{inf}}$ as 0.5 and 0.15, and the budget for $L_2$ as 10 and 2. For SVHN and CIFAR-100, we set the budget for $L_{\text{inf}}$ as 0.15 and 0.15, and the budget for $L_2$ as 5 and 2. For CIFAR-10, we set the budget for $L_{\text{inf}}$ as 0.15 and 0.15, and the budget for $L_2$ as 5 and 1. Note that DeepFool directly optimizes adversarial perturbation, so no budget setting is needed. After attack, the accuracy of 4 victim models over adversarial examples all lower than 0.1.

**Visual inspection**: We first visually compare the reconstruction of the benign and adversarial examples for all the AE-based detectors, some examples are shown in Fig. 3. Inspecting cols. 2 and 4 of Fig. 3, it is clear that DRR and HLR generally have bigger REs over adversarial examples, which validates that logits (of the victim model) reduce undesirable generalization capability of the detectors.

We further box-plot the REs of all benign and adversarial examples from CIFAR-10 and CIFAR-100 in Fig. 4. It is clear from this figure, for the attacks on CIFAR-10 and CIFAR-100, the REs of DRR are clearly separable. The REs of HVR is also generally separable, but it has a higher FNR or FPR than that of DRR (depending on the concrete threshold of RE). And HLR performs the worst among all settings. This validates that DRR not only prevents undesirable generalization (large REs over adversarial examples), but also retains the merits of HVR-based methods (small REs over benign examples).

**AUC and TPR**: We quantitatively study the effectiveness of all detectors over the 4 datasets by tabulating the results of AUC of ROC and TPR in Table I. From this table, it is clear that DRR consistently performs the best in terms of true positive detection rate when keeping the true negative detection rate at 0.9. Since all detectors aim for binary classification, the larger the size of the AUC of ROC, the better the detector. Ideally, AUC of ROC is 1. For the SVHN, CIFAR-10 and CIFAR-100 datasets, the AUC of DRR is the largest over all different attacks. For MNIST, DRR outperforms other detectors for most cases with the exceptions when all detectors' AUC values are close to 1. Since these exceptions occur for simple attack (FGSM) on simple dataset (MINST), we speculate all the detectors' performances are roughly the same on these cases.

## V. Ablation Study

We study the influences of $\mathrm{L}_{hinge}$ by considering 3 Combined Representation Reconstruction (CRR) methods defined below:

$$\mathrm{CRR0} : \mathrm{Loss} = \lambda \mathrm{L}_1, \tag{11}$$

$$\mathrm{CRR1} : \mathrm{Loss} = \lambda \mathrm{L}_1 + \mathrm{L}_{hinge1}, \tag{12}$$

$$\mathrm{CRR2} : \mathrm{Loss} = \lambda \mathrm{L}_1 + \mathrm{L}_{hinge2}. \tag{13}$$

Except for the difference in Loss function, all CRR variants inherit the same settings from DRR. And they share the same model architecture with DRR, which means they all use encoder En to generate semantic features and use Eq. (4) to get label features, then they reconstruct input data with the help of decoder De. Both CRR1 and CRR2 have a part of $\mathrm{L}_{hinge}$, this enables use further evaluate the contribution of each part of $\mathrm{L}_{hinge}$. Note that HVR uses semantic feature as hidden state representation, while HLR uses logits (label feature) as hide state representation. So, from the view of architecture, CRR0 can be seen as the combination of HLR and HVR.

Table II lists the performance of these CRRs and the original DRR when detecting PGD attack with $L_\infty$ (recall that $\epsilon = 0.15$) on CIFAR10. It is clear from this table (and Table I) that the direct combination of HLR and HVR, i.e., CRR0, does not promote detection performance. Moreover, from this table, it is easy to see that $\mathrm{L}_{hinge1}$ is the main reason for

TABLE I: AUC/TPR of different detectors over different datasets. TPR is calculated when TNR=0.9.

| Dataset | Attack | HVR | HLR | DRR |
|---|---|---|---|---|
| MNIST | FGSM $L_2$ | **0.999 / 1.000** | 0.985 / **1.000** | 0.998 / **1.000** |
| | FGSM $L_\infty$ | **0.998 / 1.000** | 0.967 / 0.954 | 0.995 / **1.000** |
| | PGD $L_2$ | 0.655 / 0.152 | 0.827 / 0.450 | **0.981 / 0.962** |
| | PGD $L_\infty$ | 0.804 / 0.349 | 0.856 / 0.495 | **0.983 / 0.972** |
| | DeepFool $L_2$ | 0.994 / 0.984 | 0.991 / 0.977 | **0.997 / 0.995** |
| | DeepFool $L_\infty$ | 0.997 / 0.994 | 0.996 / 0.993 | **0.998 / 1.000** |
| SVHN | FGSM $L_2$ | 0.919 / 0.629 | 0.636 / 0.173 | **0.955 / 0.930** |
| | FGSM $L_\infty$ | 0.915 / 0.662 | 0.620 / 0.129 | **0.948 / 0.893** |
| | PGD $L_2$ | 0.593 / 0.113 | 0.539 / 0.106 | **0.889 / 0.579** |
| | PGD $L_\infty$ | 0.918 / 0.677 | 0.643 / 0.125 | **0.985 / 0.986** |
| | DeepFool $L_2$ | 0.869 / 0.761 | 0.815 / 0.621 | **0.941 / 0.878** |
| | DeepFool $L_\infty$ | 0.827 / 0.627 | 0.704 / 0.343 | **0.910 / 0.742** |
| CIFAR 10 | FGSM $L_2$ | 0.757 / 0.308 | 0.673 / 0.249 | **0.911 / 0.758** |
| | FGSM $L_\infty$ | 0.761 / 0.288 | 0.678 / 0.222 | **0.904 / 0.744** |
| | PGD $L_2$ | 0.505 / 0.099 | 0.616 / 0.190 | **0.839 / 0.539** |
| | PGD $L_\infty$ | 0.778 / 0.326 | 0.767 / 0.360 | **0.942 / 0.851** |
| | DeepFool $L_2$ | 0.819 / 0.646 | 0.777 / 0.538 | **0.899 / 0.778** |
| | DeepFool $L_\infty$ | 0.749 / 0.519 | 0.677 / 0.377 | **0.848 / 0.703** |
| CIFAR 100 | FGSM $L_2$ | 0.750 / 0.287 | 0.717 / 0.246 | **0.844 / 0.485** |
| | FGSM $L_\infty$ | 0.754 / 0.292 | 0.713 / 0.227 | **0.859 / 0.535** |
| | PGD $L_2$ | 0.550 / 0.114 | 0.649 / 0.214 | **0.710 / 0.273** |
| | PGD $L_\infty$ | 0.772 / 0.255 | 0.745 / 0.257 | **0.894 / 0.632** |
| | DeepFool $L_2$ | 0.923 / 0.854 | 0.876 / 0.752 | **0.927 / 0.861** |
| | DeepFool $L_\infty$ | 0.861 / 0.714 | 0.804 / 0.590 | **0.881 / 0.743** |

promoting performance, and the standalone usage of $\mathrm{L}_{hinge2}$ harms detection. We noted that without the help of $\mathrm{L}_{hinge1}$, $\mathrm{L}_{hinge2}$ is hard to converge in training. Only when combining $\mathrm{L}_{hinge1}$ and $\mathrm{L}_{hinge2}$ will the detection performance be boosted.

TABLE II: AUC and TPR (TNR=0.9) for DRR and CRRs.

| | CRR0 | CRR1 | CRR2 | DRR |
|---|---|---|---|---|
| AUC | 0.769 | 0.885 | 0.726 | 0.942 |
| TPR | 0.278 | 0.732 | 0.211 | 0.851 |

## VI. Defensing Adaptive Adversarial Attack

To further evaluate the performance of DRR, we assume that the attacker can not only access the victim model but also knows all the details of the detector. Under this adaptive assumption, the attacker's goal is to fool both the victim model and the detector. Following the most-widely used adaptive attack strategy [13], [23], [27], the attacker now aims to solve:

$$\min_{x_{\text{adv}}} \quad \alpha \mathrm{L}_{\text{RE}}(x_{\text{adv}}) - \mathrm{L}_{\text{CE}}(x_{\text{adv}}, y) \\ \text{s.t.} \quad \|x_{\text{adv}} - x\|_{\text{inf}} < \epsilon, \tag{14}$$

where $\mathrm{L}_{\text{RE}}$ and $\mathrm{L}_{\text{CE}}$ are respectively the attack's loss function for the detector (i.e., reconstruction error) and for the victim model (i.e., cross entropy), and $y = \mathsf{F}(x)$ is the true label of $x$ and $\alpha$ is the parameter to control the relative importance of the two loss functions. It is worth mentioning that the detectors [15], [18], [19] in comparison fail to resist the adaptive attack defined above[1]. The reason is, for other AE-based detectors, no matter high-level representation (i.e., logits) is used or not, they fail to capture true features of adversarial examples by certain. In contrast, the AE in DRR mimics the behavior of adversarial examples via constructing counterexamples with disentangled representation.

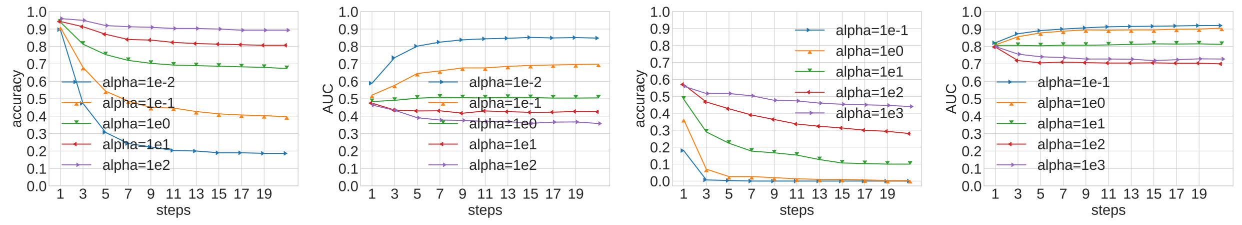
[1]We note that [18] considered a weaker adaptive attack strategy (maximize the cross entropy after AE reconstruction but not directly optimize reconstruction error) and HVR is robust to the weaker adaptive notion.

We evaluate this adaptive attack strategy with the same setting used in Sec. IV, and the results for MNIST and SVHN are depicted in Fig. 5. Observing Figs. 5(a) and (c), if the attacker set a larger value of $\alpha$ (e.g., $\alpha = 1e2$) and penalize the loss function $\mathrm{L}_{\text{RE}}$ more (to avoid being detected), it is clear that the accuracy of the model is still high (i.e., the success rate of attack is low). That said, it becomes harder for the attacker to succeed in attacking when the detector DRR is deployed. Similarly, observing Figs. 5(b) and (d), if the attacker set a smaller value of $\alpha$ (e.g., $\alpha = 1e-2$) and focus more in attack, the AUC of the detector is still high (i.e., the successful attack can be still detected). It is worth mentioning similar results can be also observed on CIFAR10/CIFAR100. To conclude, compared to other self-supervised adversarial detectors, DRR is the first of its kind that offers good resistance to adaptive adversarial attacks.

## VII. Detecting Textual Adversarial Examples

DRR is based on the very basic fact that adversarial examples lead to wrong decisions of neural models without changing semantics of the examples, and this fact is data format-agnostic. We hereby exemplify the effectiveness of DRR on textual data.

The works in [36], [37] propose to generate textual adversarial examples to attack natural language processing (NLP) models. These works try to fool victims models by modifying similar words or characters. For word-level attacks, attackers typically add perturbations by changing words to synonyms. For



(a) acc. after attack (b) AUC of detection (c) acc. after attack (d) AUC of detection

Fig. 5: Accuracy of victim model under the adaptive PGD attack with different $\alpha$. (a, b) MNIST, (c, d) SVHN.

character-level attacks, attackers typically add perturbations by modifying characters to similar ones, for example, by replacing character 'o' by '0'. We still focus on the detection of textual adversarial examples of classification models.

Different from image data, natural language is discrete and harder to reconstruct. Instead of directly reconstructing natural language sentences, we choose to transform sentences to embeddings before encoding and decoding. Then we use the reconstruction error of embeddings as a metric to detect adversarial examples. As shown in Fig. 6, we first translate discrete textual data into continuous embedding by using the Universal Sentence Encoder for English (USE) [38]. USE is widely used for representing natural language, and the embedding produced by USE can preserve the information of natural language sentences.

Similar to the method mentioned in Sec. III-A, the label feature $f_l'$ is obtained via

$$y = \mathsf{V}(s_i) = \text{softmax}(\mathsf{V}_z(s_i)), \tag{15}$$

$$f_l' = \mathsf{V}_R(s_i) = R_S(\mathsf{V}_z(s_i)), \tag{16}$$

where $s_i$ represents the input sentences. We use the same rescale function $R_S$ as in Eq. (4). In contrast to image tasks, in textual tasks, the semantic feature $f_s'$ is encoded over USE-generated embedding $e_i$ instead of the original input sentences. Similarly, the outputs of decoder are reconstructed embedding vectors. This process is characterized by

$$e_i = \mathsf{USE}(s_i), \tag{17}$$

$$f_s' = \mathsf{En}(e_i), \tag{18}$$

$$\hat{e}_i = \mathsf{De}(f_l', f_s'). \tag{19}$$

We then apply the idea of using disentangled feature to mimic both benign and adversarial examples for training DRR over embedded textual data, which is similar to that of Sec. III-A.

As shown in Fig. 6, for benign embedding $e_i$, we use MAE to make sure it is similar to the its reconstructed version $\hat{e}_i$, thus the associated loss is:

$$\mathrm{L}_1' = \mathbb{E}_{e_i}\mathsf{MAE}(e_i, \hat{e}_i). \tag{20}$$

Moreover, we modify $\mathrm{L}_{hinge}$ in the same way as above to make them fit into textual attack. The textual version hinge

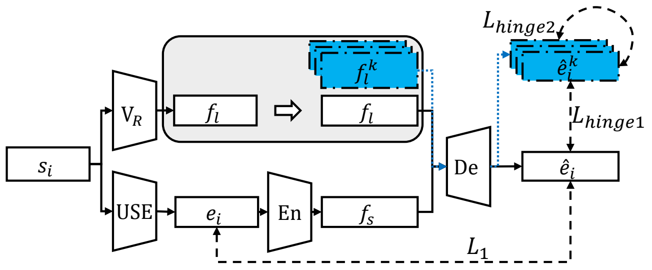


Fig. 6: Overview of DRR in NLP.

loss functions are as follows:

$$\mathrm{L}_{hinge1}' = \mathbb{E}_{x_i}\left[\max\left(0, d - \min_{\substack{k_1,k_2\in\Sigma,\\ k_1\neq k_2}}\left(\mathsf{MAE}(\hat{e}_i^{k_1}, \hat{e}_i^{k_2})\right)\right)\right], \tag{21}$$

$$\mathrm{L}_{hinge2}' = \mathbb{E}_{x_i}\left[\max\left(0, d - \min_{\substack{k\in\Sigma,\\ k\neq i}}\left(\mathsf{MAE}(\hat{e}_i^{k}, e_i)\right)\right)\right], \tag{22}$$

$$\mathrm{L}_{hinge}' = \mathrm{L}_{hinge1} + \mathrm{L}_{hinge2}, \tag{23}$$

The final loss $\text{Loss}_{text}$ to train encoder and decoder of DRR in textual tasks is thus

$$\text{Loss}_{text} = \lambda \mathrm{L}_1' + \mathrm{L}_{hinge}'. \tag{24}$$

To evaluate the performance of DRR for textual attacks, we choose the widely used BERT [39] and ALBERT [40] as victim models, and train them with datasets SST-2 [41] and AG_News [42], respectively. The SST-2 consists of 11,855 single sentences extracted from movie reviews, and there are two classes in SST-2: negative and positive. The AG_News contains more than one million news articles, which contains 4 kinds of news, including World, Sports, Business and Sci/Tech. BERT and ALBERT are pre-trained language models, after fine-tuned on SST-2 and AG_News, they can be used as classification models.

We adopt two state-of-the-art word-level attacks, Probability Weighted Word Saliency (PWWS) [36] and Genetic [37], to generate adversarial examples to evaluate the effectiveness of DRR. PWWS and Genetic generate adversarial examples by replacing synonym words. Follow the setting of previous sections, we still adopt AUC to evaluate the performances of DRR in detecting textual attack. From the AUC value of each

TABLE III: AUC of different textual attack over different datasets.

| | SST-2 | AG_News |
|---|---|---|
| PWWS | 0.79 | 0.821 |
| Genetic | 0.78 | 0.888 |

attack for different datasets listed in Table III, it is concluded DRR is also valid for defending against adversarial textual examples.

## VIII. Conclusion

In this study, we propose to use disentangled representation for self-supervised adversarial examples detection. The proposed DRR is based on the very nature of adversarial examples: misleading classification results without changing the semantics of inputs (a lot). With disentangled label and semantic features, this nature inspires us to construct counterexamples to better guide the training of DRR. Compared with previous self-supervised detectors, DRR generally performs better under various measurements over different datasets and different adversarial attack methods. Not surprisingly, compared with other AE-based adversarial detectors, DRR is also more robust to adaptive adversaries. As exemplified over textual data, the rationale for designing DRR is universal, it is possible to extend DRR for defensing adversarial attacks in other domains. These merits make DRR a promising candidate, when combined with other proactive strategies, for the defense of adversarial attacks in real applications.

## References

[1] C. Szegedy, W. Zaremba, I. Sutskever, and et al., “Intriguing properties of neural networks,” in *ICLR*, 2013.

[2] I. J. Goodfellow, J. Shlens, C. Szegedy, and et al., “Explaining and harnessing adversarial examples,” in *ICLR*, 2015.

[3] S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, and et al., “DeepFool: A simple and accurate method to fool deep neural networks,” in *CVPR*, pp. 2574–2582, 2016.

[4] N. Carlini, D. Wagner, and et al., “Towards evaluating the robustness of neural networks,” in *Oakland*, pp. 39–57, 2017.

[5] A. Madry, A. Makelov, L. Schmidt, and et al., “Towards deep learning models resistant to adversarial attacks,” in *ICLR*, 2018.

[6] K. Roth, Y. Kilcher, T. Hofmann, and et al., “The odds are odd: A statistical test for detecting adversarial examples,” in *ICML*, pp. 5498–5507, 2019.

[7] K. Grosse, P. Manoharan, N. Papernot, M. Backes, P. McDaniel, and et al., “On the (statistical) detection of adversarial examples,” *arXiv:1702.06280*, 2017.

[8] D. Hendrycks, K. Gimpel, and et al., “Early methods for detecting adversarial images,” in *ICLR*, 2016.

[9] S. Hu, X. Liu, Y. Zhang, M. Li, L. Y. Zhang, H. Jin, and L. Wu, “Protecting facial privacy: Generating adversarial identity masks via style-robust makeup transfer,” in *CVPR*, 2022.

[10] S. Hu, Y. Zhang, X. Liu, L. Y. Zhang, M. Li, and H. Jin, “Advhash: Set-to-set targeted attack on deep hashing with one single adversarial patch,” in *ACM MM*, 2021.

[11] Z. Zhang, L. Y. Zhang, X. Zheng, B. H. Abbasi, and S. Hu, “Evaluating membership inference through adversarial robustness,” *The Computer Journal*, vol. DOI: 10.1093/comjnl/bxac080, 2022.

[12] N. Carlini, D. Wagner, and et al., “Adversarial examples are not easily detected: Bypassing ten detection methods,” in *CCS*, pp. 3–14, 2017.

[13] J. Tian, J. Zhou, Y. Li, and et al., “Detecting adversarial examples from sensitivity inconsistency of spatial-transform domain,” in *AAAI*, 2021.

[14] K. Lee, K. Lee, H. Lee, and et al., “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” in *NeurIPS*, pp. 7167–7177, 2018.

[15] D. Meng, H. Chen, and et al., “Magnet: A two-pronged defense against adversarial examples,” in *CCS*, pp. 135–147, 2017.

[16] C. Chen, J. Liu, Y. Xie, Y. X. Ban, and et al., “Latent Regularized Generative Dual Adversarial Network For Abnormal Detection,” in *IJCAI*, pp. 760–766, 2020.

[17] D. Gong, L. Liu, V. Le, and et al., “Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection,” in *CVPR*, pp. 1705–1714, 2019.

[18] G. Vacanti, A. Van Looveren, and et al., “Adversarial detection and correction by matching prediction distributions,” *arXiv:2002.09364*, 2020.

[19] B. Wójcik, P. Morawiecki, M. Śmieja, and et al., “Adversarial examples detection and analysis with layer-wise autoencoders,” *arXiv:2006.10013*, 2020.

[20] Y. Qin, N. Frosst, S. Sabour, and et al., “Detecting and diagnosing adversarial images with class-conditional capsule reconstructions,” in *ICLR*, 2019.

[21] L. Schott, J. Rauber, M. Bethge, and et al., “Towards the first adversarially robust neural network model on MNIST,” in *ICLR*, 2018.

[22] K. Yang, T. Zhou, Y. Zhang, X. Tian, and D. Tao, “Class-disentanglement and applications in adversarial detection and defense,” in *NeurIPS*, 2021.

[23] N. Carlini, D. Wagner, and et al., “Adversarial examples are not easily detected: Bypassing ten detection methods,” in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14, 2017.

[24] J. Lu, T. Issaranon, D. Forsyth, and et al., “Safetynet: Detecting and rejecting adversarial examples robustly,” in *ICCV*, pp. 446–454, 2017.

[25] R. Feinman, R. R. Curtin, S. Shintre, and et al., “Detecting adversarial samples from artifacts,” *arXiv:1703.00410*, 2017.

[26] X. Ma, B. Li, Y. Wang, and et al., “Characterizing adversarial subspaces using local intrinsic dimensionality,” in *ICLR*, 2018.

[27] N. Carlini, D. Wagner, and et al., “Magnet and "efficient defenses against adversarial attacks" are not robust to adversarial examples,” *arXiv:1711.08478*, 2017.

[28] X. Chen, Y. Duan, R. Houthooft, and et al., “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” in *NeurIPS*, pp. 2172–2180, 2016.

[29] M. Sang, W. Xia, J. H. Hansen, and et al., “Deaan: Disentangled embedding and adversarial adaptation network for robust speaker representation learning,” *arXiv:2012.06896*, 2020.

[30] L. Tran, X. Yin, X. Liu, and et al., “Disentangled representation learning gan for pose-invariant face recognition,” in *CVPR*, pp. 1415–1424, 2017.

[31] R. Shwartz-Ziv, N. Tishby, and et al., “Opening the black box of deep neural networks via information,” *arXiv:1703.00810*, 2017.

[32] A. Ilyas, S. Santurkar, D. Tsipras, and et al., “Adversarial examples are not bugs, they are features,” in *NeurIPS*, pp. 125–136, 2019.

[33] G. E. Hinton, O. Vinyals, J. Dean, and et al., “Distilling the knowledge in a neural network,” *ArXiv*, vol. 1503.02531, 2015.

[34] J. Rauber, W. Brendel, M. Bethge, and et al., “Foolbox: A python toolbox to benchmark the robustness of machine learning models,” in *ICML*, 2017.

[35] K. Simonyan, A. Zisserman, and et al., “Very deep convolutional networks for large-scale image recognition,” in *ICLR*, 2015.

[36] S. Ren, Y. Deng, K. He, and et al., “Generating natural language adversarial examples through probability weighted word saliency,” in *ACL*, 2019.

[37] M. Alzantot, Y. Sharma, A. Elgohary, and et al., “Generating natural language adversarial examples,” in *EMNLP*, 2018.

[38] D. M. Cer, Y. Yang, S. yi Kong, and et al., “Universal sentence encoder for english,” in *EMNLP*, 2018.

[39] J. Devlin, M.-W. Chang, K. Lee, and et al., “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL*, 2019.

[40] Z. Lan, M. Chen, S. Goodman, and et al., “ALBERT: A lite bert for self-supervised learning of language representations,” in *ICLR*, 2019.

[41] R. Socher, A. Perelygin, J. Wu, and et al., “Recursive deep models for semantic compositionality over a sentiment treebank,” in *EMNLP*, 2013.

[42] X. Zhang, J. J. Zhao, Y. LeCun, and et al., “Character-level convolutional networks for text classification,” in *NeurIPS*, 2015.