What You See is Not What the Network Infers: Detecting Adversarial Examples Based on Semantic Contradiction

Yijun Yang*, Ruiyuan Gao*, Yu Li*, Qiuxia Lai† and Qiang Xu*
*CUhk REliable Computing Laboratory (CURE Lab.), Dept. of Computer Science and Engineering,
The Chinese University of Hong Kong, Hong Kong S.A.R., China
{yjyang, rygao, yuli, qxu}@cse.cuhk.edu.hk
†State Key Laboratory of Media Convergence and Communication, *Communication University of China*, Beijing, China

***Abstract*—Adversarial examples (AEs) pose severe threats to the applications of deep neural networks (DNNs) to safety-critical domains, e.g., autonomous driving. While there has been a vast body of AE defense solutions, to the best of our knowledge, they all suffer from some weaknesses, e.g., defending against only a subset of AEs or causing a relatively high accuracy loss for legitimate inputs. Moreover, most existing solutions cannot defend against adaptive attacks, wherein attackers are knowledgeable about the defense mechanisms and craft AEs accordingly.**

In this paper, we propose a novel AE detection framework based on the very nature of AEs, i.e., their semantic information is inconsistent with the discriminative features extracted by the target DNN model. To be specific, the proposed solution, namely *ContraNet*[1], models such contradiction by first taking both the input and the inference result to a generator to obtain a synthetic output and then comparing it against the original input. For legitimate inputs that are correctly inferred, the synthetic output tries to reconstruct the input. On the contrary, for AEs, instead of reconstructing the input, the synthetic output would be created to conform to the wrong label whenever possible. Consequently, by measuring the distance between the input and the synthetic output with metric learning, we can differentiate AEs from legitimate inputs. We perform comprehensive evaluations under various AE attack scenarios, and experimental results show that ContraNet outperforms existing solutions by a large margin, especially under adaptive attacks. Moreover, our analysis shows that successful AEs that can bypass ContraNet tend to have much-weakened adversarial semantics. We have also shown that ContraNet can be easily combined with adversarial training techniques to achieve further improved AE defense capabilities.

I. Introduction

Deep learning-based systems have achieved unprecedented success in numerous long-standing machine learning tasks [29], [39]. Their safety and trustworthiness have become public concerns with the broader deployment of Deep Neural Networks (DNNs) in various mission-critical applications, e.g., autonomous driving [17] and medical diagnosis [35], [53]. In these applications, incorrect decisions or predictions could cause catastrophic financial damages or even life losses [32].

One of the primary threats to DNNs is Adversarial Examples (AEs), which introduce subtle malicious perturbation to the inputs to fool the DNN model [2], [71]. While the perturbation is small and often imperceptible to humans, it dramatically changes the features extracted by the targeted DNN model, leading to wrong inference results. Depending on attackers' knowledge, adversarial attacks can be classified into three categories: *black-box attack*, *white-box attack*, and *adaptive attack*. Their detailed descriptions are provided in Table I.

A vast body of research has been dedicated to AE defense, considering the severity of the threat. Existing methods include model robustification with adversarial training techniques (e.g., [49], [66]), input transformation to mitigate the impact of AEs (e.g., [51], [61]), and various types of AE detectors that try to differentiate legitimate inputs and AEs according to specific criteria (e.g., [13], [67]). While effectively improving the robustness of DNN models, to the best of our knowledge, they all suffer from some weaknesses, e.g., defending against only a subset of AEs or causing a relatively high accuracy loss for legitimate inputs.

More importantly, almost all defense solutions suffer from significant performance degradation with *adaptive attacks*, wherein attackers craft new AEs with the knowledge about the defense solutions. For example, AutoAttack [19] reports lower robust accuracy on most previous defense solutions by automatically tuning the attack hyperparameters. Recently, the Orthogonal Project Gradient Descent (Orthogonal-PGD) attack [7] has reduced the robust accuracy of several earlier unbroken defenses to nearly zero in the worse case.

Generally speaking, a DNN model performs complicated non-linear mappings from the high-dimensional input space to a low-dimensional discriminative feature space [8]. Since the training is conducted on limited samples, it is likely to have loopholes in the model where adjacent inputs differ significantly in the feature space during the mapping. We argue that existing AE defense solutions fail to defend against adaptive AEs because *it is extremely challenging, if not impossible, to eliminate or model all the loopholes at the feature space.*

Unlike existing solutions, we do not intend to eliminate or model the loopholes of the DNN model. The *very nature* of AEs is the contradiction between the semantics of the input samples and the model outputs. We propose a novel AE detection framework by directly modeling such contradiction, namely *ContraNet*. As we *conduct AE detection at the input space* with ContraNet, it is less likely to evade by adaptive attacks.

[1]Our code and models are available at https://github.com/cure-lab/ContraNet.git.

Network and Distributed Systems Security (NDSS) Symposium 2022
24-28 April 2022, San Diego, CA, USA
ISBN 1-891562-74-6
https://dx.doi.org/10.14722/ndss.2022.24001
www.ndss-symposium.org

TABLE I: Adversarial Attack Types

Attack Type	Description
Black-box attack	*Adversaries have no access to either the detailed information of the target model or its defense mechanism, but can query the model* [57].
White-box attack	*The model architecture and parameters are exposed to the adversaries, but the defense mechanism is kept confidential (non-adaptive)* [9], [45].
Adaptive attack	*Adversaries have full knowledge of both the target model and the defense mechanism, and could craft attacks accordingly* [9].

To be specific, we first train a generator by encoding the legitimate samples and their corresponding labels as the generator's inputs. The generated synthetic results strive to preserve the semantics of the samples according to their labels. During inference, we feed both the input and the output of the targeted DNN model to obtain the synthetic result. For legitimate inputs that are correctly inferred, the synthetic output tries to reconstruct the input. For AEs, instead of reconstructing the input, the synthetic result would be created to conform to the wrong label whenever possible. Consequently, by estimating the similarity between the input and the synthetic result, we can differentiate AEs from legitimate inputs.

The contributions of this work include:

- To the best of our knowledge, ContraNet is the first work to explore the AE's intrinsic property for detection at input space, which has the potential to resist adaptive attacks.
- To realize the potential of ContraNet, we propose to generate synthetic samples that keeps the semantics of the inference result and compare with input samples for AE detection. This is achieved by designing a new conditional Generative Adversarial Network (cGAN) network.
- We develop an effective and efficient similarity measurement model to tell the semantic difference between the input and the synthetic samples.

We perform comprehensive evaluations on several popular image classification datasets under various AE attack scenarios. Experimental results show that ContraNet outperforms existing solutions by a large margin, especially under adaptive attacks. Moreover, our analysis shows that those successful AEs that can bypass ContraNet tend to have much-weakened adversarial semantics. We have also shown that ContraNet can be easily combined with adversarial training techniques to further improve the AE defense capabilities of DNN models.

The remainder of this paper is organized as follows. Sec. II discusses related work in AE attacks and defenses. Next, we give an overview of ContraNet in Sec. III, followed by the concrete design shown in Sec. IV. Then, we empirically evaluate the performance of ContraNet against both white-box attacks and adaptive attacks in Sec. V and Sec. VI, respectively. Sec. VII discusses the limitations of ContraNet. Finally, Sec. VIII concludes this paper.

II. Background and Related Works

A. Adversarial Example Attacks

FGSM. Goodfellow *et al.* [26] propose the first simple yet efficient method – Fast Gradient Sign Method (FGSM) – to construct AEs against a given DNN classifier. FGSM generates the AE by performing a one-step optimization on the input image towards the gradient ascent direction.

BIM. Basic Iterative Method (BIM) [40] is an iterative variant of FGSM. Under a certain perturbation budget, instead of optimizing the AE in one step as in FGSM, BIM uses smaller steps and iteratively optimizes the AE.

PGD. Project Gradient Descent (PGD) proposed by Madry *et al.* [49] is a powerful iterative attack method, where the search step starts from a random position in the neighborhood of the clean input. As PGD relaxes the search direction, it can search AEs with subtle perturbations faster. PGD has also been used as a basic building block to construct stronger attacks, e.g., AutoAttack [19] and Orthogonal-PGD [7], breaking many state-of-the-art AE defenses.

C&W. While the above methods are all variants of FGSM and are based on gradient ascent, C&W attack [12] is the first work that formulates the AE generation process as an optimization problem, and it can successfully construct AEs with much smaller perturbations when compared to earlier techniques.

EAD. In [14], Chen *et al.* treat the adversarial attack as an elastic-net regularized optimization problem and propose the EAD attack. EAD enhances the attack transferability, i.e., the ability of AEs generated against one model being able to attack another unseen model successfully.

B. Defenses against Adversarial Examples

Existing defense techniques alleviate the impact of AEs by conducting transformation to the inputs, adversarially training the model, or detecting anomalies based on specific criteria.

Gradient masking/ obfuscation schemes (e.g., [42], [57]) try to construct robust models with gradients that are difficult to use by attackers. However, such defense solutions can be easily circumvented with black-box attacks such as Zoo [15] or attacks with gradient approximation capabilities [1].

Adversarial training methods aim at improving the robustness of a model by adding AEs into the training phase. For example, [49] trains a robust model with AEs generated using PGD attack on the fly. Other methods include changing the loss function (e.g., TRADE [83]), changing the activation function (e.g., [25]), utilizing artificially generated training examples (e.g., [27]) or reweighting misclassified samples (e.g., [76], [84]). There are also adversarial training methods that utilize ensemble learning [55], metric learning [43], and self-supervised learning [16] techniques. Adversarial training methods are easy to implement, but they inevitably decrease the accuracy of legitimate inputs [19].

Input transformation-based defense techniques (e.g., [51], [61]) try to "purify" the inputs before feeding them into the DNN model. By doing so, carefully crafted adversarial perturbations are changed, thereby mitigating their attack abilities. Due to the nature of this technique, there is an inherent tradeoff between tolerable perturbations and prediction accuracy for legitimate inputs.

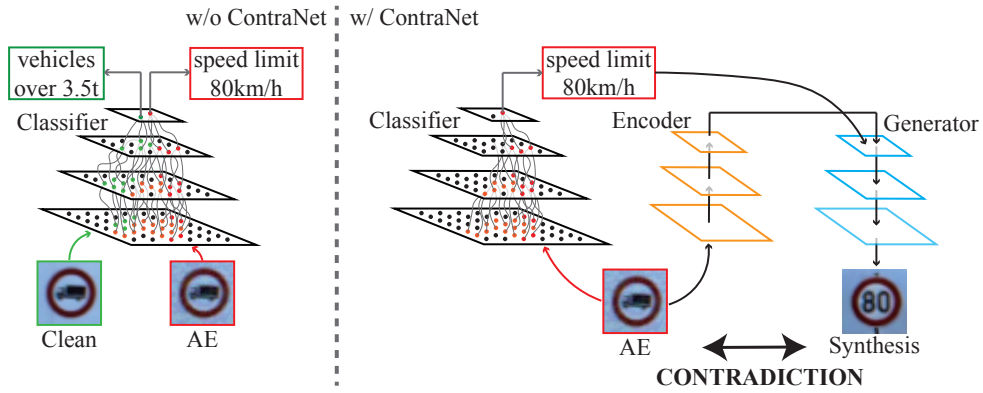

Fig. 1: The overview of the proposed *ContraNet* solution. The clean input and the crafted AE are similar to each other while getting distinct discriminative features during DNN inference. ContraNet highlights this contradiction by projecting the DNN-extracted discriminative features back to the input space, i.e., ContraNet flags it to be an AE if the synthetic image is dissimilar to the input.

Detection-based methods aim at building an auxiliary module, i.e. an *AE detector*, to reject AEs. Some AE detectors are constructed without changing the target DNN model. For example, *MagNet* [51] proposes two AE detectors based on reconstruction error and probability divergence, respectively. *Feature Squeezing (FS)* [79] builds a detector based on the instability of the output from AEs under different feature squeezing methods. Other AE detectors are tightly coupled with the protected DNN model, thus retraining is needed. In [67], Shan *et al.* introduce trapdoors into the target classifier and leads attacks to gravitate towards trapdoors. AEs are identified by comparing neuron activation signatures of inputs to those of trapdoors. Pang *et al.* [56] propose to use true confidence as an indicator of the uncertainty of the predictive results. Since true confidence is not available during inference, they use an MLP branch to learn the rectified confidence as a substitute and reject those samples with low uncertainty as AEs.

As discussed earlier, the existence of AEs is due to the loopholes in the DNN model, which map some adjacent inputs to distant discriminative features. From this perspective, adversarial training techniques try to reduce the number of loopholes with additional samples. However, the added training samples could be far from complete to cover all possible loopholes. Input-transformation defense tries to shift the AE that maps to a loophole point to a safe zone, but its success is not guaranteed, and it may also shift a legitimate input and mistakenly map it to a loophole point. AE detectors try to find one or several criteria to model the loopholes. Their capability against AE attacks is thus dependent on how easy to bypass the corresponding criteria.

III. ContraNet Overview

A. Threat Model

Adversarial goals. Adversarial attacks are divided into two categories: targeted and untargeted attacks. Targeted attacks succeed when the DNN model outputs a specified incorrect label set by the attacker. In comparison, untargeted attacks lead DNN to provide an arbitrary label that differs from the ground truth [12], [58]. In this paper, we conduct both targeted and untargeted attacks for the evaluations.

Adversary's Knowledge. We consider the two most challenging cases for the defender: white-box and adaptive attacks, as shown in Tab. I. The adversary in a white-box attack scenario has complete knowledge of the classifier, including its network architecture, exact parameters, but the proposed defense is kept secret, i.e. in the white-box attack scenario the attacker is unaware of the existence of the defense method. Further, in the adaptive attacks[2], we assume that the adversary has full access to the proposed defense, i.e., ContraNet, including the detection algorithm, the Encoder $\mathbf{E}$, the Generator $\mathbf{G}$, and the Similarity Measurement Model, but not the discriminators, i.e., D_Φ and D_{aux}, which are only used in training [3]. This adaptive attack scenario represents the most potent adversaries; as informed in [7], [9], a substantial number of existing defenses are broken under adaptive attacks.

Adversarial Capabilities. We impose several reasonable constraints on the adversary. The adversary cannot destroy the integrity of the model, i.e., the attacker cannot attack the training process, such as poisoning attacks [70], [77], backdoor attacks [44], [74], nor directly modify the parameters inside the models, e.g., bit-flips attack [31], [73], [80], fault injection attack [4], [47]. Additionally, we assume that the data pre-processing stage cannot be tampered with by an adversary.[4]

B. Design Motivation

According to the definition of AE [71], we deduce two objectives an AE needed to fulfill simultaneously: 1) causing DNN to make a wrong prediction; 2) being indistinguishable from its source image. The first item ensures the impaction of AE, while the second objective avoids AE being easily identified. Formally, we summarize the intrinsic feature of AEs as the following corollary:

Corollary 1: AE's visual semantic information contradicts its discriminative features extracted by the DNN under adversarial attack.

[2]This is a white-box attack setting for the proposed defense method.

[3]E, G, D_Φ, D_{aux} are the components of ContraNet whose detailed design information can be found in the following sections

[4]We further discuss the possibilities for implementing ContraNet to be resistant to the data pre-processing attack in Sec. VII.

Fig. 1's left part illustrates this corollary: the clean input and the crafted AE appear almost identical to humans. However, the DNN model is activated quite differently in the feature space, thus predicting the clean sample as the "vehicles over 3.5t" sign and the AE as the "speed limit 80km/h" sign.

DNN-based image classifiers perform complicated non-linear mappings from the high-dimensional input image space to the low-dimensional feature space. Especially considering that DNN models are constructed with limited training samples, the multi-to-one mapping from DNN models leaves many loopholes for attackers to generate possible AEs that are similar to the clean sample in the image space but differ significantly in the feature space.

Directly checking whether the semantic consistency holds between the input image space and the feature space is intractable. However, if we project the inference result of the DNN model back to the input space, the contradiction within Corollary. 1 can be reflected in the image space. We could then compare it with the original input at the same space, as shown in the right part of Fig. 1. This is the key idea of ContraNet.

C. Overview of ContraNet

To perform semantic comparison at the input space, ContraNet employs an Encoder to capture the low-level features of the input image, and feed it together with the inference result of the DNN model to a Generator. The semantics of the synthetic image would conform to the DNN model's output, e.g., the semantics of the synthesis shown in the right part of Fig. 1 is faithful to DNN's prediction, i.e., "speed limit 80 km/h". We can then easily tell the difference between the AE input and the synthetic image and use it for AE detection.

To further demonstrate our motivation, we empirically show some cases in Fig. 2, where *the synthetic images' semantic information is highly dependent on the inferred result from the classifier*. To be specific, given input images in the first column, we generate synthetic images (in other columns) by varying the label (shown on the top) associated with the generation. Such procedures mimic the semantics changes caused by AEs at the feature space. As can be observed, ContraNet faithfully reconstructs the synthetic images when the label is congruent with the input's semantics. However, when the given label contradicts the input image's semantics, the generated synthetic image would be pretty different from the input image. Due to space limitations, more empirically generated syntheses are provided in Appendix D, as shown in Fig. 17.

D. Resistance to Adaptive Attacks

Because ContraNet is designed based on the semantic contradiction, attackers cannot inject imperceptible noise-like perturbations (without any semantic information) to bypass ContraNet. Instead, the adversarial perturbations need to be added toward the target class. However, such kinds of perturbations naturally weaken the AEs' evil properties, as they would cease to be malevolent if the semantics of the perturbed image, follows that of the DNN output. In other words, the only way to eliminate the semantic contradiction is to perturb the image to be alike with a clean image from another class. Such perturbations are not malicious anymore.

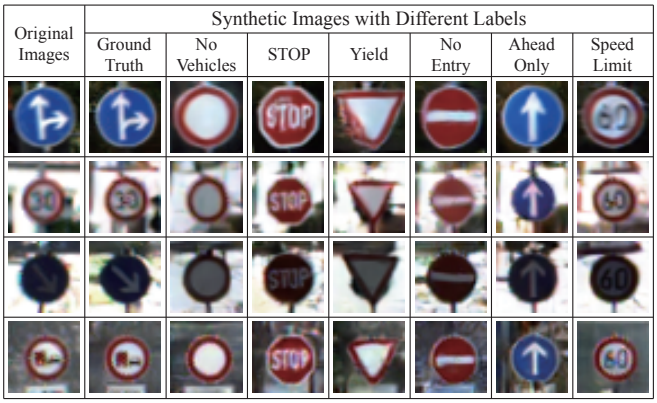

Fig. 2: The semantic meaning of the synthesis x' is faithful to the conditional label y. In each row, the first element is the input image x, and the second element depicts the generated synthetic images conditioned on the ground truth and other labels.

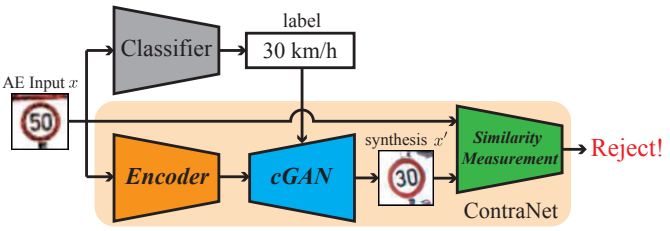

Fig. 3: The proposed ContraNet design, containing an Encoder, a conditional Generator (cGAN), and a Similarity measurement model.

IV. Concrete ContraNet Design

In this section, we detail the proposed ContraNet design. First, we depict an overview of ContraNet's implementation. Then, we discuss several key adaptations for cGAN that have aided the implementation of ContraNet's core concept. To further improve ContraNet's detection capability, we propose a similarity measurement model for determining the similarity between the input image and its synthesis. Finally, we provide a training process for ContraNet.

A. Implementation Overview

As depicted in Fig. 3, ContraNet consists of three components, an Encoder ($\mathbf{E}$), a Generator ($\mathbf{G}$), and a Similarity measurement model. To be more specific, the encoder $\mathbf{E}$ is in charge of extracting the low-level features of the input image x. The generator $\mathbf{G}$ is built based on a *class-conditional GAN* (cGAN) [5], [52]. Specifically, $\mathbf{G}$ takes as input the low-level features of x summarized by $\mathbf{E}$, while the predicted label of x given by the classifier serves as the conditional input. The main purpose of $\mathbf{G}$ is to synthesize an image x' whose low-level features such as colors and textures are faithful to the input image x, while its semantics conform to the conditional input, i.e., the predicted label of x. In this way, the semantic meaning of the synthesized image x' is highly related to the label given by the classifier. The Similarity measurement model measures the similarity between the input image x and its synthesized counterpart x'.

During inference, given an input $x \in \mathcal{X}_{\text{test}}$, we first generate its synthetic counterpart x' with $x' = \mathbf{G}(\mathbf{E}(x), y)$,

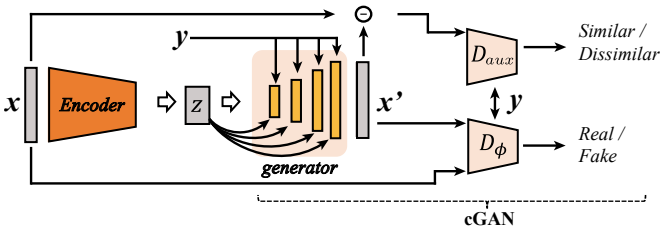

Fig. 4: The revised cGAN in ContraNet. The Encoder summaries x's low-level features as z, and splits z into several slices to impact generator's deeply. D_Φ is cGAN's vanilla component, while D_{aux} is specific for our task, used to evaluate similarity.

conditioned on its predicted label $y = C(x)$, where $C(\cdot)$ indicates the classifier. Specifically, for a legitimate input that is correctly inferred, its DNN-extracted feature is consistent with its semantic information. Therefore, its synthetic image $x' = \mathbf{G}(\mathbf{E}(x), y)$ would be similar to itself x. Otherwise, the synthetic image's semantic information will be substituted to the predicted label y. Lastly, the similarity measurement model will evaluate the similarity between x and x', to identify AEs.

B. Encoder and cGAN in Cooperation

The Encoder, $\mathbf{E}$, and the cGAN's generator, $\mathbf{G}$, are responsible for projecting the predicted label y back to image space and obtaining a synthetic image whose low-level features are faithful to the input image, and its semantic meaning highly depends on the given y. In this section, we introduce several critical design considerations when implementing these two models. The overview of Encoder and cGAN is in Fig. 4.

Enhancing y's influence. As discussed in Sec. III, to realize the detection mechanism of ContraNet, the synthetic image's semantic information should significantly correlate to the classifier's predicted label, y. To enhance the influence of y on the synthetic image's semantic meaning, we adopt the Conditional Batch Normalization (CBN) [21], [24] technique on cGAN's generator. Within CBN, y directly influences each batch normalization layer's mean and variance parameters. Therefore, the y's impaction can spread to the generator's deeper layers than merely concatenate y to the input layer. For the discriminator part, we employ the conditional projection layer technique [52]. The projection layer enhances y's influence by mapping the one-hot label y to a high-dimensional embedding and projecting the embedding onto cGAN's discriminator's middle layer. This projection structure enables high-quality class information transformation.

Getting a meaningful latent vector. One requirement of ContraNet is that the synthesis x' should be faithful to x. In this case, given a consistent pair of x and y, the synthetic image x' should be similar to x. To this end, we develop an Encoder to summarize the input's low-level features to the latent vector $z = \mathbf{E}(x)$, as shown in Fig. 4. We employ $\mathcal{SSIM}$ and ℓ_2 between the input image and its synthesis as constraints. We also apply a *KL-Divergence* loss, $\mathcal{L}_{D_{KL}}$, to regular $z \sim \mathcal{N}(0, \mathbf{I})$ [23], which guarantees the latent vector to be representative enough.

Let the latent vector z impact the deeper layers of cGAN's generator. As introduced above, we encode x's low-level features as a latent vector z with $\mathbf{E}$, then $\mathbf{G}$ is trained to synthesize x', which resembles x given a consistent y. In order to make better use of the information provided by z, following [5], we chunk the z into several slices, z_i, and distribute these z_i into deeper layers of the generator, as shown in Fig. 4's *generator* part. In this way, the influence of z can be extended to deeper layers. The synthetic image x' can reconstruct x as much as possible when the given label consistent with x's original semantic meaning.

Coupling an auxiliary discriminator. In the vanilla cGAN's adversarial training process, the discriminator, designated by D_Φ, is responsible for distinguishing generated images from real ones. While the generator is optimized to generate realistic images that trick the discriminator, D_Φ since the original task of vanilla cGAN is to generate realistic and various images. Whereas in our task, an additional criterion must be met, i.e., the synthetic image, x', should be faithful to its corresponding input image, x, given a consistent, y. As a result, adding an auxiliary discriminator, D_{aux}, is appropriage to determine if the synthesized image x' is close to the corresponding input x.

To be specific, the auxiliary discriminator, D_{aux}, focuses on the difference between the input and its synthesis. We use $x - x'$ as the input directly. The positive input, x_{pos}, is defined as the difference between x and its synthetic image under a consistent label y, denoted as x'_y (Eq. (1)). The negative input, x_{neg}, is defined as x minus its synthetic image generated under an inconsistent label y', denoted as $x'_{y'}$ (Eq. (2)).

$$x_{pos} = x - x'_y \tag{1}$$

$$x_{neg} = x - x'_{y'} \tag{2}$$

We employ the *hinge loss* (Eq. (3)) as the criterion for D_{aux}, which will push the distance of positive input to be larger than 1, while pulling that of negative input to be less than -1.

$$\mathcal{L}_{D_{aux}} = ReLU(1 - D_{aux}(x_{pos}, y)) + ReLU(1 + D_{aux}(x_{neg}, y)) \tag{3}$$

C. Similarity measurement model

We devise a similarity measurement model to distinguish the similarity between x and x'. If x and x' are deemed dissimilar, ContraNet will reject x; otherwise, x is believed to be a benign sample. We make the judgment from three different perspectives. The proposed Deep Metric Model ($\mathbf{DMM}$) provides similarity judgment by comparing a hierarchical distance between x and x'. In contrast $\mathcal{SSIM}$ provides a pixel-to-pixel similarity judgment. Further, we apply a $\boldsymbol{Dis}$ model to reject a sample according to the quality of its synthesis. The insight of $\boldsymbol{Dis}$ is that if the given label does not match a sample's semantic meaning, the synthesis will appear more unrealistic than the synthesis generated with the accordant label. These three rejectors work in a tandem way, where any rejection of the three leads to a final rejection.

Deep Metric Model. The insight of deep metric learning is to develop a network that distinguishes similar from dissimilar example pairs by learning a hierarchical distance. Deep metric learning, e.g., siamese network [37] and triplet network [33], is widely used for similarity measurement tasks, such as image retrieval, face recognition and signature verification [6], [64], [75]. Inspired by the above concept, we adopt a triplet network to our $\mathbf{DMM}$ by labeling images from the same class as positive pairs while images from different classes as negative

pairs. Since there are slight differences on the data distribution between x and x', we use two models, $\mathbf{F_{real}}$ and $\mathbf{F_{syn}}$ trained on x and x', respectively, as their feature extractors.

After getting the two pretrained feature extractors, we treat the concatenated embeddings $ep = Concat(\mathbf{F_{real}}(x), \mathbf{F_{syn}}(x'_y))$ as positive while $en = Concat(\mathbf{F_{real}}(x), \mathbf{F_{syn}}(x'_{y'}))$ as negative. Finally, we train a Multi-Layer Perceptron (MLP) model with en, ep as input and the final similarity decision as its output. We construct this MLP by three fully-connected layers. We use *Cross-Entropy loss*, denoted by $\mathcal{L}_{CE}$ [3], as the objection function for this binary classification task. In our experiments, we incorporate AEs generated by PGD attack [49] into the training set to improve the robostness of the **DMM**. A more detailed training process is provided in Algorithm 1, line 22-29.

SSIM. $\mathcal{SSIM}$ is a typical similarity metric. $\mathcal{SSIM}$ judges the similarity between x and x' in pixel-level without the learning process. Thus, $\mathcal{SSIM}$ can be used directly as a rejector, which flags AEs with low $\mathcal{SSIM}$ similarity.

Discriminator. During the training process of the generator, since the input label is accordant with the input image, the synthetic image x' will be an exact reconstruction of the input image x, and appear realistic. However, when y contradicts x, x' could be regarded as an image formed by substituting x's semantic information with y's image-space projection. Because of the contradiction, the generated x' will look unrealistic and be of poor quality. Based on the above observation, we propose to use a Discriminator, ***Dis***, as a rejector. Its objective is set in Eq. (4). Note that ***Dis*** receives only the synthetic image x' as input and rejects a sample according to its synthesis's quality. It is difficult for the adversary to influence ***Dis***'s decision by adding subtle adversarial perturbation to x. Therefore, ***Dis*** can promote the robustness of ContraNet against AEs.

$$\mathcal{L}_{Dis} = ReLU(1 - Dis(x'_y, y)) + ReLU(1 + Dis(x'_{y'}, y)) \quad (4)$$

D. Implementation details

The whole ContraNet consists of the Encoder $\mathbf{E}$, the Generator $\mathbf{G}$, the two discriminators D_ϕ and D_{aux}, and the similarity measurement model (**DMM** and ***Dis***). We train ContraNet using Adam [36] with $\beta_1 = 0.9$, $\beta_2 = 0.99$, the batch size setting at 128. We use a typical linear learning rate decay strategy with a starting point at 0.0005. Following [81], we employ EMA with decay factor at 0.9999 as Generator's regularization technique. We adopt adding random noise and *DiffAugment* [85] as data augmentation. The overall training algorithm for ContraNet has been summarized as Algorithm 1, and we provided the detailed model architectures in Appendix C.

Training process. To obtain $\mathbf{E}$ and $\mathbf{G}$, there are four models needed to be trained: an encoder ($\mathbf{E}$), a generator ($\mathbf{G}$), a discriminator (D_Φ), and an auxiliary discriminator (D_{aux}). The $\mathbf{E}$, $\mathbf{G}$, D_Φ together can be viewed as a revised cGAN. Therefore, we train these three models following the standard GAN training process. The adversarial loss for the revised cGAN's generator part ($\mathbf{E}$ and $\mathbf{G}$) is denoted as $\mathcal{L}_{\mathbf{E+G}}$ in Eq. (5):

$$\mathcal{L}_{\mathbf{E+G}} = \mathcal{L}_{D_{KL}} + \ell_2 + \mathcal{SSIM} + \mathcal{L}_{\mathbf{G}}, \quad (5)$$

Algorithm 1 ContraNet Training Framework

Stage1: train ContraNet's Encoder and Generator
Input: Training data $\mathcal{X} = \{x\}^N$, $\mathcal{Y} = \{y\}^N$
Output: The parameters of $\mathbf{E}$, $\mathbf{G}$, $\boldsymbol{D_\Phi}$
▷ Train $\mathbf{G}$, $\mathbf{E}$ and $\boldsymbol{D_\Phi}$.
1: **for** some training iterations **do**
2: $x'_y = \mathbf{G}(\mathbf{E}(x), y)$;
3: Feed x, x'_y and y into $\boldsymbol{D_\Phi}$;
4: Optimize $\mathbf{G}$ and $\mathbf{E}$ for $\mathcal{L}_{\mathbf{E+G}}$ (Eq. (5));
5: Optimize $\boldsymbol{D_\Phi}$ for $\mathcal{L}_{D_\Phi}$ (Eq. (6));
6: **end for**
▷ Train $\boldsymbol{D_{aux}}$ with fixed $\mathbf{G}$ and $\mathbf{E}$.
7: **for** some training iterations **do**
8: $x_{pos} = x - \mathbf{G}(\mathbf{E}(x), y)$, $x_{neg} = x - \mathbf{G}(\mathbf{E}(x), y')$;
9: Feed x_{pos}, x_{neg} and y into $\boldsymbol{D_{aux}}$;
10: Optimize $\boldsymbol{D_{aux}}$ for $\mathcal{L}_{D_{aux}}$ (Eq.(3));
11: **end for**
▷ Fine-tune $\mathbf{G}$, $\mathbf{E}$ and $\boldsymbol{D_\Phi}$ with fixed $\boldsymbol{D_{aux}}$.
12: **for** some training iterations **do**
13: $x'_y = \mathbf{G}(\mathbf{E}(x), y)$, $x'_{y'} = \mathbf{G}(\mathbf{E}(x), y')$;
14: $x_{pos} = x - x'_y$, $x_{neg} = x - x'_{y'}$;
15: Feed x_{pos} and x_{neg} into $\boldsymbol{D_{aux}}$ to compute $\mathcal{L}'_{\mathbf{E+G}}$ (Eq. (7));
16: The same as line 3-5;
17: **end for**
18:
Stage2: train Similarity measurenet model's Dis
Input: Training data $\mathcal{X} = \{x\}^N$, $\mathcal{Y} = \{y\}^N$, fixed $\mathbf{E}$ and $\mathbf{G}$.
Output: The parameters of ***Dis***.
19: **for** some training iterations **do**
20: $x'_y = \mathbf{G}(\mathbf{E}(x), y)$, $x'_{y'} = \mathbf{G}(\mathbf{E}(x), y')$;
21: Feed x'_y, $x'_{y'}$ and y into ***Dis*** and optimize for $\mathcal{L}_{Dis}$ (Eq. (4));
22: **end for**
23:
Stage3: train Similarity measurenet model's **DMM**
Input: Training data $\mathcal{X} = \{x\}^N$, $\mathcal{Y} = \{y\}^N$, fixed $\mathbf{E}$ and $\mathbf{G}$.
Output: The parameters of **DMM** including $\mathbf{F_{real}}$, $\mathbf{F_{syn}}$ and **MLP**.
24: Train feature extractor $\mathbf{F_{real}}$ on x;
25: Train feature extractor $\mathbf{F_{syn}}$ on $x' = \mathbf{G}(\mathbf{E}(x), \cdot)$;
▷ Train **DMM** with pretrained $\mathbf{F_{real}}$ and $\mathbf{F_{syn}}$
26: **for** some training iterations **do**
27: e_p = Concat($\mathbf{F_{real}}(x), \mathbf{F_{syn}}(x'_y)$), label as positive;
28: e_n = Concat($\mathbf{F_{real}}(x), \mathbf{F_{syn}}(x'_{y'})$), label as negative;
29: Feed e_p, e_n to **MLP**
30: Optimize $\mathbf{F_{real}}$, $\mathbf{F_{syn}}$ and **MLP** for $\mathcal{L}_{CE}$;
31: **end for**

where $\mathcal{L}_{\mathbf{G}} = ReLU(1 - D_\Phi(x'_y, y))$ is the GAN loss on $\mathbf{G}$. Furthermore, the adversarial loss for the revised cGAN's discriminator part (D_Φ) can be written as Eq. (6).

$$\mathcal{L}_{D_\Phi} = ReLU(1 - D_\Phi(x, y)) + ReLU(1 + D_\Phi(x'_y, y)) \quad (6)$$

The training procedure of this revised cGAN is demonstrated in Algorithm 1, line 1-6.

The purpose of D_{aux} is to improve the similarity between x and x' when the given conditional label matches x's semantic information. Thus, adding D_{aux} in the latter training phase when $\mathbf{E}$ and $\mathbf{G}$ can already generate meaningful synthesis makes sense. As opposed to letting D_{aux} join the training from scratch, adding D_{aux} in the later training phase reduces the difficulty of optimization and makes the training process smoother and easier to converge. To be more specific, first, we obtain D_{aux} by training it with fixed $\mathbf{E}$ and $\mathbf{G}$, which can already generate realistic but similarity unsatisfactory synthesis, as demonstrated in Algorithm 1 line 7-11. Then we freeze D_{aux} and add it to the $\mathcal{L}_{\mathbf{E+G}}$ and get the fine-tune loss, $\mathcal{L}'_{\mathbf{E+G}}$, as demonstrated in Eq. (7). We show the fine-tuning procedure in Algorithm 1 line 12-17.

TABLE II: Information of Datasets and Classifiers.

Dataset	# of Class	Trainset Size	Testset Size	Classifier Architecture
MNIST [41]	10	50,000	10,000	2Conv, 2FC
GTSRB [69]	43	35,288	12,630	ResNet18 [29]
CIFAR-10 [38]	10	50,000	10,000	DenseNet169 [34]

$$\mathcal{L}'_{\mathbf{E}+\mathbf{G}} = \mathcal{L}_{D_{KL}} + \ell_2 + \mathcal{SSIM} + \mathcal{L}_{\mathbf{G}} + \mathcal{L}_{D_{aux}} \tag{7}$$

Note that D_ϕ and D_{aux} are only auxiliary models to help the revised cGAN consider the label information. Only **E** and **G** are used during inference.

V. EVALUATION

This section reports ContraNet's performance against white-box attacks. Following previous defenses [51], [56], [67], [79], we evaluate ContraNet on four popular adversarial attacks together with a new adversarial attack benchmark, *AutoAttack*. [19] Note that a more rigorous evaluation against adaptive attacks is given in Sec. VI.

A. Experimental Settings

White-box attack. In the white-box attack setting, the attacker has complete knowledge of the classifier, whereas the detector is confidential.

Datasets. We conduct experiments on MNIST, GTSRB, and CIFAR-10, which are the *de facto* datasets used to evaluate AE defenses. We use commonly adopted classifier architectures, such as ResNet18, DenseNet169. More details of the datasets and classifiers are in Tab. II.

Evaluation metrics. The metrics employed evaluate ContraNet from two perspectives: 1) the impact on classification accuracy on legitimate samples, and 2) the ability of AE detection. We conventionally denote the adversarial example as the *Positive sample* (P) and the clean sample as the *Negative sample* (N). Unless specified, we use the same evaluation metrics for the adaptive attack in Sec. VI.

- **Detector's metric.** TPR@FPR n% indicates the True Positive Rate (TPR) when fixing the threshold with False Positive Rate (FPR) $\leq n\%$. This metric evaluates the performance of the detector alone. The FPR describes the fraction of normal samples being flagged as AE. In general, lower FPR induces lower TPR. Therefore, there is a trade-off between TPR and FPR depending on the application scenario. Commonly, TPR@FPR 5% serves as the primary metric to evaluate the detector's performance [51], [54], [67]. In this work, we report the typical TPR@FPR 5% as the main indicator. The TPR at a lower FPR, TPR@FPR3%, is also provided to demonstrate ContraNet's performance further.
- **Detector's accuracy on clean samples.** Acc_{dec} indicates the detector's accuracy on clean samples by combining the detector with the classifier, as addressed in Eq. (8). This metric reflects ContraNet's impact on clean samples.

$$Acc_{dec} = \frac{\#Classifer\ correct\&Detector\ pass}{\#all\ clean\ samples} + \frac{\#Classifier\ wrong\&Detector\ reject}{\#all\ clean\ samples} \tag{8}$$

TABLE III: Detector's Accuracy on Clean Sample ($Acc_{dec} \uparrow$)

Dataset	Acc_{ori}*	**ContraNet**	Trapdoor	RR	MagNet	FS
CIFAR-10 [38]	95.50%	**92.02%**	80.22%	82.69%	88.30%	*91.38%*
GTSRB [69]	98.90%	**95.04%**	92.11%	92.09%	91.03%	*94.12%*
MNIST [41]	99.8%	*95.56%*	94.40%	**95.75%**	90.13%	94.2%

* indicates the classification accuracy on clean samples without detectors.
◇ The **bolded** values are the highest performance. The *underlined italicized* values are the second highest performance.

- **ROC curve & AUC.** We plot Receiver Operating Characteristic (ROC) curves to elaborate the influence of the various threshold settings. A more general metric, Area Under the Curve (AUC), is provided to give an overall summary for its corresponding ROC curve.
- **Robust accuracy on AEs.** Acc_{rob} demonstrates the attacker's failure rate, as shown in Eq. (9), where the *Successful AEs* are AEs that fool the classifier and bypass the detector, and *all AEs* are samples that have been perturbed by attacks. This metric follows [7], [51], [54], [67]. Acc_{rob} can well reflect the overall performance of the whole system (considering both the classifier and the detector).

$$Acc_{rob} = 1 - \frac{\#Successful\ AEs}{\#all\ AEs} \tag{9}$$

Baselines. We choose four detection-based defense methods as the baselines, including two high cited schemes: MagNet [51] and Feature Squeezing (FS) [79] and two most recent schemes: Trapdoor [67] and Rectified Rejection (RR) [56]. Although most baselines have been bypassed under adaptive attacks by later literature or the authors themselves [11], [28], [30], they still get excellent performance against attackers with limited power, e.g., the white-box attacks. Thanks to their authors, we reproduce their work with officially open-source codes (For TensorFlow projects, we re-implement them using Pytorch).

B. ContraNet against White-box Attacks

We evaluate the performance of ContraNet across three datasets with their associated classifiers, as shown in Tab. II. Note that, Trapdoor and RR require specially/adversarially trained classifiers. We show respect to this setting when implementing their methods. We use four typical untargeted attacks for evaluation, including the iterative attack PGD [49] and BIM [40], and optimization-based C&W [12] and EAD [14]. These attacks are broadly used as evaluations of our baselines. We included the exact attack parameter settings in Appendix A. The misclassification rate can achieve over 98% for every attack on the vanilla classifier.

Impact on accuracy with clean images. Tab. III summarizes the performance of ContraNet and baselines on clean samples after fixing the threshold @FPR5%. As can be seen, the accuracy decrease caused by ContraNet is at most 4.5% across three datasets, while that of RR is 12.8%, Trapdoor 15.3%, MagNet 9.7%, FS 5.6%. ContraNet causes little impact on the original accuracy, especially when compared to methods requiring re-training the classifier, e.g., RR and Trapdoor.

Robust Accuracy with Detector. ContraNet maintains a high Acc_{rob} across a variety of attacks and datasets, as seen in Tab. IV. The above attack-agnostic property is thanks

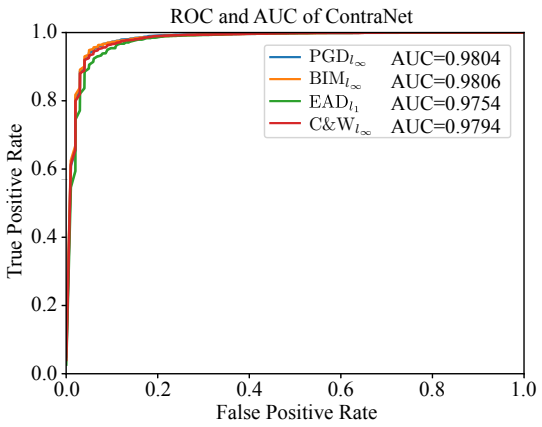

Fig. 5: ContraNet's ROC and AUC. ROCs are generated under four attacks PGD$_{\ell_\infty}$, BIM$_{\ell_\infty}$, EAD$_{\ell_1}$, C&W$_{\ell_\infty}$, along with the corresponding AUC (higher is better) demonstrated in the legend block.

to ContraNet's detection mechanism. By using the semantic contradiction, ContraNet identifies AEs without the need for classifier knowledge or assuming the attack types. Consequently, ContraNet becomes more general than non-semantic-based methods.

Detection performance. Tab. V reports the TPR@FPR5% or 3% to present ContraNet's AE detection ability. It is confirmed that ContraNet can generalize well to different attacks for the same reason outlined in Acc_{rob}. In addition, we can observe that ContraNet outperforms or is at least on par with other detection-based defenses. Especially when encountering CIFAR-10 dataset whose distribution is relatively complex, ContraNet leads all other methods by a large margin.

ROC curve and AUC. In Fig. 5, we plot ROC curves together with AUC values for CIFAR-10. TPR rises rapidly, with FPR increasing from low values. This trendency indicates ContraNet can detect AE with high accuracy while keeping a low impact on legitimate samples. Furthermore, AUC can reflect the overall performance with different thresholds. ContraNet's AUCs have all been close to 1, indicating it can detect AEs very well under different attacks. ROC under ℓ_2-norm is in Appendix B.

Failure Cases. In Fig. 6, we visualize representative failure cases (false positives and false negatives, respectively) of ContraNet. As can be observed, those legitimate input images that are misjudged as AEs (Fig. 6 (a)) are corrupted to some degree, which is difficult to perceive even for humans. Interestingly, many of the AEs misjudged as clean inputs (Fig. 6 (b)) have somehow lost their adversarial properties to humans as the semantics of the AEs are indeed close to that of the adversarial label. The above phenomena further demonstrate the effectiveness of ContraNet.

C. ContraNet against AutoAttack

AutoAttack [19] is designed for evaluating Adversarially Trained Classifier (ATC). It is an ensemble of various PGD attacks and covers a wide range of attack settings (e.g., targeted/untargeted attacks and white-box/black-box attacks). As a benchmark, AutoAttack dedicates to providing a sufficient and impartial evaluation of adversarial defenses with auto-tuned hyperparameters. However, AutoAttack can only attack

(a) False positives of ContraNet. First row: clean images misjudged as AEs. Second row: the corresponding synthetic images.

(b) False negatives of ContraNet. First row: AEs generated by C&W attack but misjudged as clean images by ContraNet. Second row: the corresponding synthetic images.

Fig. 6: False positives and false negatives of ContraNet.

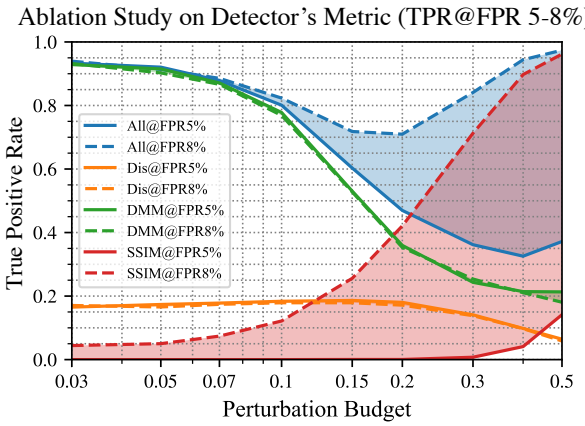

Fig. 7: Ablation study for Similarity measurement model on TPR@FPR5% to 8%. AEs are generated by PGD-ℓ_∞ on CIFAR-10.

classifiers, but not detection-based methods. Therefore, we follow the white-box setting demonstrated in Sec. V-A, and use AEs generated by AutoAttack on each ATC from [18] for evaluation. As shown in Tab. VI, the addition of ContraNet increases the robustness of ATC by a significant margin on both clean samples and AEs. To be more specific, compared with the vanilla classifier ($> 95\%$ on CIFAR-10), the standalone ATC induces a poor Acc. When equipped with ContraNet, the accuracy increases to $> 91.4\%$ indicated by Acc_{dec}. This is reasonable because ContraNet can distinguish misclassification clean samples caused by ATC, thus increasing accuracy on clean samples. When it comes to AEs, ContraNet can filter out AEs that have fooled ATC, resulting in a 15.87% uplift on Acc_{rob}, as shown in Tab. VI's the 3rd and 4th columns.

D. Ablation Study

In this section, we perform an ablation study on each component of the Similarity measurement model, i.e., **DMM**, $\boldsymbol{Dis}$ and $\mathcal{SSIM}$, and analysis their detection ability under a wide scope of perturbation budgets. We show the *TPR@FPR 5-8% v.s. Perturbation budgets* curves keeping the thresholds as in Sec. V-B (Fig. 7).

TABLE IV: Acc_{rob} under White-box Attack

Dataset		CIFAR-10	GTSRB	MNIST	CIFAR-10	GTSRB	MNIST
Attack Method	Defense	Acc_{rob}@FPR5%↑			Acc_{rob}@FPR3%↑		
PGD_{ℓ_∞}	**ContraNet**	**90.58%**	**97.63%**	**100%**	**81.59%**	**95.05%**	**100%**
	Trapdoor	58.55%	*94.95%*	99.88%	55.59%	*94.75%*	99.88%
	RR	57.15%	75.86%	99.34%	54.73%	74.01%	98.97%
	MagNet	*74.12%*	47.50%	**100%**	*72.56%*	43.09%	**100%**
	FS	63.57%	91.47%	96.69%	56.05%	85.98%	96.61%
BIM_{ℓ_∞}	**ContraNet**	**91.45%**	*97.55%*	**100%**	**81.29%**	*94.30%*	**100%**
	Trapdoor	58.55%	**97.78%**	99.51%	*72.08%*	**97.16%**	*99.47%*
	RR	54.24%	71.75%	99.3%	51.85%	70.39%	98.97%
	MagNet	*61.35%*	44.51%	**100%**	59.86%	40.39%	**100%**
	FS	48.26%	94.59%	96.52%	42.90%	90.81%	96.48%
EAD_{ℓ_1}	**ContraNet**	**86.55%**	**98.59%**	*94.54%*	**74.09%**	**96.44%**	**93.1%**
	Trapdoor	45.36%	1.22%	83.51%	42.54%	0.36%	80.37%
	RR	11.92%	16.32%	**99.87%**	7.74%	8.24%	**99.67%**
	MagNet	*75.41%*	*98.14%*	85.58%	*73.22%*	*96.11%*	83.49%
	FS	49.75%	64.57%	5.05%	*73.22%*	35.16%	3.23%
$C\&W_{\ell_\infty}$	**ContraNet**	**89.49%**	**99.04%**	*99.75%*	*78.35%*	**97.59%**	**99.75%**
	Trapdoor	25.39%	4.94%	82.73%	22.86%	4.2%	79.72%
	RR	11.58%	15.27%	**99.87%**	7.19%	7.43%	*99*.71%
	MagNet	*82.82%*	*96.39%*	95.74%	**81.29%**	*93.36%*	94.45%
	FS	50.79%	41.57%	17.51%	22.85%	30.86%	13.25%

* For PGD, BIM, and C&W, $\ell_\infty = 8/255$, and for EAD, $\ell_1 = 8/255$, the same below.
◇ The **bolded** values are the best performance, and the *underlined italicized* values are the second-best performance, the same below.

TABLE V: TPR@FPR 5% or 3% of White-box Attack

Dataset		CIFAR-10	GTSRB	MNIST	CIFAR-10	GTSRB	MNIST
Attack Method	Defense	**TPR@FPR5%** ↑			**TPR@FPR3%** ↑		
PGD_{ℓ_∞}	**ContraNet**	**92.6%**	**97.81%**	**100%**	**83.55%**	**95.38%**	**100%**
	Trapdoor	*81.07%*	*95.00%*	99.87%	59.32%	*94.8%*	99.87%
	RR	16.32%	21.97%	**100%**	59.32%	*94.8%*	99.87%
	MagNet	76.68%	44.26%	**100%**	*75.19%*	39.77%	39.51%
	FS	65.50%	90.76%	14.81%	57.67%	84.82%	11.11%
BIM_{ℓ_∞}	**ContraNet**	**93.62%**	*97.79%*	**100%**	**83.49%**	*94.70%*	**100%**
	Trapdoor	*86.86%*	**97.98%**	99.50%	*75.28%*	**97.44%**	99.46%
	RR	13.92%	15.31%	**100%**	8.51%	11.05%	97.73%
	MagNet	63.82%	43.00%	**100%**	62.42%	39.56%	94.41%
	FS	50.38%	94.39%	13.79%	44.91%	90.46%	12.07%
EAD_{ℓ_1}	**ContraNet**	**89.12%**	**99.07%**	*94.62%*	**76.96%**	*96.88%*	93.78%
	Trapdoor	51.24%	1.23%	84.17%	48.12%	0.39%	81.06%
	RR	10.73%	14.49%	**100%**	7.11%	6.74%	**99.96%**
	MagNet	*78.59%*	*98.61%*	86.85%	*76.65%*	**97.21%**	96.12%
	FS	31.29%	65.43%	5.08%	*76.65%*	56.10%	3.28%
$C\&W_{\ell_\infty}$	**ContraNet**	**92.51%**	**99.11%**	**99.92%**	*81.50%*	**98.50%**	99.92%
	Trapdoor	25.39%	5.06%	83.38%	25.06%	4.3%	80.4%
	RR	11.58%	13.15%	*99.72%*	6.45%	5.85%	**100%**
	MagNet	*82.82%*	*96.82%*	96.93%	**85.24%**	*94.41%*	73.75%
	FS	50.79%	41.97%	17.73%	23.57%	31.13%	13.41%

TABLE VI: ATC with ContraNet against AutoAttack on CIFAR-10

ATC Methods	Acc	Acc_{dec}	Acc_{rob} (under AutoAttack)	
	ATC	ATC + ContraNet	ATC	ATC + ContraNet
G.2020Uncovering [27]	85.28%	91.73%	57.14%	84.96%
R.2021Fixing_28 [60]	87.32%	91.43%	57.42%	82.54%
R.2021Fixing_70 [60]	88.97%	91.40%	57.33%	83.58%
S.2020Hydra [65]	88.53%	91.69%	64.46%	80.33%

Deep Metric Model (DMM) is the most effective detector when the adversarial perturbation falls in small scope ($< 10^{-1}$, green lines in Fig. 7). Especially when the adversarial perturbation ranges from 0.03 to 0.07, DMM alone can achieve an over 90% TPR. The detection ability of the Similarity measurement model relies on DMM for FPR@5%-8% situations in this perturbation range. However, when the perturbation grows, DMM's performance degrades sharply. It is not hard to reason about this observation. When the adversarial perturbation is larger than 0.1 on ℓ_∞-norm , obvious image quality degrade occurs. The noise-like (non-semantic) perturbation will result in the loss of the image's original semantic information. Since the DMM has no access to such distorted images during training, it performs poorly.

$\mathcal{SSIM}$ is responsible for detecting AEs with extra-large adversarial perturbations. $\mathcal{SSIM}$ is not a learning-based similarity metric and its judgment criterion can be consistent with human perception [63]. Therefore, $\mathcal{SSIM}$ can hardly be broken with the same perturbation budgets as DNN-based metrics. We find almost no effect of $\mathcal{SSIM}$ on the clean sample's accuracy when we set the FPR to 5%. However, this also limits $\mathcal{SSIM}$'s ability to detect AE (see Fig. 7). $\mathcal{SSIM}$'s detection ability can show up when relaxing the FPR a bit. If we relax FPR from 5% to 8%, the Detector's Acc obtained by $\mathcal{SSIM}$ will increase to 40% for $\epsilon = 0.2$, and this number will rise to 80% for $\epsilon = 0.5$, as shown by red shadow lines. Such increases contribute much to the overall Detector's Acc, as indicated by the blue shadow.

Discriminator. We visualize Dis's TPR curves by orange lines

in Fig. 7. ***Dis*** assists **DMM** against middle-level adversarial perturbation (0.1-0.2 under ℓ_∞-norm). The adding of ***Dis*** acts as a relay linking **DMM** with $\mathcal{SSIM}$. The overall performance is better than using any component alone, as shown by blue lines in Fig. 7. Although ***Dis*** has a limited effect on the white-box attacks, *the real power of* ***Dis*** *is its robustness against adaptive attacks*, as analyzed in Sec. VI.

In conclusion, **DMM**, ***Dis***, and $\mathcal{SSIM}$ detect AE using different mechanisms; thus, their cooperation contributes to the overall performance of ContraNet. Note that each detector's performance can be configured relying on the actual situation.

E. Comparison with Other Detection Methods

Trapdoor. The detection principle of Trapdoor is to induce the adversary to generate AEs fall into pre-designed trap door(s) by retraining the classifier with stamped samples. We observe that this strategy works well with gradient-based attacks, such as PGD and BIM, where the perturbation is relatively large. In terms of optimization-based attacks like C&W, and EAD, the small perturbation constraint remains in their objective functions. These attacks tend to find optimal alternatives other than the trap door to fool the classifier.

RR. RR is an adversarial training detection framework, which gains better performance for seen attacks, i.e., PGD and similar BIM attacks but this advantage will vanish when encountering attacks not included in the training process, like C&W or EAD.

MagNet and FS. As two early detection-based defenses, MagNet and FS display inconsistent detection ability across datasets and attacks. For example, MagNet is vulnerable to BIM attacks, and FS performs poorly on CIFAR-10.

VI. Adaptive Attacks

In this section, we switch our role from a defender to an attacker. To best utilize ContraNet's knowledge, we first give a detailed adaptive objective loss function design; then, we try to break ContraNet using three types of adaptive attacks. The first two adaptive attacks are based on public-known strong iterative attacks, PGD [49] and C&W [12]. To further evaluate ContraNet's robustness, we employ a rising adaptive attack benchmark, *Orthogonal PGD* [7], which focuses on breaking detection-based defenses.

A. Experimental Settings

Adaptive attack. In the adaptive attack setting, an adversary has *complete knowledge* of the classifier and the defense scheme. Therefore, the adversary can develop an *adaptive attack* to fool both simultaneously.

Dataset. All experiments are conducted on CIFAR-10, which serves as a standard task by several public robustness test-benches [18], [48].

Evaluation Metrics. The evaluation metrics are consistent with those introduced in Sec. V-A. Additionally, we demonstrate the Acc_{rob} *v.s. Perturbation budget* curve for ContraNet in order to further evaluate its effectiveness against various adversarial capabilities. The perturbation budget represents the upper limit of the adversarial capabilities.

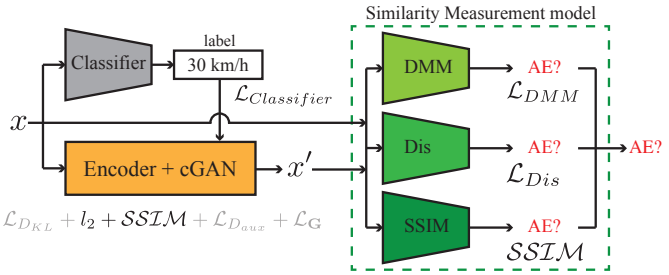

Fig. 8: Loss terms in ContraNet that can be attacked adaptively. We drop $\mathcal{L}_{D_{KL}}$ for its little impact on adaptive attack. We drop the $\mathcal{L}_{\mathbf{G}}$ and $\mathcal{L}_{D_{aux}}$ because D_Φ and D_{aux} are only used during training.

B. Customizable Adaptive Objective Loss Function

As evidenced in [9], [10], it is critical to choose the proper loss function for adaptive attacks. On the one hand, an adaptive attack aims to simultaneously bypass the detector and fool the classifier. On the other hand, an overly complex objective loss function will complicate the optimization process resulting in failure attacks or suboptimal solutions. Notice that only the classification loss can mislead the prediction result. Therefore, the attacker should always consider the classification loss. The attackers may not consider all the losses items in the defense. Instead, they can focus on the defense's weakest point.

We re-depict ContraNet's components with their training losses in Fig. 8. ContraNet's loss functions can be further fine-grained into *Deep Metric Model's loss* ($\mathcal{L}_{DMM}$, use Cross Entropy here), *Discriminator's loss* ($\mathcal{L}_{Dis} = ReLU(1 - Dis(x))$), $\mathcal{SSIM}$ [63] and ℓ_2. Adaptive attacks are conducted by varying the above loss items from targeting the single one to combinations, as shown in Eq. (10) – (17). Among these objectives, Eq. (10) covers all losses used to train ContraNet; Eq. (11) covers all three functions used in the Similarity measurement model during inference. In addition, other objectives (e.g., Eq. (15) – (17)) are also promising due to their simpler optimization process.

$$\mathcal{L}_{ContraNet_1} = \ell_2 + \mathcal{SSIM} + \mathcal{L}_{DMM} + \mathcal{L}_{Dis} \tag{10}$$
$$\mathcal{L}_{ContraNet_2} = \mathcal{SSIM} + \mathcal{L}_{DMM} + \mathcal{L}_{Dis} \tag{11}$$
$$\mathcal{L}_{ContraNet_3} = \mathcal{L}_{DMM} + \mathcal{L}_{Dis} \tag{12}$$
$$\mathcal{L}_{ContraNet_4} = \mathcal{SSIM} + \mathcal{L}_{DMM} \tag{13}$$
$$\mathcal{L}_{ContraNet_5} = \mathcal{SSIM} + \mathcal{L}_{Dis} \tag{14}$$
$$\mathcal{L}_{ContraNet_6} = \mathcal{L}_{DMM} \tag{15}$$
$$\mathcal{L}_{ContraNet_7} = \mathcal{L}_{Dis} \tag{16}$$
$$\mathcal{L}_{ContraNet_8} = \mathcal{SSIM} \tag{17}$$

Objective loss function for PGD adaptive attack. To conduct adaptive attack based on PGD attack, we modify the objective loss function as following:

$$\mathcal{L}'_{PGD} = \mathcal{L}_{Classifier} + \underbrace{\lambda \cdot \mathcal{L}_{ContraNet_i}}_{where\ i=1,\dots,7}, \tag{18}$$

where the $\mathcal{L}_{Classifier}$ is the same as original *PGD* to attack the classifier, and $\lambda \cdot \mathcal{L}_{ContraNet_i}$ will try to evade ContraNet. Notably, Orthogonal-PGD is a variant of PGD that has the same adaptive objective loss function as PGD.

Objective loss function for C&W adaptive attack. As for C&W attack, we keep the original C&W objective loss items

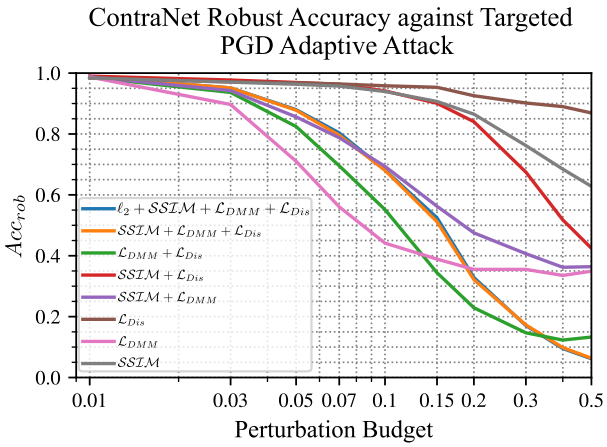

Fig. 9: Adaptive PDG attacks on ContraNet according to multiple objectives under different perturbation budgets.

proposed in [12], while introducing the $\mathcal{L}_{ContraNet_i}$ to generate AEs to evade ContraNet. As stated in Eq. (19), the first item of $\mathcal{L}_{C\&W}$ minimizes the distance between x and x_{adv}. The second item, $c \cdot f(x_{adv})$, aims at deceiving the classifier. Finally, $\lambda \cdot \mathcal{L}_{ContraNet_i}$ is to evade the ContraNet.

$$\mathcal{L}'_{C\&W} = \underbrace{\overbrace{\|x_{adv} - x\|^2}^{min\ perturbation} + \overbrace{c \cdot f(x_{adv})}^{misclassificated}}_{\mathcal{L}_{C\&W}} + \underbrace{\lambda \cdot \mathcal{L}_{ContraNet_i}}_{where\ i=1,\ldots,7} \quad (19)$$

C. Performance against PGD Adaptive Attacks

PGD experimental settings. We perform the PGD targeted adaptive attacks on various objective loss functions defined in Sec. VI-B. We varies the adversary perturbation budget from 0.01 to 0.5 with ℓ_∞-norm, to cover a large range of adversarial capabilities. The iteration steps are set to 200 (we also tested with 400 to verify the attacker's ability, as recommended in [9]). We set $\lambda = 1$ in Eq. (18).

Robust accuracy v.s. perturbation budget curves. Fig. 9 summarizes the Acc_{rob} varying with different perturbation budgets associated with various objective loss functions. We analyze Fig. 9 as follows: 1) When the perturbation is under 0.1, with $\mathcal{L}_{DMM}$, PGD can attack the whole model to the lowest Acc_{rob}. This may be because that **DMM** plays a significant role mostly in detecting the AEs with small perturbations, as stated in Section V-D. 2) The line for $\mathcal{L}_{DMM}$ flattens out as the perturbation budget grows (> 0.1). This might result from ***Dis*** and $\mathcal{SSIM}$ starting to work, and only attack **DMM** can not degrade ***Dis*** and $\mathcal{SSIM}$'s performance. Therefore, to further degrade ContraNet's performance, we have to take ***Dis*** into consideration. 3) By comparing the green and pink lines, we can find that as the perturbation budget grows, attacking $\mathcal{L}_{DMM} + \mathcal{L}_{Dis}$ is more effective than $\mathcal{L}_{DMM}$. 4) With the perturbation increasing continually, $\mathcal{SSIM}$'s detection capabilities begin to show. In this case, merely attacking **DMM** and ***Dis*** cannot influence $\mathcal{SSIM}$'s performance. Consequently, (ℓ_2+) $\mathcal{SSIM} + \mathcal{L}_{DMM} + \mathcal{L}_{Dis}$ shown by the yellow (blue) line becomes the most effective attack. 5) Other objective loss functions are either too simple or have not targeted the weakest point of ContraNet, therefore, less effective.

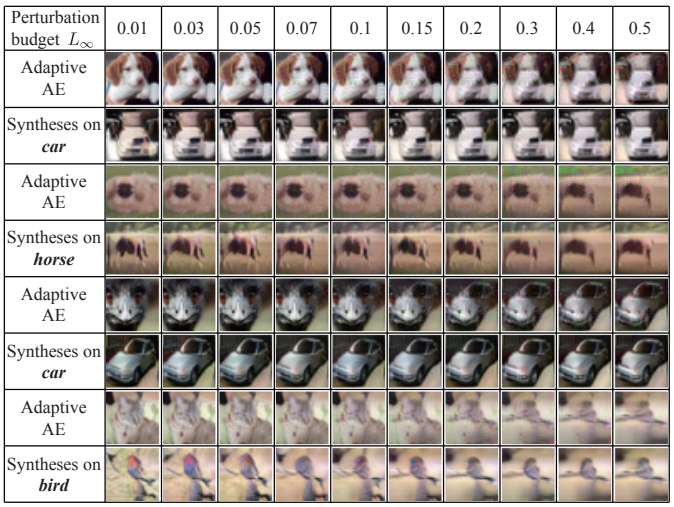

Fig. 10: Adaptive AE attacks under different perturbation budgets. To bypass ContraNet, the generated AEs resemble the target class, thereby gradually losing malignity. We call it AE's "self-defeat".

AE's "self-defeat". According to Corollary 1, AEs will lose their malignity if their semantic information changes. We call this phenomenon AE's "self-defeat". Therefore, it is beneficial to trigger AE's self-defeat, when defending adaptive attacks.

We notice that ContraNet's detection mechanism can result in adaptive AEs losing their malignancy. As shown in Fig. 10, all adaptive AEs will end up being similar to their synthetic images, and appear the semantic features of its targeted label. Therefore, these AEs are "self-defeated".

The observed self-defeat phenomenon is related to ContraNet's detection mechanism. Consider a successful AE that fools the classifier to a targeted label and bypasses the ContraNet. To evade ContraNet, 1) $\mathcal{SSIM}$ requires AE perceptually similar to the synthesis; 2) **DMM** requires AE semantically similar to the synthesis; and 3) ***Dis*** requires AE with high visual quality. Therefore, this successful AE should be very similar to its synthetic image. Notice that the synthesis from ContraNet always keeps the visible semantic feature of the targeted class. Hence, a successful AE will resemble a clean sample from that class.

Enhancing ContraNet's performance with adversarial training. Incorporating with Adversarial-Trained Classifier (ATC) can further improve the robustness performance of ContraNet. We directly use an ATC, Gowal2020Uncovering [27], from RobustBench [18], to incorporate with ContraNet forming a robust classifier (*ATC + ContraNet*). Fig. 11 shows this robust classifier's Acc_{rob} v.s. *perturbation budget* curve in orange. The green line demonstrates the combination of the normal-trained classifier and ContraNet, denoted as *ContraNet*. Clearly, when the perturbation is relatively small, *ATC + ContraNet* outperforms the non-adversarial classifier. However, when the adversarial perturbation grows larger than 0.1, *ContraNet* tends to overtake *ATC + ContraNet*.

Actually, ATC helps ContraNet defend against large adver-

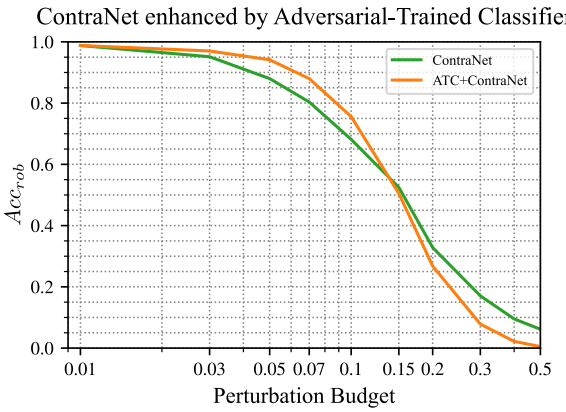

Fig. 11: Robust accuracies under adaptive PGD attack on CIFAR-10. *ContraNet* indicates applying ContraNet on normal-trained classifier, while *ATC+ContraNet* indicates combining ContraNet with Adversarially-Trained Classifier (Gowal2020Uncovering [27] from RobustBench [18]).

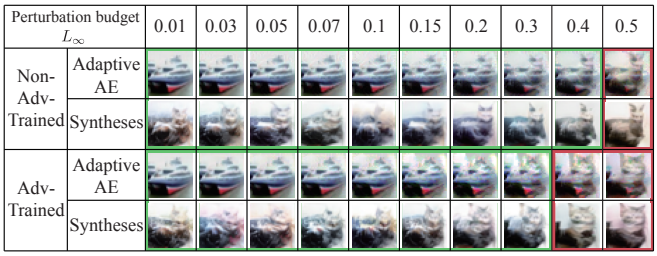

Fig. 12: AEs and synthetic images on Adversarially (Adv) / Normal (Non-Adv) -Trained classifier + ContraNet against adaptive PGD attack on CIFAR-10. The targeted class is "cat". Cases in green boxes are rejected by ContraNet. Cases in red boxes can bypass ContraNet. $\ell_\infty = 0.4$ shows that although ATC+ContraNet performs poorly on rejection than normal classifier+ContraNet, the corresponding AE are more likely to be self-defeated.

sarial perturbation by triggering AE's self-defeat. [62], [66] have stated that when the perturbation goes larger, the gradients of ATC will become interpretable. Such property of ATC can help trigger AE's self-defeat. To further explain this phenomenon, we give concrete examples in Fig. 12. Despite the attacker requiring lower perturbation budgets, $\ell_\infty = 0.4$, to breach *ATC + ContraNet* than a normal-trained classifier, the adaptive AE generated on *ATC + ContraNet* is more like a cat than the one produced on *normal-trained classifier + ContraNet*. As a result, adversarial training and ContraNet can benefit each other in triggering AE's self-defeat.

To sum up, for small perturbations, ATC enhances ContraNet's performance by increasing the Acc_{rob} directly. While for large perturbations, ATC supports ContraNet to lure the adaptive AE to be self-defeated, i.e., look like the clean sample from the targeted class.

D. Performance against C&W Adaptive Attacks

C&W experimental settings. We employ the adaptive losses designed for ContraNet in Sec. VI-B with C&W's objectives as the final adaptive objectives. We set the iteration steps to 1000 (verified to converge the optimal attack), the initial constant to 0.01, and binary search steps to 9. We employ Adam [36] with a 0.005 learning rate as the optimizer to search for the AEs. As for the hyperparameter λ in Eq. (19), we try $\lambda = 1$ and $\lambda = c$ in our experiments. [5]

TABLE VII: Robust Accuracy of ContraNet against Targeted C&W Adaptive Attack

Objective Loss Function	Acc_{rob}
$\mathcal{L}_{C\&W} + \mathcal{L}_{DMM}$	56.30%
$\mathcal{L}_{C\&W} + \mathcal{L}_{Dis}$	56.37%
$\mathcal{L}_{C\&W} + \mathcal{L}_{Dis} + \mathcal{L}_{DMM}$	56.25%
$\mathcal{L}_{C\&W} + \mathcal{L}_{Dis} + \mathcal{L}_{DMM} + \mathcal{SSIM}$	56.09%
$\mathcal{L}_{C\&W} + l_2 + \mathcal{L}_{Dis} + \mathcal{L}_{DMM} + \mathcal{SSIM}$	**55.28%**
$\mathcal{L}_{C\&W} + \lambda \cdot (l_2 + \mathcal{L}_{Dis} + \mathcal{L}_{DMM} + \mathcal{SSIM})$	56.37%

◇ The **bolded** value indicates the worst performance.

Robustness accuracy against C&W adaptive attacks. Tab. VII demonstrates the performance of ContraNet against targeted and untargeted C&W attacks. As can be seen, under C&W attacks, ContraNet can still keep a 55.28% Acc_{rob}. Moreover, different $\mathcal{L}_{ContraNet_i}$ tend to result in similar Acc_{rob}. This may be due to $\mathcal{L}_{C\&W}$, which optimizes AE's perturbation to be small. From Sec. V-D and VI-C, we notice that **DMM** is the most effective component to small perturbation. Therefore, except $\mathcal{L}_{DMM}$, other loss items in the adaptive C&W attacks only have little impact on Acc_{rob}.

E. Performance against Orthogonal-PGD Adaptive Attacks

Orthogonal-PGD experimental settings. Orthogonal Projected Gradient Descent (Orthogonal-PGD) [7] is the most recent proposed benchmark for AE detection defenses. There are two attack strategies in Orthogonal-PGD, *Selective strategy* (Select) and *Orthogonal strategy* (Orth). In the *Selective strategy*, Orthogonal-PGD update the input by selectively use the perturbation generated by the classifier or that of the detector to avoid the over-optimization on either of the two. The *Orthogonal strategy* only keeps the orthogonal component of the gradient from both the classifier and the detector when optimizing to prevent them from disturbing each other. We adopt these two strategies to attack ContraNet from three aspects:

- *Adopting Orthogonal-PGD to ContraNet directly.* We perform the two optimization strategies of Orthogonal-PGD, selective and orthogonal on ContraNet, respectively and report the results.
- *Combining Orthogonal-PGD with adaptive loss designs.* Sec. VI-C shows that attacking **DMM** is the most effective when the perturbation is relatively small. Since we use Orthogonal-PGD with $\ell_\infty = 0.01$ and $8/255$, we strengthen Orthogonal-PGD by letting it attack **DMM** alone.
- *Explore how adversarial training helps ContraNet.* We also test ContraNet's performance by letting it work with Adversarial-Trained Classifier (ATC) (as in Sec. VI-C) to further explore how adversarial training techniques help ContraNet.

We set the optimization step to 1000 for all the experiments. We consider two adversarial perturbation budgets, $\ell_\infty = 0.01$ and $\ell_\infty = 8/255$, and report the Acc_{rob} @FPR5% and @FPR50% as the evaluation metrics (following Orthogonal-PGD, performance @FPR50% serves as the worst case).[6]

[5] c in Eq. (19) is binary searched by C&W

[6] In [7], the authors report the successful rate, which is equivalent to ours Acc_{rob}, and the relationship of these two metrics is depicted in Eq. (9)

TABLE VIII: Robust Accuracy under Orthognal-PGD Attack

Attack	Defense	$L_\infty = 0.01$		$L_\infty = 8/255$	
		Acc_{rob} @FPR5%	Acc_{rob} @FPR50%	Acc_{rob} @FPR5%	Acc_{rob} @FPR50%
Orth	**ContraNet**[*]	*85.4%*	*97.3%*	*63.7%*	*83.0%*
	ContraNet[†]	79.3%	95.7%	38.1%	77.9%
	ContraNet[†]**+ATC**	**93.7%**	**99.2%**	**89.8%**	**94.7%**
	Trapdoor [67]	0%	7.0%	0%	8.0%
	DLA'20 [68]	62.6%	83.7%	0%	28.2%
	SID'21 [72]	6.9%	23.4%	0%	1.6%
	SPAM'19 [46]	1.2%	46.0%	0%	38.0%
Select	**ContraNet**[*]	*85.4%*	*97.0%*	*63.4%*	*83.3%*
	ContraNet[†]	79.4%	95.7%	38.2%	78.2%
	ContraNet[†]**+ATC**	**93.7%**	**99.3%**	**89.7%**	**95.1%**
	Trapdoor	0.2%	49.5%	0.4%	37.2%
	DLA'20	17.0%	55.9%	0%	13.5%
	SID'21	8.9%	50.9%	0%	11.4%

[*] The adaptive objective function contains all losses, $\mathcal{L}_{ContraNet_1}$.
[†] The adaptive objective function only contains $\mathcal{L}_{DMM}$ (the weakest component of ContraNet under adaptive attack).
[‡] Results for baselines are from [7]. ATC we use here is Gowal2020Uncovering [27] from RobustBench [18]

Robust Accuracy against Orthogonal-PGD. As shown in Tab. VIII, ContraNet outperforms four baselines by a considerable margin cross both "Select" and "Orth" attacks for all attack scenarios (ContraNet, ATC + ContraNet). For the worst-case test, ContraNet can still keep 38.1% (Orth) Acc_{rob} when $\ell_\infty = 8/255$. Further, incorporating the adversarial training technique can significantly improve ContraNet's performance from 38.1% (Orth) to 89.7% (Select). The detection mechanism guarantees the ContraNet's robustness against Orthogonal-PGD: since the synthesis's semantic information is highly dependent on the classifier's discriminative features, one can hardly alter this dependency with bounded perturbation upon input image.

F. Summary of the Adaptive Attacks

From the attacker's point of view, Orthogonal-PGD achieved a higher attack success rate than the prior conducted PGD and C&W adaptive attacks (Sec. VI-C and VI-D). However, the attack performance of Orthogonal (Orth) and Selective (Select) are close to each other. To analyze this phenomenon, we check the gradients for both classifier and detector, and find out that gradients of classifier and ContraNet is almost orthogonal to each other. In this case, either Orthogonal or Selective will optimize the AE similarly. Therefore, there is only a slight difference between the two results. The nearly orthogonal gradients provide an explanation for the difficulty of breaching both the ContraNet and the classifier simultaneously.

From the adaptive objective loss function's point of view, focusing on the weakest point of ContraNet can simplify the optimization process, and get a better attack success rate. By comparing the performance of ContraNet on $\mathcal{L}_{DMM}$ and ContraNet on $\mathcal{L}_{ContraNet_1}$ in Tab. VIII, it is clear that $\mathcal{L}_{DMM}$ as the objective function will further decline ContraNet's Acc_{rob} than take all loss items, $\mathcal{L}_{ContraNet_1}$, into consideration. This verifies our intention to replace $\mathcal{L}_{ContraNet_1}$ with $\mathcal{L}_{DMM}$.

The adding of ATC can enhance ContraNet's performance. As shown in Tab. VIII, the performance for ATC + ContraNet is the best for all test settings. Therefore, it is promising to combine ContraNet with ATC to improve robustness. We leave this as future work.

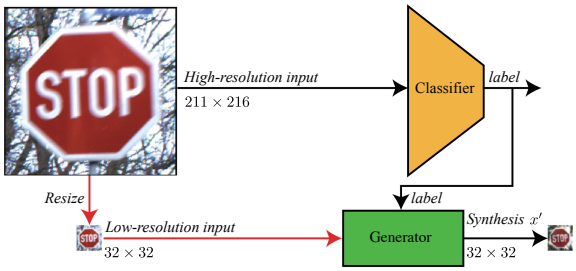

Fig. 13: Workflow of ContraNet protecting the high-resolution classifier. ContraNet can be performed on compressed inputs for a lower computational cost yet effective detection performance.

TABLE IX: Computation and Storage Cost of ContraNet

Modules	Params(M)	FLOPs(G)
DenseNet169 [34]	12.49	0.27
Resnet152 [34]	60.19	11.58
WideResNet-70-16[*][82]	266.80	38.78
WideResNet-28-10[†][82]	36.48	5.25
Encoder	0.99	0.02
Generator	9.42	3.83
Deep Metric Model	5.82	0.18
Discriminator	2.41	0.62
Total ContraNet	10.35	4.65

[*] used by Gowal2020Uncovering in Sec. V-C and VI and Rebuffi2021Fixing_70 in Sec. V-C, from [18].
[†] used by Rebuffi2021Fixing_28 and Sehwag2020Hydra in Sec. V-C, from [18].

VII. Discussions

Computation and memory costs. ContraNet constructs several additional modules (e.g., the conditional generator and the deep metric model) aside from the target DNN model for AE detection, which inevitably incurs extra computation and memory costs. The relative cost of ContraNet is mainly dependent on the size and computational requirement of the protected DNN model. Tab. IX presents the comparison between ContraNet and several classifiers used for CIFAR-10 dataset. As can be seen, in comparison to SOTA classifiers, especially those using adversarial training techniques [18], ContraNet has relatively low storage, *Params*, and compute costs, *FLOPs*. Considering the trend in deep learning is to use an ever-larger pre-trained model for performance enhancement and the severity of AEs for safety-critical applications, we believe the extra computation and memory costs of ContraNet are acceptable.

Dealing with high-resolution images. In our earlier experiments, most of the images in our datasets are low-resolution. To evaluate ContraNet's scalability on high-resolution (HR) images, we construct a new dataset IMAGENET10, a subset of the IMAGENET [22]. We combine the classes with similar semantic information in IMAGENET to be a general class, ending up ten different classes with 5000 images for each class, the same number as CIFAR-10. Containing more data in each class can mitigate the common overfitting issue [85]. We use 256×256 as the input size. For each class, we randomly pick 4800 images for training, 200 images for testing. As for the classifier, we choose ResNet50, which achieves 99% accuracy on the test set. ContraNet trained on IMAGENET10 achieves $Acc_{dec} = 95\%$ on clean samples (FPR$= 5\%$), which is acceptable compared to the classifier's original accuracy. We summarize ContraNet's performance against four white-box attacks in Tab. X.

TABLE X: The performance of ContraNet on IMAGENET10.

Attack Method	Acc_{rob}	TPR@FPR5%
PGD_{ℓ_∞}	93.85%	94.19%
BIM_{ℓ_∞}	92.15%	92.63%
EDA_{ℓ_1}	95.8%	96.16%
$C\&W_{\ell_\infty}$	95.8%	96.11%

TABLE XI: ContraNet protects classifier with high-resolution inputs.

Dataset	IMAGENET-CINIC		GTSRB-BTSC	
Attack Method	Acc_{rob}	TPR@FPR5%	Acc_{rob}	TPR@FPR5%
PGD_{ℓ_∞}	86.30%	87.91%	92.86%	95.81%
BIM_{ℓ_∞}	85.40%	86.85%	93.28%	95.51%
EDA_{ℓ_1}	85.51%	87.06%	92.86%	96.57%
$C\&W_{\ell_\infty}$	86.20%	87.38%	93.28%	97.06%

Comparing to the performance on low-resolution images (refer to Tab. IV- V), ContraNet achieves better results on HR images, thanks to the high quality of images and correspondingly scaled-up model capability. We visualize HR images with their synthesis in Appendix D.

At the same time, the computational overhead might be a concern when applying ContraNet to HR images directly. Note that ContraNet detects AEs based on semantic contradiction, while the semantic information is largely preserved after compression. Therefore, a possible solution to reduce the overhead is to resize the input images to compressed ones before feeding them to ContraNet. Whereas the classifier can remain unchanged. Fig. 13 illustrates this workflow.

We verify the feasibility of this solution by transferring ContraNet trained on CIFAR-10 or GTSRB to the new datasets (IMAGENET-CINIC, GTSRB-BTSC) with high-resolution images and test their performance. For IMAGENET-CINIC, we construct the dataset following the selection procedures of CINIC dataset [20] [7]. IMAGENET-CINIC shares similar classes as CIFAR-10 while contains images of higher resolutions. Then, we train a ResNet152 classifier on IMAGENET-CINIC with 224×224 as the input size. After that, we apply ContraNet trained on 32×32 to protect the classifier. The workflow for detection is shown in Fig. 13. We directly employ the $\mathbf{E}$ and $\mathbf{G}$ of ContraNet trained on CIFAR-10 (32×32) and fine-tune the deep metric model on IMAGENET-CINIC for 15 epochs. For GTSRB-BTSC, since GTSRB dataset already contains partial HR images, we train a classifier with input size of 224×224 directly. We only boost the *test* dataset with extra HR traffic sign images from *Belgium Traffic Sign* [50]. As can be seen in Tab. XI, both the Acc_{rob} and TPR@FPR5% are comparable to the results on CIFAR-10 or GTSRB dataset (refer to Tab. IV-V). Such generalization capability proves it possible for ContraNet to detect AEs on resized images.

To better understand the advantage of image compression, we compare the overhead of ContraNet on different input sizes. The computational cost (FLOPs) and storage consumption (Params) for ContraNet with 32×32 inputs are only 4.25% and 10.16% to those on 256×256 [8] inputs. Further, if compared with the ResNet152 on 224×224 inputs, the numbers will change to 31.99% and 40.63%, respectively. Therefore, image compression effectively reduces the overhead of ContraNet without lowering AE detection capabilities.

Data pre-processing attack. There is a kind of adversarial attack targeting at the *data pre-processing* stage. For example, the *Image-Scaling* attacks can obtain an output image that resembles the target image after downscaling the input image [59], [78]. The image-scaling attacks are operated on the image resize step, which may alter the semantic information of the image fed to the DNN model. Such impact on DNN system is preserved in all subsequent steps.

ContraNet cannot detect such kinds of AEs because there is no semantic contradiction between the input of DNN and the discriminative features from the classifier. However, the defender can easily fix the vulnerability caused by an image-scaling attack due to the explicitly working mechanism of the image-scaling function [78]. Such a defensive mechanism can work seamlessly with ContraNet without affecting each other.

Limitations. One of the limitations of ContraNet is that it has difficulty in differentiating classes that are inherently similar in semantics. For example, there are more than one hundred types of dogs in ImageNet dataset [22], and the synthetic images for some of them could be pretty similar.

However, this is usually not a security concern. On the one hand, from the adversary's perspective, an AE attack targeted on a class similar to its original is usually of minor severity, e.g., from one type of dog to another type. On the other hand, if a particular class needs to be secured, its semantic meaning should be identifiable from others[9], e.g., the "Stop" sign.

VIII. CONCLUSION AND FUTURE WORK

We proposed a novel AE detection framework, ContraNet, which focuses on identifying the intrinsic contradiction between AE's semantic meaning (in human eyes) and its DNN-extracted discriminative features (in DNN's eyes). We empirically evidence the effectiveness of ContraNet via adequate white-box and adaptive attacks. Experimental results show that ContraNet outperforms state-of-the-art AE detectors by a large margin. We have also shown that ContraNet can be combined with adversarial training techniques to enhance DNN model robustness further.

While most AE attacks and defenses focus on image classifiers and ContraNet is also applied in this context, adversarial examples for other DNN models and tasks have proliferated recently. We believe the basic concept of ContraNet still holds, and we plan to extend it to detect these new types of AEs.

ACKNOWLEDGEMENTS

We would like to acknowledge the contributions of Miss Bo Luo for her early exploration of this topic, and we thank the anonymous reviewers for their valuable comments.

This work was supported in part by General Research Fund of Hong Kong Research Grants Council (RGC) under Grant No. 14203521, No. 14205420, and No. 14205018.

[7] CINIC [20] provides a class mapping from CIFAR-10 to IMAGENET. We follow the mapping and fetch HR images from IMAGENET.

[8] cGAN typically works with an integral power of 2 as the input size.

[9] Otherwise, it may be misclassified even without an attack, causing security concerns.

REFERENCES

[1] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International Conference on Machine Learning*. PMLR, 2018.

[2] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 387–402.

[3] A. Board, *Stochastic modelling and applied probability*. Springer, 2005.

[4] J. Breier, X. Hou, D. Jap, L. Ma, S. Bhasin, and Y. Liu, "Practical fault attack on deep neural networks," in *ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 2204–2206.

[5] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *International Conference on Learning Representations*, 2019.

[6] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, no. 04, pp. 669–688, 1993.

[7] O. Bryniarski, N. Hingun, P. Pachuca, V. Wang, and N. Carlini, "Evading adversarial example detection defenses with orthogonal projected gradient descent," *arXiv preprint arXiv:2106.15023*, 2021.

[8] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow, "Thermometer encoding: One hot way to resist adversarial examples," in *International Conference on Learning Representations*, 2018.

[9] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. J. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

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

[11] ——, "MagnNet and 'efficient defenses against adversarial attacks' are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[12] ——, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy*, 2017, pp. 39–57.

[13] F. Carrara, R. Becarelli, R. Caldelli, F. Falchi, and G. Amato, "Adversarial examples detection in features distance spaces," in *Proceedings of the European Conference on Computer Vision Workshops*, 2018.

[14] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "EAD: Elastic-net attacks to deep neural networks via adversarial examples," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[15] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, p. 15–26.

[16] T. Chen, S. Liu, S. Chang, Y. Cheng, L. Amini, and Z. Wang, "Adversarial robustness: From self-supervised pre-training to fine-tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 699–708.

[17] M. Cococcioni, F. Rossi, E. Ruffaldi, S. Saponara, and B. Dupont de Dinechin, "Novel arithmetics in deep neural networks signal processing for autonomous driving: Challenges and opportunities," *IEEE Signal Processing Magazine*, pp. 97–110, 2021.

[18] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, "Robustbench: a standardized adversarial robustness benchmark," *arXiv preprint arXiv:2010.09670*, 2020.

[19] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International conference on machine learning*. PMLR, 2020, pp. 2206–2216.

[20] L. N. Darlow, E. J. Crowley, A. Antoniou, and A. J. Storkey, "Cinic-10 is not imagenet or cifar-10," 2018.

[21] H. de Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. Courville, "Modulating early visual processing by language," in *Conference on Neural Information Processing Systems*, 2017, pp. 1–14.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "ImageNet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[23] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[24] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," in *5th International Conference on Learning Representations*, 2017.

[25] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018.

[26] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *3rd International Conference on Learning Representations*, 2015.

[27] S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli, "Uncovering the limits of adversarial training against norm-bounded adversarial examples," *arXiv preprint arXiv:2010.03593*, 2020.

[28] C. He, B. B. Zhu, X. Ma, H. Jin, and S. Hu, "Feature-indistinguishable attack to circumvent trapdoor-enabled defense," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 3159–3176.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[30] W. He, J. Wei, X. Chen, N. Carlini, and D. Song, "Adversarial example defenses: Ensembles of weak defenses are not strong," in *Proceedings of the 11th USENIX Conference on Offensive Technologies*, 2017.

[31] Z. He, A. S. Rakin, J. Li, C. Chakrabarti, and D. Fan, "Defending and harnessing the bit-flip based adversarial weight attack," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 095–14 103.

[32] H. Hirano, A. Minagi, and K. Takemoto, "Universal adversarial attacks on deep neural networks for medical image classification," *BMC medical imaging*, vol. 21, no. 1, pp. 1–13, 2021.

[33] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International workshop on similarity-based pattern recognition*. Springer, 2015, pp. 84–92.

[34] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[35] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren, "Secure, privacy-preserving and federated machine learning in medical imaging," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 305–311, 2020.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[37] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015.

[38] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Master's thesis, University of Tront*, 2009.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[40] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *International Conference on Learning Representations workshop*, 2017.

[41] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[42] M. Lécuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," *IEEE Symposium on Security and Privacy*, pp. 656–672, 2019.

[43] P. Li, J. Yi, B. Zhou, and L. Zhang, "Improving the robustness of deep neural networks via adversarial training with triplet loss," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, S. Kraus, Ed. ijcai.org, 2019, pp. 2909–2915.

[44] J. Lin, L. Xu, Y. Liu, and X. Zhang, "Composite backdoor attack for deep neural network by mixing existing benign features," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 113–131.

[45] X. Ling, S. Ji, J. Zou, J. Wang, C. Wu, B. Li, and T. Wang, "Deepsec: A uniform platform for security analysis of deep learning model," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 673–690.

[46] J. Liu, W. Zhang, Y. Zhang, D. Hou, Y. Liu, H. Zha, and N. Yu, "Detection based defense against adversarial examples from the steganalysis point of view," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4825–4834.

[47] Y. Liu, L. Wei, B. Luo, and Q. Xu, "Fault injection attack on deep neural network," in *IEEE/ACM International Conference on Computer-Aided Design*, 2017, pp. 131–138.

[48] A. Madry, A. Athalye, D. Tsipras, L. Engstrom, D. Wagner, N. Carlini, P. Liang, and Z. Kolter, "RobustML," https://www.robust-ml.org/, accessed: 2021-07-15.

[49] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *6th International Conference on Learning Representations*, 2018.

[50] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool, "Traffic sign recognition — how far are we from the solution?" in *The International Joint Conference on Neural Networks (IJCNN)*, 2013.

[51] D. Meng and H. Chen, "MagNet: A Two-Pronged Defense against Adversarial Examples," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 135–147.

[52] T. Miyato and M. Koyama, "cGANs with Projection Discriminator," in *International Conference on Learning Representations*, 2018.

[53] N. Naren, V. Chamola, S. Baitragunta, A. Chintanpalli, P. Mishra, S. Yenuganti, and M. Guizani, "Iomt and dnn-enabled drone-assisted covid-19 screening and detection framework for rural areas," *IEEE Internet of Things Magazine*, vol. 4, no. 2, pp. 4–9, 2021.

[54] R. Pang, H. Shen, X. Zhang, S. Ji, Y. Vorobeychik, X. Luo, A. X. Liu, and T. Wang, "A tale of evil twins: Adversarial inputs versus poisoned models," *ACM SIGSAC Conference on Computer and Communications Security*, 2020.

[55] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu, "Improving adversarial robustness via promoting ensemble diversity," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97. PMLR, 2019, pp. 4970–4979.

[56] T. Pang, H. Zhang, D. He, Y. Dong, H. Su, W. Chen, J. Zhu, and T.-Y. Liu, "Adversarial training with rectified rejection," *arXiv preprint arXiv:2105.14785*, 2021.

[57] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 2017, pp. 506–519.

[58] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*, 2016, pp. 372–387.

[59] E. Quiring, D. Klein, D. Arp, M. Johns, and K. Rieck, "Adversarial preprocessing: Understanding and preventing image-scaling attacks in machine learning," in *Proc. of USENIX Security Symposium*, 2020.

[60] S.-A. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. Mann, "Fixing data augmentation to improve adversarial robustness," *arXiv preprint arXiv:2103.01946*, 2021.

[61] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-GAN: Protecting classifiers against adversarial attacks using generative models," in *International Conference on Learning Representations*, 2018.

[62] S. Santurkar, D. Tsipras, B. Tran, A. Ilyas, L. Engstrom, and A. Madry, "Image synthesis with a single (robust) classifier," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 1262–1273.

[63] U. Sara, M. Akter, and M. S. Uddin, "Image quality assessment through fsim, ssim, mse and psnr—a comparative study," *Journal of Computer and Communications*, vol. 7, no. 3, pp. 8–18, 2019.

[64] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[65] V. Sehwag, S. Wang, P. Mittal, and S. Jana, "HYDRA: Pruning adversarially robust neural networks," in *Conference on Neural Information Processing Systems*, 2020.

[66] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!" in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 3358–3369.

[67] S. Shan, E. Wenger, B. Wang, B. Li, H. Zheng, and B. Zhao, "Gotta catch'em all: Using honeypots to catch adversarial attacks on neural networks," *ACM SIGSAC Conference on Computer and Communications Security*, 2020.

[68] P. Sperl, C.-Y. Kao, P. Chen, X. Lei, and K. Böttinger, "DLA: dense-layer-analysis for adversarial example detection," in *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020.

[69] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, 2012.

[70] J. Steinhardt, P. W. Koh, and P. Liang, "Certified defenses for data poisoning attacks," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 3520–3532.

[71] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[72] J. Tian, J. Zhou, Y. Li, and J. Duan, "Detecting adversarial examples from sensitivity inconsistency of spatial-transform domain," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9877–9885.

[73] V. Venceslai, A. Marchisio, I. Alouani, M. Martina, and M. Shafique, "Neuroattack: Undermining spiking neural networks security through externally triggered bit-flips," *International Joint Conference on Neural Networks (IJCNN)*, 2020.

[74] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 707–723.

[75] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1386–1393.

[76] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, and Q. Gu, "Improving adversarial robustness requires revisiting misclassified examples," in *International Conference on Learning Representations*, 2020.

[77] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?" in *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[78] Q. Xiao, Y. Chen, C. Shen, Y. Chen, and K. Li, "Seeing is not believing: Camouflage attacks on image scaling algorithms," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 443–460.

[79] W. Xu, D. Evans, and Y. Qi, "Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks," in *Proceedings 2018 Network and Distributed System Security Symposium*. San Diego, CA: Internet Society, 2018.

[80] F. Yao, A. S. Rakin, and D. Fan, *DeepHammer: Depleting the Intelligence of Deep Neural Networks through Targeted Chain of Bit Flips*, 2020.

[81] Y. Yaz, C.-S. Foo, S. Winkler, K.-H. Yap, G. Piliouras, V. Chandrasekhar *et al.*, "The unusual effectiveness of averaging in gan training," in *International Conference on Learning Representations*, 2018.

[82] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[83] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7472–7482.

[84] J. Zhang, J. Zhu, G. Niu, B. Han, M. Sugiyama, and M. Kankanhalli, "Geometry-aware instance-reweighted adversarial training," in *International Conference on Learning Representations*, 2021.

[85] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han, "Differentiable augmentation for data-efficient gan training," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

APPENDIX

A. White-box Attack Configurations

The hyperparameters we used for the four typical untargeted attacks are as follow. For PGD and BIM, we set the iteration step to 50. We set the adversarial perturbation for PGD and BIM to 8/255 for ℓ_∞ norm , and 1 for ℓ_2 norm. For C&W and EAD, we set the binary search steps to 9, the step size to 0.01, optimization iteration to 1000, and the attack confidence to 0. For C&W, we employ *Adam* [36] as the optimizer.

B. ROC Curves with AUC under ℓ_2-norm

We generate the ROC curves together with AUC on CIFAR-10 under various attacks under ℓ_2-norm, as shown in Fiugre 14. We can observe that the TPR can rapidly increase when the FPR is increased by a modest amount. This demonstrates that ContraNet is capable of detecting AE with high accuracy while having a negligible influence on normal samples. Additionally, AUC ContraNet's AUCs, which are used to indicate the overall performance for various attacks are all close to 1, indicating that ContraNet performs admirably at AE detection.

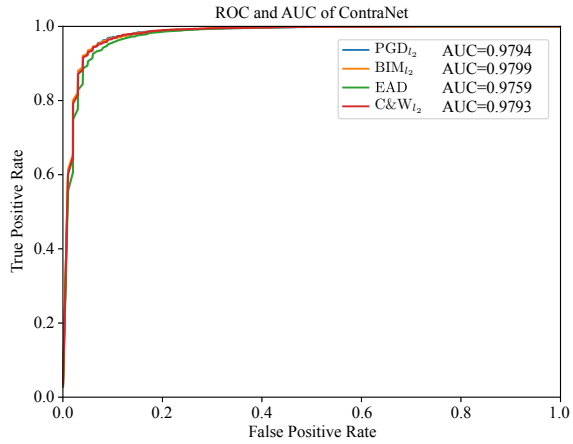

Fig. 14: ROCs are generated under four attacks PGD$_{\ell_2}$, BIM$_{\ell_2}$, EAD, C&W$_{\ell_2}$, along with corresponding AUC (higher is better) demonstrated in legend block.

C. Model architectures

In this section, we give a detailed description of the model architecture we used when implementing ContraNet. The architecture of the Generator is shown in Fig. 15. The architecture of the Discriminator is shown in Fig. 16. The architecture of Encoder is only convolutional neural networks. We show the parameters for each layer of the Encoder and MLP in Tab. XII and Tab. XIII.

D. More Synthetic Images

We demonstrate empirically that the semantic information included in the synthetic images generated by ContraNet is strongly dependent on the associated labels. Fig. 17 shows more synthetic images on MNIST, GTSRB and CIFAR-10. Fig. 18 shows more synthetic images on IMAGENET10.

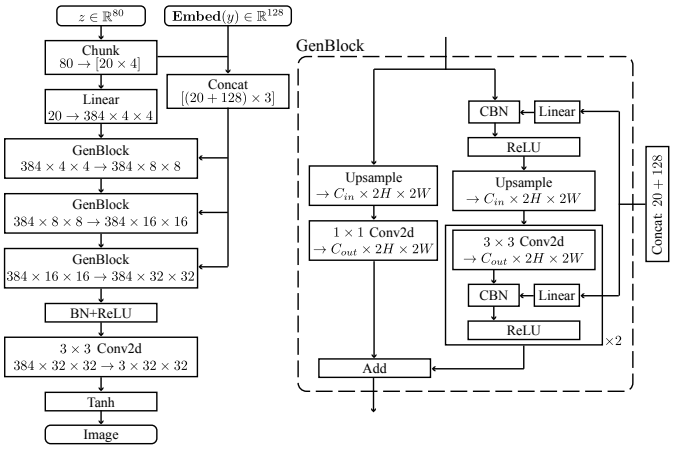

Fig. 15: Generator's Architecture of ContraNet.

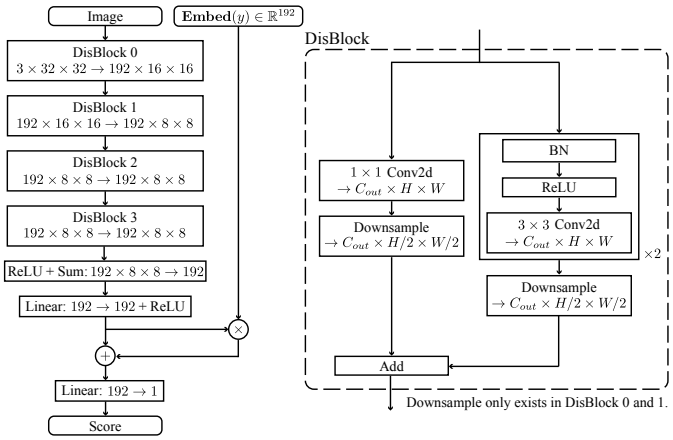

Fig. 16: Discriminator's Architecture of ContraNet.

TABLE XII: Parameters for Each Layer of the Encoder

Input shape	$32 \times 32 \times 3$	
Operator	parameters	Output shape
Conv2d	channel: $3 \to 64$	$16 \times 16 \times 64$
ReLU	-	$16 \times 16 \times 64$
Conv2d	channel: $64 \to 128$	$8 \times 8 \times 128$
BN + ReLU	-	$8 \times 8 \times 128$
Conv2d	channel: $128 \to 256$	$4 \times 4 \times 256$
BN + ReLU	-	$4 \times 4 \times 256$
Conv2d	channel: $256 \to 80$, no padding	$1 \times 1 \times 80$
Linear	shape: $80 \to 80$	80

◇ Conv2d use kernel size $= 4$, stride $= 2$, padding $= 1$ as default.

TABLE XIII: Parameters for Each Layer of the MLP

Input	$\{\mathbf{en}, \mathbf{ep}\} \in \mathbb{R}^{2560}$
Operator	Parameters
Linear	shape: $2560 \to 512$
Dropout + ReLU	drop rate: 0.1
Linear	shape: $512 \to 64$
Dropout+ReLU	drop rate: 0.1
Linear	shape: $64 \to 2$

Fig. 17: Synthetic images on MNIST, GTSRB and CIFAR-10. For each task we randomly sample 64 clean images from the test set, and generate the synthetic images under their groundtruth labels, the first and the second label in order. As can be observed, the semantic information contained in the synthetic images is significantly dependent on the given label.

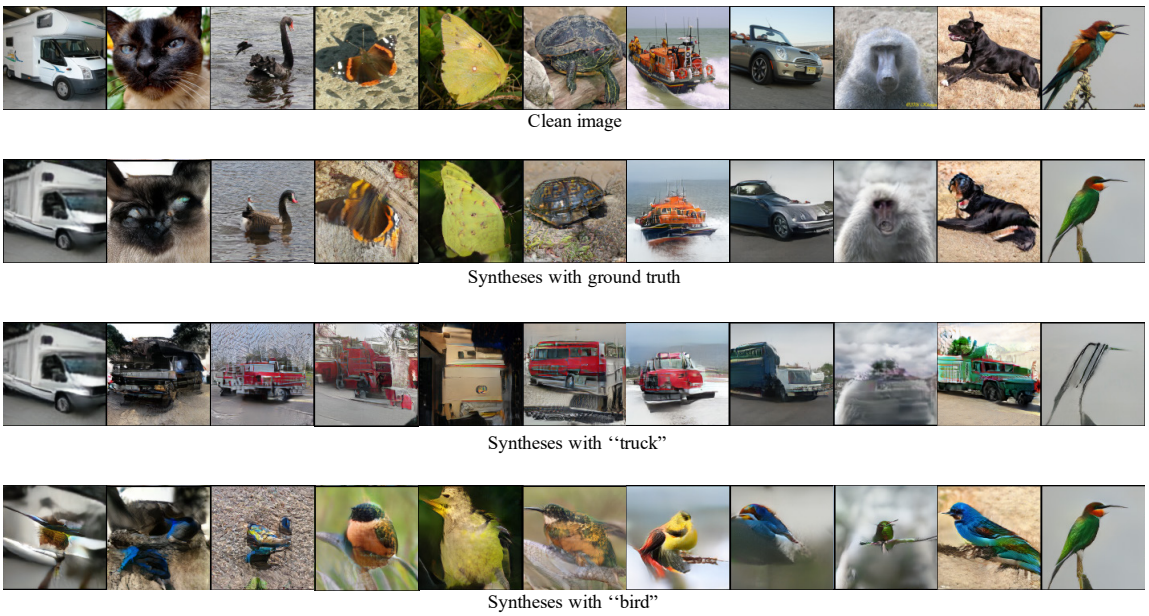

Fig. 18: High-resolution synthetic images on IMAGENET10. As can be observed, the semantic information contained in the synthetic images is significantly dependent on the given label.