DEFLECTING ADVERSARIAL ATTACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

There has been an ongoing cycle where stronger detection mechanisms and defenses against adversarial attacks are subsequently broken by a more advanced defense-aware attack. We present a new approach, which we argue is a step towards ending this cycle by *deflecting* adversarial attacks, i.e. by forcing the attacker to produce an input which semantically resembles the attack's target class. To this end, we first propose a stronger defense mechanism based on capsule networks which combines three detection mechanisms to achieve state-of-the-art detection performance on both standard and defense-aware attacks. We then show that undetected attacks against our defense are often classified as the adversarial target class by performing a human study where participants are asked to label the class of images produced by the attack. These attack images thus can no longer be called adversarial, as our network classifies them the same way as humans do.

1 INTRODUCTION

Adversarial attacks have been the subject of constant research since they were first discovered (Szegedy et al., 2013; Goodfellow et al., 2014; Kurakin et al., 2016; Madry et al., 2017). Most of this research has been focused on the creation of more robust models to **defend** against adversarial attacks (Song et al., 2017; Madry et al., 2017; Yang et al., 2019), or stronger attack algorithms that break these defenses (Madry et al., 2017; Carlini & Wagner, 2017b; Chen et al., 2018; Athalye et al., 2018). After several iterations of defense creating and breaking, some research focused on adversarial attack **detection** (Grosse et al., 2017; Feinman et al., 2017; Metzen et al., 2017; Lee et al., 2018; Qin et al., 2019; Roth et al., 2019). Detection algorithms aim to distinguish adversarial attacks from real data, instead of attempting to correctly classify such inputs. However, this strategy fell into the same creating/breaking cycle. Many state-of-the-art methods (Roth et al., 2019; Ma et al., 2018; Lee et al., 2018) claiming to detect adversarial attacks were broken shortly after publication with a defense-aware attack (Hosseini et al., 2019; Carlini & Wagner, 2017a; Athalye et al., 2018). We attempt to get ahead of this cycle by focusing on the **deflection** of adversarial attacks: If the result of the adversarial optimization of an image looks to a human like the adversarial target class rather than its original class, then the image can hardly be called adversarial anymore. We call such attacks "deflected". Some examples are shown in Figure 1. In this paper, we propose a network and detection mechanism that either detects attacks accurately or, for undetected attacks, forces the attacker to produce images that resemble the target class, thereby deflecting them.

In this paper we make use of capsule networks and class conditional capsule reconstruction networks (Sabour et al., 2017; Qin et al., 2019). This architecture is made up of two components: A capsule classification network that classifies the input, and a reconstruction network that reconstructs the input conditioned on the pose parameters of the predicted capsule. Apart from the classification loss and ℓ_2 reconstruction loss used in (Sabour et al., 2017; Qin et al., 2019), we introduce an extra cycle-consistency training loss which constrains a classification of the winning capsule reconstruction to be the same as the classification of the original input. This new auxiliary training loss encourages the reconstructions to more closely match the class-conditional distribution and helps the model detect and deflect adversarial attacks.

In addition, we propose two new attack-agnostic detection methods based on the discrepancy between the winning-capsule reconstruction of clean and adversarial inputs. We find that a detection method that combines ours with the one proposed by Qin et al. (2019) performs best. We show that this method can accurately detect white-box and black-box attacks based on three different distortion metrics (EAD (Chen et al., 2018), CW (Carlini & Wagner, 2017b) and PGD (Madry et al., 2017)) on

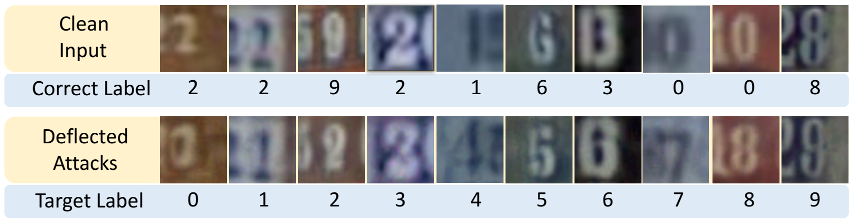

Figure 1: Deflected adversarial attacks on the SVHN dataset. These images were generated by a defense aware attack and the maximal adversarial perturbation is bounded by 16/255.

both the SVHN and CIFAR-10 datasets. Following the suggestions in (Athalye et al., 2018; Carlini & Wagner, 2017a), we also propose defense-aware attacks for our new detection method. We find that our detection methods significantly outperform state-of-the-art methods on defense-aware attacks. Finally, we perform a human study to verify that many of the undetected adversarial attacks against our model have been successfully deflected, i.e. adversarial images from both defense-aware and standard attacks against our detection mechanism are frequently classified by humans as the target class. In contrast, successful attacks against baseline models do not have this property.

To summarize, our main contributions are as follows:

- We introduce the notion of **deflected adversarial attacks**, which presents a step towards ending the battle between defenses and attacks.
- We propose a new cycle-consistency loss which trains a CapsNet to encourage the winning-capsule reconstruction to closely match the class-conditional distribution and show that this can help detect and deflect adversarial attacks.
- We introduce two attack-agnostic detection methods based on the discrepancy between the winning-capsule reconstruction of the clean and the adversarial input, and design a defense-aware attack to specifically attack our detection mechanisms.
- Extensive experiments on SVHN and CIFAR-10 show that our detection mechanism can achieve state-of-the-art performance in detecting white-/black-box standard and defense-aware attacks.
- We perform a human study to show that our approach, unlike previous methods, is able to deflect a large percentage of the undetected adversarial attacks.

2 Network Architecture

In order to design a model that is strong enough to deflect adversarial attacks, we build our network based on CapsNet (Sabour et al., 2017). Figure 2 shows the pipeline of our network architecture. The final layer of our classifier is a Capsule layer ("CapsLayer" for short) which includes both class capsules and background capsules. These capsules are intended to encode feature attributes corresponding to the class and the background respectively. Given an input $\boldsymbol{x}$, the output of a CapsLayer is a prediction $f(\boldsymbol{x})$ and a pose parameter $\boldsymbol{v}$ for all the classes and the background, where $\boldsymbol{v}_i$ denotes the pose parameter for class i. As in the initial Capsules proposed in (Sabour et al., 2017), the magnitude of the activation vector of a capsule encodes the existence of an instance of the class and the orientation of the activation vector encodes instantiation parameters of the instance, such as its pose. Therefore, the magnitudes of the capsules' activations are used to perform classification while the activation vector of the winning class capsule together with the activation vectors of the background capsules are used as the input to the reconstruction network. We use $r(\boldsymbol{v}_{i=f(\boldsymbol{x})})$ and $r(\boldsymbol{v}_{i\neq f(\boldsymbol{x})})$ to represent the reconstruction from the winning capsule and a losing capsule respectively. The reconstruction network uses the activations of all the background capsules as well as the activation of one class capsule but we omit this to simplify the notation. More details of the network architecture and implementation details used in this paper are provided in Section A and B of the Appendix.

Cycle-consistent Winning Capsule Reconstructions The CapsNet (Sabour et al., 2017) is trained with two loss terms: a marginal loss for the classification and an ℓ_2 reconstruction loss. To encourage the reconstruction to more closely match the class conditional distribution and help the model detect and deflect adversarial attacks, we additionally incorporate an extra cycle-consistency loss ℓ_{cyc} which

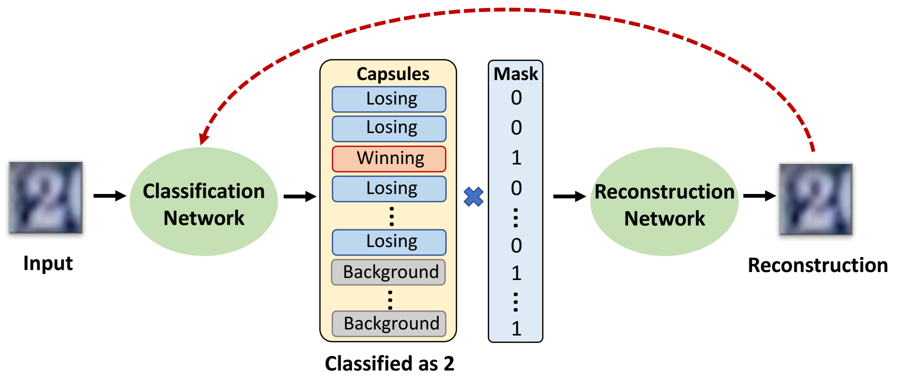

Figure 2: The network architecture with cycle-consistent winning capsule reconstructions.

constrains the reconstruction from the winning capsule to be classified as the same class as the input, formulated as:

$$\ell_{cyc} = \ell_{net}(f(r(\boldsymbol{v}_{i=f(\boldsymbol{x})})), f(\boldsymbol{x})), \tag{1}$$

where ℓ_{net} is the cross-entropy loss function and $i \in \{0, 1, \dots, n\}$, n denotes the number of classes in the dataset. This can be achieved by feeding the reconstruction corresponding to the winning capsule back into the classification network, shown as the dotted red line in Figure 2. This extra training loss together with our Cycle-consistent Detector (introduced in Section 3) can help detect adversarial attacks. In addition, since the winning-capsule reconstructions are optimized to more closely match the class conditional data distribution, it becomes easier for our model to deflect adversarial attacks.

3 Detection Methods

In this paper, we will use three reconstruction-based detection methods to detect standard attacks. They are: **G**lobal **T**hreshold **D**etector (GTD), first proposed in Qin et al. (2019), **L**ocal **B**est **D**etector (LBD) and **C**ycle-**C**onsistency **D**etector (CCD).

Global Threshold Detector When the input is adversarially perturbed, the classification given to the input may be incorrect, but the reconstruction is often blurry and therefore the distance between the adversarial input and the reconstruction is larger than that would be expected from normal input. This allows us to detect the input as adversarial with the Global Threshold Detector. This method, proposed in Qin et al. (2019), measures the reconstruction error between the input and its reconstruction from the winning capsule. If the reconstruction error is greater than a global threshold θ, that is:

$$\|r(\boldsymbol{v}_{i=f(\boldsymbol{x})}) - \boldsymbol{x}\|_2 > \theta, \tag{2}$$

then the input is flagged as an adversarial example.

Local Best Detector When the input is a clean image, the reconstruction error from the winning capsule is smaller than that of the losing capsules, where an example is shown in the first row of Figure 3. This is likely because the ℓ_2 reconstruction objective only minimizes the reconstruction from the winning capsule during training. However, when the input is an adversarial example, the reconstruction from the capsule corresponding to the correct label can be even closer to the input compared to the reconstruction corresponding to the winning capsule (see the second row in Figure 3). Therefore, we propose the "Local Best Detector" (LBD) to detect such adversarial images whose reconstruction error from the winning capsule is not the smallest, that is:

$$\arg\min_j \|r(\boldsymbol{v}_j) - \boldsymbol{x}\|_2 \neq f(\boldsymbol{x}), \quad j \in \{0, 1, \dots, n\}, \tag{3}$$

where n is the number of classes in the dataset.

Cycle-Consistency Detector If the input is a clean image, the reconstruction from the winning capsule will resemble the input. Our model should ideally assign the same class to the reconstruction of the winning capsule as the clean input. This behavior is reinforced by training with the cycle-consistency loss. For example, as shown in Figure 3 both the clean input and its winning-capsule reconstruction are classified as 4. However, when the input is an adversarial example which is

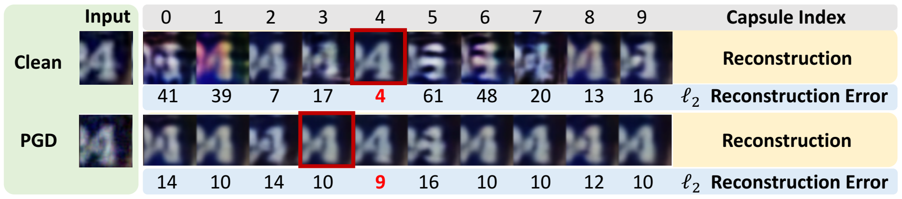

Figure 3: An example of a clean input, an adversarial example generated via a PGD attack, and the reconstructions for the clean and adversarial inputs from each class capsule. The reconstruction corresponding to the winning capsule is surrounded by a red box. Under each reconstruction is its ℓ_2 reconstruction error; the smallest reconstruction error is highlighted in red. Both the clean input and its winning capsule reconstruction are classified as '4'. The PGD attack is classified as the target class '3' but its winning capsule reconstruction is classified as '4'.

perceptually indistinguishable from the clean image but causes the model to predict the target class, the reconstruction of the winning capsule often appears closer to the clean input and/or is blurry. As a result, the reconstruction of the winning capsule is often not classified as the target class. As shown in Figure 3, the adversarial input has been classified as the target class "3" while the reconstruction corresponding to the winning capsule is classified as "4". Therefore, the Cycle-Consistency Detector (CCD) is designed to flag the input as an adversarial example if the input $\boldsymbol{x}$ and its reconstruction of the winning capsule $r(\boldsymbol{v}_{i=f(\boldsymbol{x})})$ are not classified as the same class:

$$f(r(\boldsymbol{v}_{i=f(\boldsymbol{x})})) \neq f(\boldsymbol{x}). \tag{4}$$

In this paper, we use these three detectors together to detect adversarial examples. In other words, we flag any input as adversarial if it is classified as adversarial by any of the detection mechanisms. As a result, an adversarial input can only go undetected if it passes all three detection mechanisms.

4 The Defense-Aware CC-PGD Attack

In order for an attack mechanism to generate an adversarial example $\boldsymbol{x}' = \boldsymbol{x} + \Delta$ (where Δ is a small adversarial perturbation) that can both cause a misclassification and is not detected by our detection mechanisms, the constructed adversarial attack must:

- successfully fool the classifier: $f(\boldsymbol{x}') = t$ and $f(\boldsymbol{x}) \neq t$, where t is the target class.
- avoid being detected by the Global Threshold Detector (GTD), the attack needs to constrain the reconstruction of the winning capsule to be close to the input.[1]
- fool the Local Best Detector (LBD), the attack should encourage the reconstructions from all the losing capsules to be far away from the input to ensure the reconstruction error of the winning capsule is the smallest.
- circumvent the Cycle-Consistency Detector (CCD) by fooling the classifier into making the target prediction when it is fed the winning-capsule reconstruction of the adversarial input, that is: $f(r(\boldsymbol{v}_{i=f(\boldsymbol{x}')})) = f(\boldsymbol{x}') = t$.

To generate such an attack, we follow Qin et al. (2019) and devise attacks which consist of two stages at each gradient step. The first stage attempts to fool the classifier by following a standard attack (e.g., a standard PGD attack) which follows the gradient of the cross-entropy loss function with respect to the input. Then, in the second stage, we focus on fooling the detection mechanisms by taking the reconstruction error and cycle-consistency into consideration. This can be formulated as minimizing the reconstruction loss ℓ_r, which consists of three components: the reconstruction loss corresponding to the Global Threshold Detector ℓ_g, the reconstruction loss corresponding to the Local Best Detector ℓ_l and the cycle-consistency classification loss corresponding to the Cycle-Consistency Detector ℓ_{cyc}.

[1]Here we assume that the attack is not aware of the specific value of the global threshold p used in the defense.

Specifically, the reconstruction loss is defined as:

$$\begin{aligned}\ell_r(\boldsymbol{x}') &= \alpha_1 \cdot \ell_g(\boldsymbol{x}') + \alpha_2 \cdot \ell_l(\boldsymbol{x}') + \alpha_3 \cdot \ell_{cyc}(\boldsymbol{x}') \\ &= \alpha_1 \cdot \|r(\boldsymbol{v}_{i=f(\boldsymbol{x}')}) - \boldsymbol{x}'\|_2 - \alpha_2 \cdot \frac{\sum_{k \neq f(\boldsymbol{x}')}^{n} \|r(\boldsymbol{v}_k) - \boldsymbol{x}'\|_2}{n-1} \\ &\quad + \alpha_3 \cdot \ell_{net}(f(r(\boldsymbol{v}_{i=f(\boldsymbol{x}')})), f(\boldsymbol{x}'))\end{aligned} \tag{5}$$

where $\boldsymbol{x}' = \boldsymbol{x} + \Delta$ is the adversarial example, n is the number of the classes in the dataset, $\|r(\boldsymbol{v}_{i=f(\boldsymbol{x}')}) - \boldsymbol{x}'\|_2$ is the winning-capsule reconstruction error and $\|r(\boldsymbol{v}_{k\neq f(\boldsymbol{x}')}) - \boldsymbol{x}'\|_2$ is the losing-capsule reconstruction error. The hyperparameters α_1, α_2 and α_3 are used to balance the importance of attacking each detector. Then, the adversarial perturbation can be updated in the second stage as:

$$\Delta \leftarrow \text{clip}_{\epsilon_\infty}(\Delta - c \cdot \text{sign}(\nabla_\Delta(\ell_r(\boldsymbol{x} + \Delta))), \tag{6}$$

where ϵ_∞ is the ℓ_∞ norm bound and c is the step size in each iteration.

5 Experiments

Now that we have proposed our new defense model, we first verify its detection performance on the SVHN and CIFAR-10 datasets on a variety attacks. Then we use a human study to demonstrate that our model frequently causes attacks to be deflected.

5.1 Evaluation Metrics and Datasets

In this paper, we use **Accuracy** to represent the proportion of clean examples that are correctly classified. To measure the ability of attacks, we use **Success Rate** which is defined as the proportion of adversarial examples that successfully fool the classifier into making the targeted prediction. In order to evaluate the performance of different detection mechanisms, we report both **False Positive Rate (FPR)** and **Undetected Rate**. The False Positive Rate is the proportion of clean examples that are flagged as an adversarial example by the detection mechanism while the Undetected Rate, first proposed in (Qin et al., 2019), denotes the proportion of adversarial examples that successfully fool the classifier and go undetected. Finally, we perform a human study in Section 6 in order to show that our model is able to effectively deflect adversarial attacks.

We test our models on the SVHN (Netzer et al., 2011) and CIFAR-10 datasets (Krizhevsky, 2009). The classification accuracy on the clean test set is 96.5% on SVHN and 92.6% on CIFAR-10.

5.2 Adversarial Attacks and Threat Model

Following the suggestions in (Carlini et al., 2019), we test our attack-agnostic detection mechanisms on three standard targeted attacks based on different distance metrics: ℓ_1 norm-based EAD (Chen et al., 2018), ℓ_2 norm-based CW (Carlini & Wagner, 2017b), and ℓ_∞ norm-based PGD (Madry et al., 2017). In addition, we follow the suggestions in (Carlini & Wagner, 2017a) to report the performance of our detection mechanisms against defense-aware attacks. In our case, the defense-aware attack which we call CC-PGD takes all the detectors into account. For the ℓ_∞ norm-based attacks, we set the maximal perturbation ϵ_∞ to be 16/255 on SVHN and 8/255 on CIFAR-10 as is typically used (Buckman et al., 2018; Madry et al., 2017).

In this paper, we consider two commonly used threat models: white-box and black-box. For white-box attacks, the adversary has the full knowledge of the network architecture and parameters and is allowed to construct the adversarial attack by computing the gradient of the input. In the black-box setting, the adversary is aware of the network architecture of the target model but does not have direct access to the parameters.

5.3 Ablation Study for Detection Methods

In this section, we study the effectiveness of our proposed detection mechanisms: Local Best Detector (LBD) and Cycle-Consistency Detector (CCD) and compare them with the Global Threshold Detector (GTD) from (Qin et al., 2019). Since the False Positive Rate (FPR) of clean input flagged by the Global Threshold Detector (GTD) varies as the chosen global threshold, in Figure 4 we plot the undetected rate of white-box adversarial attacks flagged by different detectors versus the False Positive Rate (FPR) of the clean input. The global threshold θ is chosen from the range [0, 20] with a

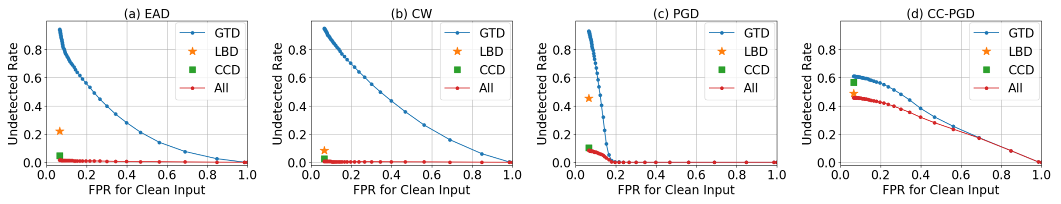

Figure 4: The Undetected Rate of different detectors for white-box attacks versus False Positive Rate (FPR) for clean input on the SVHN dataset. "All" denotes GTD, LBD and CCD are all used to detect adversarial attacks. The better detection mechanism has a smaller FPR for clean input and smaller undetected rate for attacks.

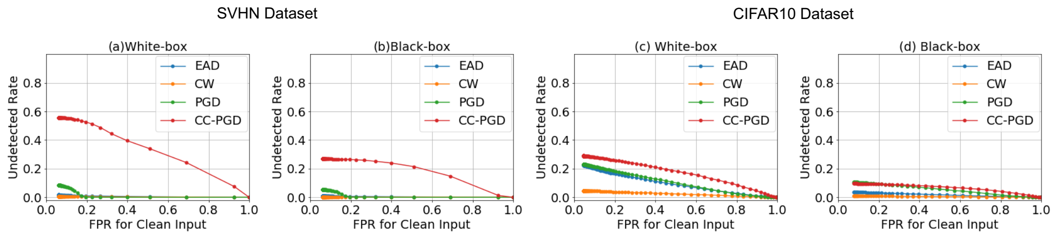

Figure 5: The Undetected Rate for white-box and black-box attacks versus False Positive Rate (FPR) for clean input on the SVHN and CIFAR-10 datasets. The strongest attack has the largest area under the line.

step size of 0.4. In Figure 4, we can clearly see that: 1) A single Global Threshold Detector (GTD) proposed in (Qin et al., 2019) is not enough to effectively detect adversarial attacks. 2) In standard attacks, the CCD is the most effective detector at a low False Positive Rate. However, it becomes less effective than LBD when the inputs are created with the defense-aware CC-PGD attack which is designed to specifically attack the three detection mechanisms. 3) In all the attacks, the combination of all three detectors always performs the best. Therefore, we only report the performance of the undetected rate of the combination of all three detectors in the following experiments.

5.4 Detection of White-box Attacks

Before showing that our defense produces deflective attacks, we must first validate that it improves detection performance. Therefore, we test our model on standard and defense-aware attacks and compare it with state-of-the-art detection methods in this section.

Standard Attacks As shown in Figure 5, our detection method has a very small undetected rate for standard white-box attacks on both the SVHN and CIFAR-10 dataset. EAD and CW attacks are always more easily detected than PGD attacks. For PGD attacks, we achieve an undetected rate below 10% with a small False Positive Rate on the SVHN dataset. The undetected rate for white-box PGD is around 22% with the smallest False Positive Rate on the CIFAR-10 dataset. These demonstrate that our detection mechanism is very effective in detecting standard white-box attacks.

Defense Aware Attacks Following the suggestions in (Carlini & Wagner, 2017a), we test our detection mechanism in the setting where the adversary is fully aware of the defense ("defense-aware attacks") using the CC-PGD attack. Since the PGD attack is stronger than EAD and CW, the first stage of our CC-PGD attack is to construct an adversarial image via standard PGD and then, in the second stage, take the reconstruction error and cycle-consistency into consideration in order to fool the detection methods. In Figure 5 we can clearly see the undetected rate of CC-PGD increases compared to a standard PGD attack. However, there is a significant performance drop in the success rate of White-box CC-PGD (from PGD: 96.0% to CC-PGD: 69.0% on SVHN) as shown in Table 1. This indicates that the adversary needs to sacrifice some success rate in order not to be detected by our detection mechanism.

Comparison with State-of-the-Art Detection Methods We compare our detection methods with the most recent statistical test-based detection method (Roth et al., 2019) and a classifier-based detection method proposed in (Hosseini et al., 2019). In Table 2, we can see that although the statistical test (Roth et al., 2019) and the classifier-based detection method (Hosseini et al., 2019) can

Table 1: Success rate of the white-box and black-box attacks for our deflecting model.

Dataset	EAD		CW		PGD		CC-PGD	
	White	Black	White	Black	White	Black	White	Black
SVHN	100.0%	10.1%	97.6%	1.7%	96.0%	28.7%	69.0%	37.0%
CIFAR-10	100.0%	6.9%	78.0%	1.6%	49.3%	15.5%	46.8%	12.9%

Table 2: Comparison of the Undetected Rate of the state-of-the-art detection methods on the CIFAR-10 dataset. For all the models, the maximum ℓ_∞ perturbation is $\epsilon_\infty = 8/255$ of the pixel dynamic range and the False Positive Rate of the clean input are 5%. The best detection performance are highlighted in **bold**. (Smaller numbers indicate better detection performance.)

Detection Methods	Statistical Test	Classifier-based	Ours
CW	0.1%	**0.0**%	4.6%
Defense-aware PGD	97.8%	98.4%	**28.9**%

detect standard attacks successfully, they both fully fail against defense-aware attacks[2]. In contrast, our proposed reconstruction-based detection mechanism has the best undetected rate in detecting defense-aware adversarial attacks and a very small undetected rate of 4.6% in detecting CW attacks.

5.5 Detection of black-box Attacks

To study the effectiveness of our detection mechanisms, we also test our models on black-box attacks. In Figure 5 we can see an over 50% performance drop in the undetected rate when the inputs are black-box CC-PGD attacks on both datasets. The highest undetected rate of a black-box attack is around 13% on the CIFAR-10 dataset, which demonstrates that our detection mechanism can successfully detect black-box defense-aware attacks. In addition, the great gap of the success rate between white-box and black-box attacks shown in Table 1 indicates our defense model significantly reduces the transferability of all kinds of adversarial attacks.

6 Deflected Attacks

The numbers that we have presented earlier in this paper have implicitly assumed that all adversarial attacks still resemble the initial class, and therefore classifying them as the target class would constitute a mistake. This assumption may not be true in practice. We have discussed the ability of our model to deflect adversarial attacks by having adversarial gradients aligned with the class conditional data distribution, thereby making adversarial attacks resemble the target class. In order to quantify these claims we need to evaluate human performance on the adversarial attacks against our model.

6.1 Human Study on SVHN

In order to validate our claim that our method can deflect adversarial attacks, we performed a human study. We made use of the Amazon Mechanical Turk web service to recruit participants and asked people to label SVHN digits. Each time, they were shown a single image which was randomly sampled from the following five different sets: 1) clean images from the SVHN test set, 2) the undetected and successful black-box PGD and CC-PGD adversarial attacks against our deflecting model, 3) the undetected and successful white-box PGD and CC-PGD adversarial attacks against our deflecting model, 4) the successful black-box PGD attacks generated to attack a standard CNN classifier[3], 5) the successful white-box PGD attacks for the CNN classifier. The maximal adversarial perturbation of all the ℓ_∞ norm-based attacks are bounded by the same $\epsilon_\infty = 16/255$. The recruiters were asked to classify each image as a digit between 0 and 9. If multiple digits occurred in one image, we asked people to label the digit closest to the center of the image. We did not limit the time that people could spend in labeling each image and we did not explain the purpose of this study to the

[2]The numbers of statistical test and classifier-based detection in the Table 2 are extracted from (Hosseini et al., 2019). Since the success rate of the attacks are close to 100%, the undetected rate is roughly (1 - True Positive Rate).

[3]The CNN classifier has the same network architecture as our deflecting model except that we replace the CapsLayer with a convolutional layer.

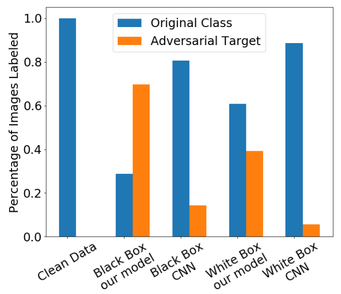

Figure 6: The human study results on SVHN. The maximal ℓ_∞ perturbation is 16/255.

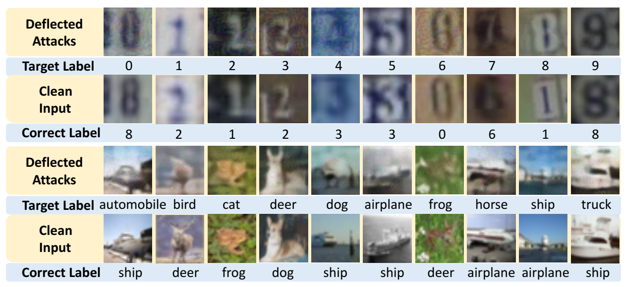
Deflected Attacks
Target Label 0 1 2 3 4 5 6 7 8 9
Clean Input
Correct Label 8 2 1 2 3 3 0 6 1 8
Deflected Attacks
Target Label automobile bird cat deer dog airplane frog horse ship truck
Clean Input
Correct Label ship deer frog dog ship ship deer airplane airplane ship

Figure 7: Deflected adversarial attacks on SVHN and CIFAR-10. The maximal ℓ_∞ perturbation is 16/255 for SVHN and 25/255 for CIFAR-10.

users other than it was a research study. In this way, we had 1500 images labeled in total and each image was labeled by five different users. We then calculated the percentage of uniformly labeled images that were classified as either the original class or the adversarial target class. The results are summarized in Figure 6.

We can see that 69.7% of successful and undetected black-box attacks against our model were classified as the adversarial target. This means that when our defense is attacked with adversarial attacks generated within a standard ℓ_∞ bound, not only are the results visibly different than the source image, they resemble the target class. In this way, these attacks are successfully deflected and can hardly be said to be adversarial, as the network is classifying them the same way our human testers classified them. This is not the case for the baseline CNN model, where only 14.3% of the successful black-box PGD attacks were labeled as the target class. In addition, compared to the white-box attacks, more undetected and successful adversarial attacks generated under the black-box setting are deflected to resemble the target class. This suggests that to attack our deflecting model in a more practical and challenging setting (black-box), the attacker is forced to generate deflected adversarial attacks in order not to be detected. Some examples of deflected adversarial attacks on SVHN are shown in Figure 7.

We note that a certifiably ϵ-robust model (Cisse et al., 2017; Raghunathan et al., 2018; Wong & Kolter, 2017) would by definition incorrectly classify these deflected attacks because they are within ϵ of the original image but are classified but humans as a different class.

6.2 Deflected Attacks on CIFAR-10

To show that our model can effectively deflect adversarial attacks on the CIFAR-10 dataset, we have chosen a deflected adversarial attack for each class with a maximal ℓ_∞ norm as 25/255, displayed in Figure 7. It is apparent that the clean input has been perturbed to have the representative features of the target class, in order to fool both the classifier and our detection mechanisms. As a result, these adversarial attacks are also successfully deflected by our model. Unlike SVHN, for which human evaluators reliably classified the attacks as the target label, the generated adversarial attacks against our deflecting model on the CIFAR-10 do not reliably resemble the target class, though they are much harder to identify than the clean data.

7 Conclusion

In this paper, we introduce a new approach which presents a step towards ending the battle between defenses and attacks by deflecting adversarial attacks. To this end, we propose a new cycle-consistency loss to encourage the winning-capsule reconstruction of the CapsNet to closely match the class-conditional distribution. By making use of the three detection mechanisms, we are able to detect standard adversarial attacks based on three different distance metrics with a low False Positive Rate on the SVHN and CIFAR-10 datasets. To specifically attack our detection mechanisms, we propose a defense-aware attack and find that our model achieves drastically lower undetected rates for defense aware attacks compared to state-of-the-art methods. In addition, a large percentage of the undetected and successful attacks are deflected by our model to resemble the adversarial target class, so they cannot be considered as adversarial any more. This is verified by a human study showing that 70% of the successful and undetected black-box adversarial attacks are classified unanimously by humans as the target class on the SVHN dataset.

References

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. 2018.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14. ACM, 2017a.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57. IEEE, 2017b.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, and Aleksander Madry. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 854–863. JMLR. org, 2017.

Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

Hossein Hosseini, Sreeram Kannan, and Radha Poovendran. Are odds really odd? bypassing statistical detection of adversarial examples. *arXiv preprint arXiv:1907.12138*, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pp. 7167–7177, 2018.

Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.

Yao Qin, Nicholas Frosst, Sara Sabour, Colin Raffel, Garrison Cottrell, and Geoffrey Hinton. Detecting and diagnosing adversarial images with class-conditional capsule reconstructions. *arXiv preprint arXiv:1907.02957*, 2019.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The odds are odd: A statistical test for detecting adversarial examples. *arXiv preprint arXiv:1902.04818*, 2019.

Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pp. 3856–3866, 2017.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Eric Wong and J Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*, 2017.

Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. *arXiv preprint arXiv:1905.11971*, 2019.

A Model Architectures

The details of the network architecture for SVHN and CIFAR-10 dataset are shown in Table 3 and Table 4.

B Training Details

We set the batch size to be 64 and the learning rate to be 0.0001 to train the network on SVHN. For CIFAR-10, the batch size is set to be 128 and the learning rate is 0.0002. The Adam optimizer (Kingma & Ba, 2014) is used to train both networks with the loss function: $\ell = \ell_{margin} + 0.001 \times \ell_{recons} + 0.0005 \times \ell_{cyc}$.

C Attack Parameters

To generate EAD and CW attacks, we follow the previous work (Chen et al., 2018; Carlini & Wagner, 2017b) to set the binary search steps to be 9, maximum iterations to be 1000 and learning rate to be 0.01. To construct ℓ_∞ norm-based attacks, we use 0.01 as the step size in each iteration and in total 200 attack steps. Empirically, the hyperparameter α_1, α_2 and α_3 are set to be 1, 0 and 20 respectively for the best attack performance in our CC-PGD. The parameter that balances the importance of the two stages in CC-PGD is set to be 0.5 on the SVHN and 0.75 on the CIFAR-10 dataset.

D Examples of Adversarial Attacks and Reconstructions

We display successful adversarial attacks but detected by our detection mechanism, and display all the reconstructions when the input are EAD attacks (on the left) and CW attacks (on the right) in Figure 8, PGD attacks (on the left) and our CC-PGD attacks (on the right) in Figure 9 for the SVHN dataset. We also show the successful and detected adversarial EAD attacks (on the left) and CW attacks (on the right) in Figure 10, PGD attacks (on the left) and our CC-PGD attacks (on the right) in Figure 11 for CIFAR-10 dataset.

Table 3: The network architecture for the SVHN dataset.

	Layer Name	Configurations
Classification Network	Conv	filter size: 3x3, number of filters: 64x4, stride size: 1x1, activation: leaky relu
	Conv	filter size: 3x3, number of filters: 64x8, stride size: 1x1, activations: leaky relu
	Avg Pooling	pool size: 2x2, stride size: 2x2
	Conv	filter size: 3x3, number of filters: 64x2, stride size: 1x1, activation: leaky relu
	Conv	filter size: 3x3, number of filters: 64x4, stride size: 1x1, activation: leaky relu
	Avg Pooling	pool size: 2x2, stride size: 2x2
	Conv	filter size: 3x3, number of filters: 64x1, stride size: 1x1, activation: leaky relu
	Conv	filter size: 3x3, number of filters: 64x2, stride size: 1x1, activation: leaky relu
	CapsLayer	number of input capsules: 16, input atoms: 512, number of output capsules: 25, output atoms: 4, number of dynamic routing: 1
Reconstruction Network	fully connected	input size: 100, output size:1024
	fully connected	input size: 1024, output size:16384
	deconv	filter size: 4x4, number of filters: 64, stride size: 2x2
	deconv	filter size: 4x4, number of filters: 32, stride size: 2x2
	conv	filter size: 4x4 number of filters: 3, stride size: 1x1, activation: sigmoid

Table 4: The network architecture for the CIFAR-10 dataset.

	Layer Name	Configurations
Classification Network	Conv	filter size: 3x3, number of filters: 128x4, stride size: 1x1, activation: leaky relu
	Conv	filter size: 3x3, number of filters: 128x8, stride size: 1x1, activations: leaky relu
	Avg Pooling	pool size: 2x2, stride size: 2x2
	Conv	filter size: 3x3, number of filters: 128x2, stride size: 1x1, activation: leaky relu
	Conv	filter size: 3x3, number of filters: 128x4, stride size: 1x1, activation: leaky relu
	Avg Pooling	pool size: 2x2, stride size: 2x2
	Conv	filter size: 3x3, number of filters: 128x1, stride size: 1x1, activation: leaky relu
	Conv	filter size: 3x3, number of filters: 128x2, stride size: 1x1, activation: leaky relu
	CapsLayer	number of input capsules: 16, input atoms: 512, number of output capsules: 25, output atoms: 8, number of dynamic routing: 1
Reconstruction Network	fully connected	input size: 200, output size:1024
	fully connected	input size: 1024, output size:16384
	deconv	filter size: 4x4, number of filters: 64, stride size: 2x2
	deconv	filter size: 4x4, number of filters: 32, stride size: 2x2
	conv	filter size: 4x4 number of filters: 3, stride size: 1x1, activation: sigmoid

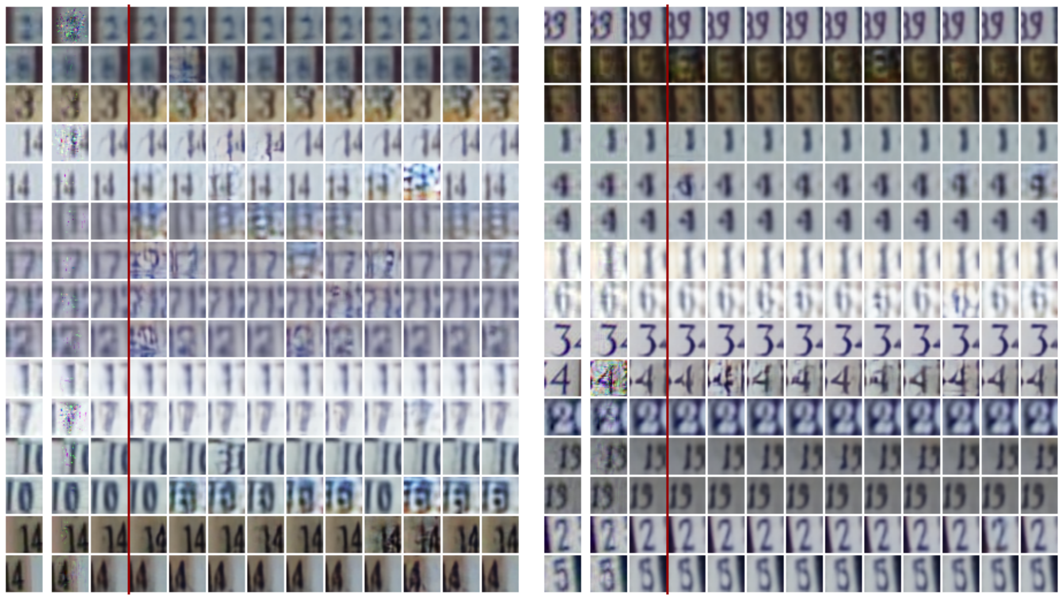

Figure 8: Successful but detected adversarial EAD attacks (on the left) and CW attacks (on the right) and the corresponding capsule reconstructions on SVHN. The first column is the clean input, the second column is the adversarial example, the third column is the winning-capsule reconstruction, the last ten columns are the reconstructions corresponding to class 0 to 9.

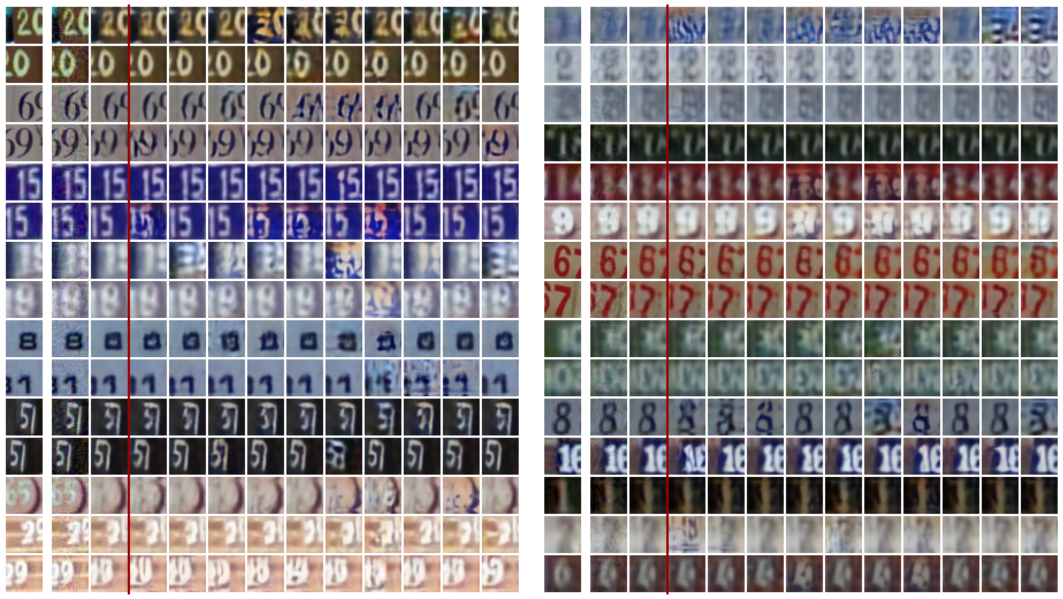

Figure 9: Successful but detected adversarial PGD attacks (on the left) and our CC-PGD attacks (on the right) and the corresponding capsule reconstructions on SVHN. The first column is the clean input, the second column is the adversarial example, the third column is the winning-capsule reconstruction, the last ten columns are the reconstructions corresponding to class 0 to 9. The maximal ℓ_∞ bound to the adversarial perturbation is 16/255.

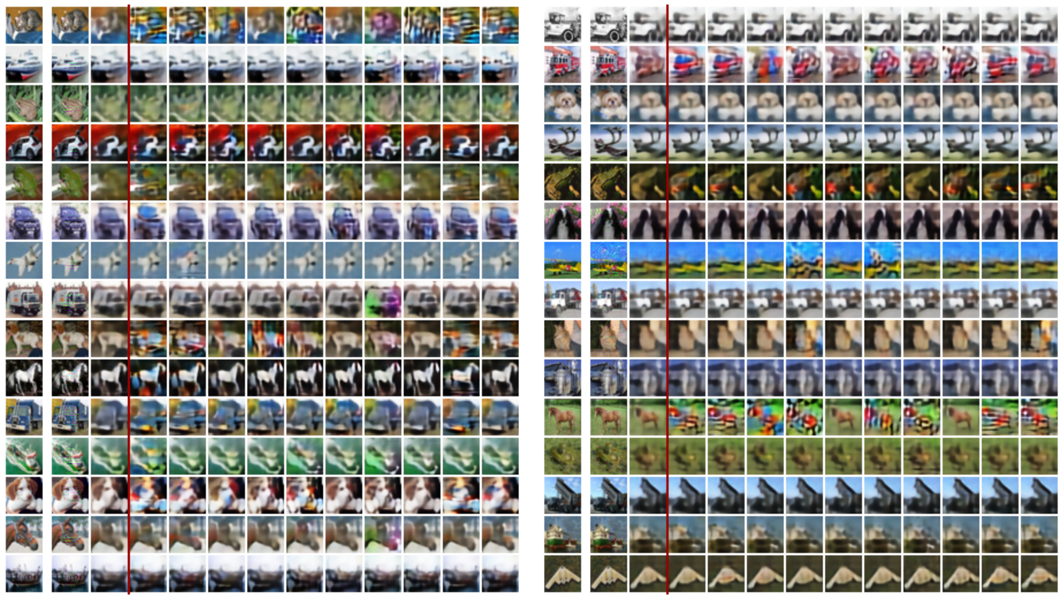

Figure 10: Successful but detected adversarial EAD attacks (on the left) and CW attacks (on the right) and the corresponding capsule reconstructions on CIFAR-10. The first column is the clean input, the second column is the adversarial example, the third column is the winning-capsule reconstruction, the last ten columns are the reconstructions corresponding to class 0 to 9.

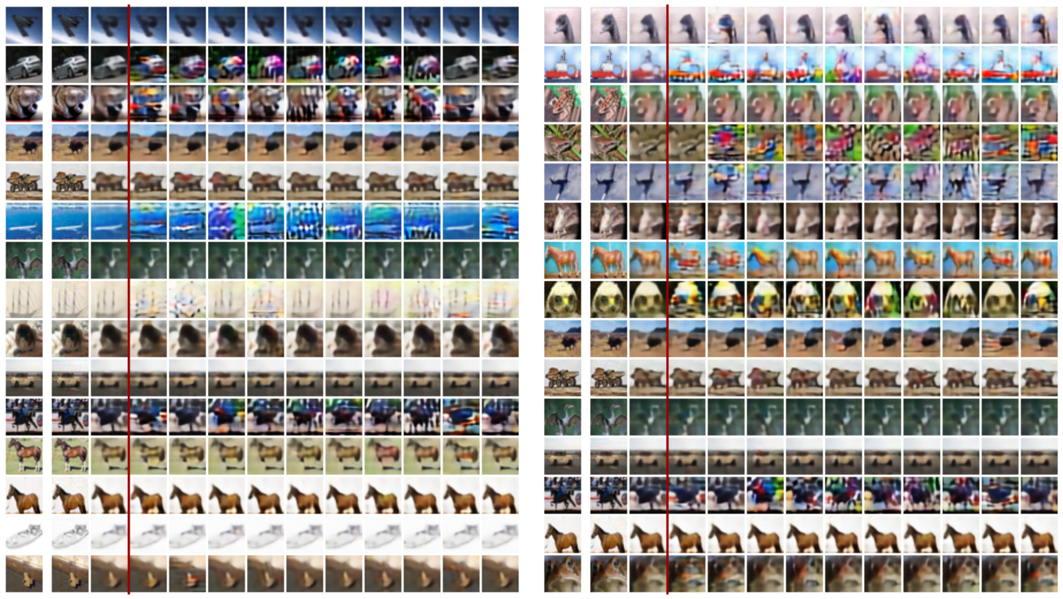

Figure 11: Successful but detected adversarial PGD attacks (on the left) and our CC-PGD attacks (on the right) and the corresponding capsule reconstructions on CIFAR-10. The first column is the clean input, the second column is the adversarial example, the third column is the winning-capsule reconstruction, the last ten columns are the reconstructions corresponding to class 0 to 9. The maximal ℓ_∞ bound to the adversarial perturbation is 8/255.