Defending Against Adversarial Attacks by Leveraging an Entire GAN

Gokula Krishnan Santhanam [1] **Paulina Grnarova** [1]

Abstract

Recent work has shown that state-of-the-art models are highly vulnerable to adversarial perturbations of the input. We propose *cowboy*, an approach to detecting and defending against adversarial attacks by using both the discriminator and generator of a GAN trained on the same dataset. We show that the discriminator consistently scores the adversarial samples lower than the real samples across multiple attacks and datasets. We provide empirical evidence that adversarial samples lie outside of the data manifold learned by the GAN. Based on this, we propose a cleaning method which uses both the discriminator and generator of the GAN to project the samples back onto the data manifold. This cleaning procedure is independent of the classifier and type of attack and thus can be deployed in existing systems.

1. Introduction

Recent work on adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014b; Kurakin et al., 2016) has shown that neural networks are brittle and can easily be confused by small imperceptible modifications of the input. While these modifications do not affect human perception, they cause misclassification of samples in deep-learning models creating a threat for security and safety.

Current defenses against adversarial samples follow two lines of work: (1) Modifying the training procedure of the classifier to make it more robust by either modifying the training data (adversarial learning (Szegedy et al., 2013; Goodfellow et al., 2014b)) or blocking gradient pathways (e.g. defensive distillation (Papernot et al., 2016)) and (2) removing the adversarial perturbations from the input (Meng & Chen, 2017; Pouya Samangouei, 2018; Song et al., 2017).

We present *cowboy*, an approach to detecting and defending against adversarial samples that is agnostic to the target classifier and its training procedure and is independent of the attack method. We leverage the power of a Generative Adversarial Network (GAN) (Goodfellow et al., 2014a) trained only on the real data, and use both the discriminator and generator to successfully defend against various attacks.

A crucial step towards tackling adversarial samples is the ability to distinguish them from real samples. We postulate that adversarial samples lie outside of the data manifold and that the discriminator can effectively detect this. We test this hypothesis by conducting experiments using different attacks (Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014b), Basic Iterative Method (BIM) (Kurakin et al., 2016), Virtual Adversary Method (VAM) (Miyato et al., 2015), Projected Gradient Descent Method (PGDM) (Madry et al., 2017) and Momentum Iterative Method (MIM) (Dong et al., 2017)) on MNIST, Fashion-MNIST, CIFAR-10, and SVHN datasets and show that the discriminator consistently scores the adversarial samples lower than the original samples.

Once detected, the adversarial samples can be cleaned by projecting them back to the data manifold learned by the GAN and pushing them to a high scoring region guided by the score of the discriminator. When the cleaning procedure is used as a pre-processing step on the samples before given to the classifier, the accuracy of classification improves from 0.02% to 0.81% for one of the worst attack.

Our main contributions can be summarized as:

- We empirically show that adversarial samples lie outside of the data manifold learned by a GAN that has been trained on the same dataset.
- We propose a simple procedure that uses both the generator and the discriminator to successfully detect and clean adversarial samples.

The key feature of our framework is that both the GAN and classifier are not modified nor are shown adversarial samples during training. This allows our approach to generalize to multiple attacks and datasets with only a few lines of code. This makes our framework an easy to plug-in step in already deployed classification pipelines

The rest of the paper is organized as follows. Necessary background including various defense mechanisms and at-

[1]Department of Computer Science, ETH Zurich, Zurich, Switzerland. Correspondence to: Gokula Krishnan Santhanam <sgokula@ethz.ch>.

tack models that we use are introduced in Section 2. In Section 3, we describe and motivate *cowboy*, a simple, yet effective defense algorithm. Finally, Section 4 presents and discusses experimental results for both detection and cleaning under different attack methods for generating adversarial samples across 4 different datasets.

2. Background and Related Work

Attack Strategies. Adversarial samples were first reported by (Szegedy et al., 2013). Since then many attack strategies have been proposed, which can broadly be classified into black-box and white-box attacks. White-box strategies have access to all the weights and gradients of the classifier. Black-box strategies on the other hand, have access only to the predictions of the network. Even though this might make the attacks more difficult, it enables successful attacks to be transferable to many classifiers.

In this work, an attack strategy is an algorithm that perturbs an original sample $X \in \mathbf{R}^n$ to an adversarial sample $X_{adv} = X + \delta \in \mathbf{R}^n$ in a way that the change is undetectable to the human eye. The change is measured by the l_∞ norm of the perturbation, denoted by ϵ. We focus on the following attack methods:

Fast Gradient Sign Method (Goodfellow et al., 2014b) The Fast Gradient Sign Method (FGSM) generates adversarial images using

$$X^{adv} = X + \epsilon \cdot sign(\bigtriangledown_X J(X, y_{true})),$$

where ϵ controls the perturbation's amplitude, with smaller values creating imperceptible perturbations. This attack moves all the pixels of the input X in the direction of the gradient simultaneously. This method is easy to implement and inexpensive to compute but is easily detected by current methods.

Basic Iterative Method (Kurakin et al., 2016) Basic Iterative Method (BIM) is a variant of the FGSM method. It applies FGSM multiple times with a smaller step size. The adversarial examples are computed as

$$X_0^{adv} = X, X_{n+1}^{adv} = X_n^{adv} + \alpha \cdot sign(\bigtriangledown_X J(X_{n+1}^{adv}, y_{true})).$$

In practice, α is smaller when compared to ϵ of FGSM. We also clip the updates such that the adversarial samples lie within the ϵ-ball of X.

Momentum Iterative Method (Dong et al., 2017) The Momentum Iterative Method (MIM) is a variant of BIM that exploits momentum (Qian, 1999) when updating X_n^{adv}. This results in adversarial samples of superior quality.[1]

[1]This method won the first places in NIPS 2017 Non-targeted Adversarial Attack and Targeted Adversarial Attack competitions.

Projected Gradient Descent Method (Madry et al., 2017) Projected Gradient Descent Method(PGDM) is an optimization based attack which finds a point, X^{adv}, that lies within the ϵ-ball of X such that it minimizes the probability of the true label y_{true}.

Virtual Adversary Method (Miyato et al., 2015) Virtual Adversary Method (VAM) uses Local Distributional Smoothness (LDS), defined as the negative sensitivity of the model distribution $p(y|x, \theta)$ with respect to the perturbation of X, measured in terms of KL divergence. It exploits LDS to find the direction in which the model is most sensitive to perturbations to generate adversarial samples.

Defense Strategies. Common approaches to defending against adversarial attacks include training with adversarial samples (Goodfellow et al., 2014b), modifying the classifier to make it more robust to attacks (Papernot et al., 2016), label-smoothing (Warde-Farley & Goodfellow, 2016), as well as using auxiliary networks to pre-process and clean the samples (Gu & Rigazio, 2014; Meng & Chen, 2017). In the context of purifying the samples, a new line of work focuses on using generative models to detect and defend against adversarial attacks. Models like RBMs and PixelRNNs (van den Oord et al., 2016) have been used to varying degrees of success as well.

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014a) are an interesting prospect in defending against adversarial attacks. Even though training GANs is still notoriously difficult, there have been successful efforts in this direction. Defense-GAN (Pouya Samangouei, 2018) uses the generator in order to sanitize inputs before passing them to the classifier. APE-GAN (Shen et al., 2017) modifies a GAN such that the generator learns to clean the adversarial samples, whereas the discriminator tries to discriminate between real and adversarial input. While these approaches focus only on the generative part of the network, we use both the generator and the discriminator as means to detecting and protecting against attacks. To our best knowledge, there has not been any prior work on using an unmodified discriminator for this task.

3. Cowboy

Our goal is twofold: we want to *(i)* be able to identify whether an input is adversarial or not, and *(ii)* clean it from its adversarial noise if it is. In order to do so, we aim at combining information drawn from two sources, respectively the generator G and the discriminator D of a GAN trained on the same dataset as our initial classifier.

3.1. Classifiers learn discriminative surfaces

As noted in (Goodfellow et al., 2014b), common activation functions used in Deep Learning like Sigmoid, ReLU and Tanh are either piece-wise linear or used in regions where they have linear-like behavior. As a result, classifiers modeled as DNNs tend to learn hypersurfaces that split the output space into finitely many disjoint open subspaces. We can see this clearly in the case of a mixture of two Gaussians respectively centered at $(3, 3)$ and $(-3, -3)$ and with variance 1, as shown in Figure 1. It is also interesting to note that the classifier assigns high probability to regions where it has never seen samples before, showing that the classifier has no notion of the data manifold. The classifier only learns a discriminative surface to maximize classification accuracy.

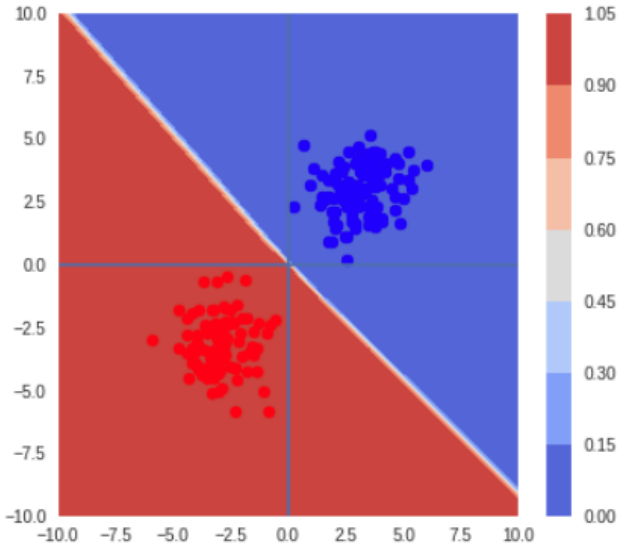


Figure 1. Linear discriminative surface learned by a classifier on a Mixture of two Gaussians.

3.2. Adversarial Examples lie outside the Data Manifold

There has been work (Gong et al., 2017; Song et al., 2017; Grosse et al., 2017) arguing that adversarial perturbations result in samples lying outside of the data manifold. Since the classifier has no knowledge of the manifold, this can result in misclassification with very high confidence of the classifier. We can consider this to be akin to testing out-of-distribution samples, which naturally results in poor test-time performance. This suggests the use of defense mechanisms involving the support of the real distribution p_{data}.

3.3. D can detect adversarial attacks

What kind of information of p_{data} is captured by the discriminator D? Consider the following two claims:

1. There exists $0 < \delta \ll 1$ such that the support of p_{data} is contained in $D^{-1}([1-\delta, 1])$.

2. If x in a real sample, then corrupting it into an adversarial sample $\tilde{x}$ using common adversarial attack methods would yield $D(\tilde{x}) \ll 1$.

The first of these claims is equivalent to saying that testing if a sample x is real by asking whether $1 - \delta \leq D(x) \leq 1$ would not yield any false negative, which can also be formulated as an absence of mode-collapse for the discriminator D.

This behaviour can be understood by reasoning about the dynamics of learning in the GAN's min-max game:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]. \quad (1)$$

The discriminator has to assign higher scores to regions around the data points as these are shown as positive examples and doing otherwise would increase its loss.

We also assess the validity of this assumption empirically by drawing violin plots of the values of D for a variety of datasets in Figure 3, which appears to hold true for each dataset. This allows us to consider claim 1 as being valid, under the assumption that the discriminator did not undergo mode collapse.

However, what can be said about false positives for this test?, *i.e.*, is there a consequent part of the space outside of the data manifold where D scores high? As argued in Section 3.1, a classifier tends to only learn discriminative surfaces in order to maximize classification accuracy, and hence D might assign high scores to the presence of real data in regions where it has never seen any sample, either real or from G. Indeed, this is what seems to happen in Figure 2.

Therefore, in order to assess whether D scoring low would give a good proxy for the detection of adversarial samples, it is both *necessary* and *sufficient* to assess the validity of claim 2, i.e. that an adversarial sample is unlikely to live in the region of the space outside of the data manifold where D scores high. Indeed, adversarial examples lie outside the data manifold (see Section 3.2), and the data manifold is contained in the region where D scores high.

Similarly as we did for claim 1, we assess the validity of claim 2 empirically by drawing violin plots of the values of

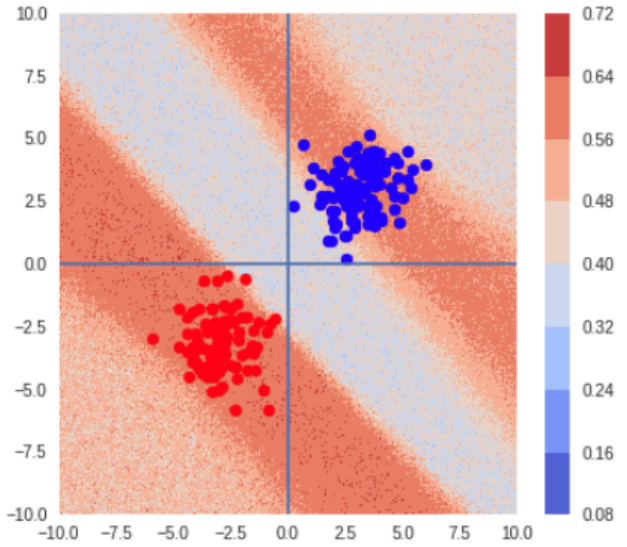


Figure 2. Score of the discriminator D of a GAN trained on a mixture of two Gaussians.

D on a variety of adversarial attacks. Figure 3 shows that adversarial samples get assigned a low score by D

This demonstrates that D scoring low provides a good proxy for the detection of adversarial attacks.

3.4. Estimating a good threshold

The concepts discussed in the previous section are agnostic w.r.t. which attack is used. This means that if two completely different attacks generate adversarial samples in the same region, then scores of D for these samples would be similar. This implies that we can obtain a good estimate for the threshold by using any attack method on the trained classifier. To do this, we can generate adversarial samples[2] using an arbitrary attack method (say FGSM) for each (or a subset) data point in the dataset. A simple threshold could then be the average discriminator score across all adversarial samples. Note that although choosing the maximum instead of the average could be very badly affected by an outlier, one can also simply choose an L^p average $((1/p)\sum(\cdot)^p)^{1/p}$ for some $p \in [1, \infty)$ as a way to interpolate between the average and the maximum.

3.5. Taming Adversarial Samples

Once an input has been detected as adversarial, we can attempt to clean this input from its adversarial noise. In order to do so, Pouya Samangouei (2018) and similar work suggest to find an optimal L^2-reconstruction of x by G:

$$z^* = \arg\min_z \parallel G(z) - x \parallel_2^2 . \tag{2}$$

However, these approaches use only the generator G of the GAN.

As already argued in the previous sections, the discriminator D contains important information about the data manifold. How can we exploit this information in order to improve reconstruction? The probability that a reconstructed input $G(z)$ lies on the data manifold can be modelled by D as $D(G(z))$, whose log-likelihood is then $\log(D(G(z)))$. Moreover, it is natural to interpret a rescaled L^2-reconstruction term $\frac{1}{2\sigma^2}\|G(z) - x\|_2^2$ as the negative log-likelihood of an isotropic Gaussian of variance σ^2 centered at x, since the logarithm and the exponential cancel. A desired reconstruction term would aim at maximizing the likelihood that $G(z)$ be close to x, while lying on the data manifold. If we model these two events as being independent and having probability densities respectively $\mathcal{N}(x, \sigma^2 I_n)$ and $D(G(z)) -$ where σ is a hyper-parameter of the model $-$, then this leads us to maximizing the following log-likelihood:

$$\log\left(\frac{1}{(2\pi\sigma^2)^{n/2}} e^{-\frac{1}{2\sigma^2}\|G(z)-x\|_2^2}\right) + \log(D(G(z))), \tag{3}$$

which is strictly equivalent to minimizing the following quantity:

$$\mathcal{L}(z) := \frac{1}{2\sigma^2}\|G(z) - x\|_2^2 - \log(D(G(z))). \tag{4}$$

Note that σ trades-off between exact reconstruction ($G(z) = x$) and realness of the reconstruction ($D(G(z)) = 1$).

We can now find a minimizer z^* of $\mathcal{L}$ by gradient-descent, as in Pouya Samangouei (2018), while keeping the generator and the discriminator fixed. We experiment with different values of σ and discuss its effect in 4.2. The two terms composing our reconstruction loss $\mathcal{L}$ ensure that the cleaned image is similar to the input and lies on the data manifold, which is achieved without evaluating any likelihood over p_{data} explicitly.

Algorithm 1 Cleaning Adversarial Inputs

Input: adversarial input x, learning rate η
Sample z_0 from noise prior $p_z(z)$
for $i = 1$ **to** m **do**
  Update $z_i \leftarrow z_{i-1} - \eta \nabla_z \mathcal{L}(z_{i-1})$
end for
return $G(z_m)$, the cleaned image

[2] Let's define adversarial samples as those that cause at least a 20% (arbitrarily chosen) drop in accuracy

Table 1. Classifier Accuracies (5,000 cleaning steps): The classifier accuracy on the adversarial samples increases significantly when the samples have been preprocessed with the Cowboy cleaning method.

ATTACK	MNIST			F-MNIST			CIFAR-10			SVHN		
	Original	Adv	Clean	Original	Adv	Clean	Original	Adv	Clean	Original	Adv	Clean
FGSM (l_∞)	0.97	0.12	0.78	0.89	0.11	0.44	0.78	0.30	0.53	0.87	0.34	0.72
PGDM (l_∞)	0.97	0.02	0.81	0.89	0.07	0.48	0.78	0.06	0.53	0.87	0.13	0.60
BIM (l_∞)	0.97	0.02	0.15	0.89	0.07	0.13	0.78	0.20	0.47	0.87	0.16	0.49
MIM (l_∞)	0.97	0.03	0.45	0.89	0.08	0.32	0.78	0.21	0.43	0.87	0.15	0.48
VAM	0.97	0.32	0.43	0.89	0.50	0.34	0.78	0.45	0.56	0.87	0.67	0.76

4. Experiments

Experimental Setup. We conduct experiments on MNIST, Fashion-MNIST (Xiao et al., 2017), CIFAR-10 and SVHN. We use the Fast Gradient Sign Method (FGSM), Basic Iterative Method (BIM), Virtual Adversary Method (VAM), Projected Gradient Descent Method (PGDM) and Momentum Iterative Method (MIM) to mount attacks on the classifier.

The classifier is a simple CNN and the GAN is based on the DCGAN family of architectures (Radford et al., 2015). Batch Normalization (Ioffe & Szegedy, 2015) and Dropout (Srivastava et al., 2014) are used only for the classifier. The weights are initialized using the Xavier Initializer (Glorot & Bengio, 2010). Other than normalizing the inputs to lie between -1 and 1, no special pre- or post-processing steps are performed. The models are implemented in TensorFlow (Abadi et al., 2016) and the implementations of the adversarial attacks are based on CleverHans (Papernot et al., 2017)[3]. Both the classifier and the GAN for SVHN and CIFAR-10 are trained for 40K steps, whereas the ones for MNIST and Fashion-MNIST are trained for 2K steps. Once training is complete, we test on the entire respective test sets. Further details on the hyperparameters along with the specific architectures can be found in the codebase and supplementary material.

Results on Detection of Adversarial Samples. We now directly test the hypothesis that the scores the discriminator assigns to samples are meaningful and can be utilized for the detection of adversarially perturbed images. Figure 3 shows the discriminator distribution of these scores through violin plots for the different datasets and attack methods, respectively. Note that adversarial samples that do not cause miss-classification are included as well. The trend of consistently assigning higher scores to unmodified samples and lower scores to adversarial inputs is seen across all settings. Furthermore, we find that the score assigned by the discriminator to adversarial samples correlates to the severeness of the attack, i.e. attacks that cause larger decrease in the classification accuracy tend to be scored lower by the discriminator.[4]

For CIFAR-10 it is noticeable that some of the real images are assigned a low score by the discriminator. A randomly sampled batch of those is shown in Figure 4. As can be seen, the images tend to contain patches and unnatural artifacts.

Despite the fact that the GAN has only been trained on the real samples, and adversarial samples generated with a specific technique are never shown to the discriminator, it is capable of detecting adversarial perturbations irrespective of the attack method and dataset. This suggests that the discriminator behaves favorably in terms of ability to detect and distinguish adversarial samples.

Results on Cleaning Adversarial Samples. Once detected, the adversarial samples can in theory be cleaned by removing the perturbation such that the purified sample is assigned to the correct class when passed to the target classifier. Table 1 gives the classification accuracy on: (i) the entire test set, (ii) the adversarially perturbed samples and (iii) the cleaned samples. When used as a preprocessing procedure, the cleaning method significantly improves the classifier accuracy on the adversarial samples. The method generalizes well as reflected by the improved accuracy across the various dataset and attack method combinations.

Figure 5 showcases adversarially modified samples from CIFAR-10 and how they look like after our cleaning procedure has been applied.

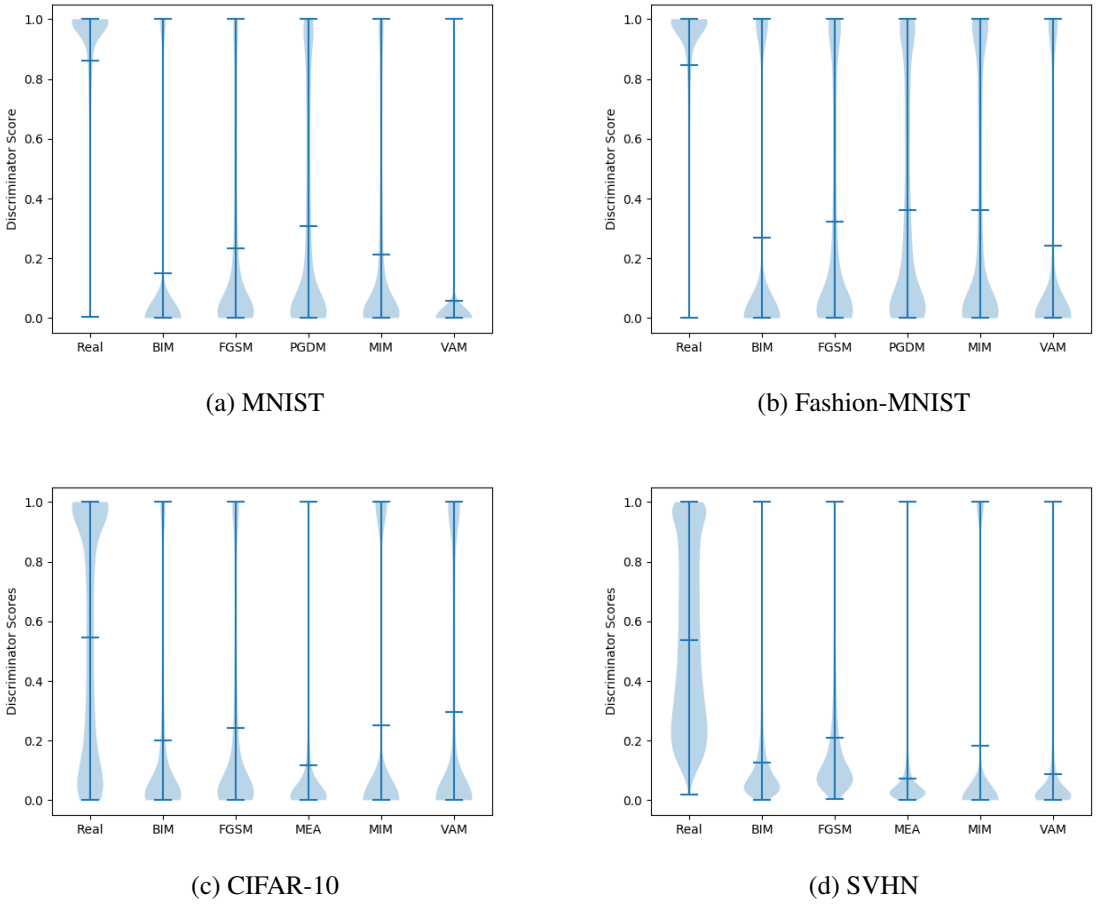
In the following we investigate how specific aspects of the algorithm affect the results.

4.1. Effect of the quality of the trained GAN

Since we use different parts of a GAN to detect and clean adversarial samples, it is only natural that the quality of the GAN itself may influence the performance of these methods.

[3] codebase will be released after the review process

[4] The severeness of the attack can be seen in Table 1 by comparing the accuracy of the classifier on the clean samples versus the accuracy of the classifier after applying the adversarial perturbations.



(a) MNIST

(b) Fashion-MNIST

(c) CIFAR-10

(d) SVHN

Figure 3. Distribution of the discriminator scores for different datasets. The violin denoted by real shows the discriminator scores on the validation set. The others show the scores of the discriminator after adding the perturbation to the images. As a result of the added perturbation, the images are pushed out of the data manifold and assigned a lower score. Note that not all perturbations lead to a successful adversarial sample, where successfulness is measured by the ability to cause miss-classification when using the target classifier.

Figure 6 shows the classification accuracy where the discriminator and generator used for the cleaning procedure are trained for increasing number of training steps. As expected, we observe that as the GAN approaches convergence, the general trend of the performance quality increases. It can also be observed that the trends are not smooth, which can be attributed to the unstable nature of the GAN training.

On the other hand, the performance of the detection mechanism of *cowboy* is not influenced as much by the quality of the GAN, as shown in Figure 7. Even when the GAN is not fully trained, it assigns lower scores to adversarial in comparison to real samples, and hence, is able to detect the former.

4.2. The Effect of The Two Terms

In Section 3, we argue that both the reconstruction term and the discriminator score of the cleaning objective are important for better purification of the adversarial samples. Table 2 gives the comparison to Defense-GAN, a cleaning algorithm that is based only on the generator of the GAN.

The results empirically justify the usefulness of the discriminator score in terms of pushing the adversarial samples on the data manifold.

4.3. Summary

For all the experiments, we use the standard GAN training procedure, without hyperparameter tuning and additional tricks for stabilizing the training (e.g. adding noise or reg-

Table 2. Defense-GAN vs. Cowboy (5000 cleaning steps): Classifier accuracy on adversarial samples cleaned with each of the two methods. Cowboy consistently gives better results by the additional utilization of the discriminator score.

ATTACK	MNIST		F-MNIST		CIFAR-10		SVHN	
	Defense-GAN	Cowboy	Defense-GAN	Cowboy	Defense-GAN	Cowboy	Defense-GAN	Cowboy
FGSM (l_∞)	0.74	0.78	0.43	0.44	0.42	0.53	0.67	0.72
PGDM (l_∞)	0.74	0.81	0.48	0.48	0.47	0.53	0.58	0.60
BIM (l_∞)	0.12	0.15	0.11	0.13	0.38	0.47	0.49	0.49
MIM (l_∞)	0.32	0.45	0.27	0.32	0.35	0.43	0.44	0.48
VAM	0.34	0.43	0.32	0.34	0.45	0.56	0.70	0.76

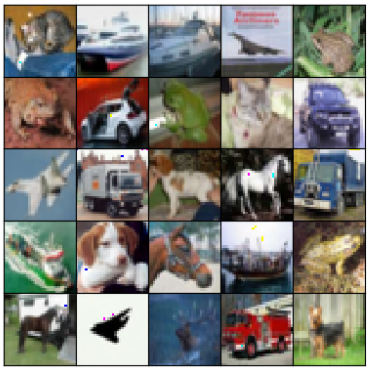


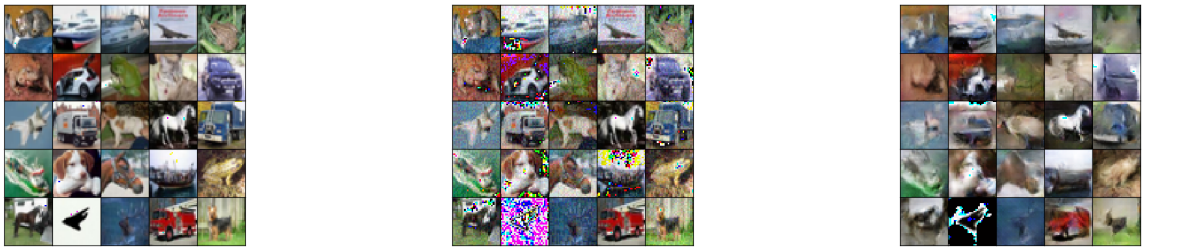
Figure 4. Randomly sampled images from the validation set of Cifar-10 for which the discriminator assigns low scores.

ularizing (Roth et al., 2017), or using a more sophisticated GAN training and architecture, such as BEGAN (Berthelot et al., 2017)). Even so, after training the GAN on the real samples, both the generator and discriminator contain useful information that can be leveraged for the defense against adversarial attacks.

The discriminator can be effectively used as a tool for detection of adversarial samples, whereas the combination of both the generator and discriminator constitutes a simple, yet powerful strategy for cleaning adversarial samples before their classification. As the method only works as a preprocessing step, it requires no modification of the training of the target classifiers, hence making it easily incorporable to already existing models.

5. Conclusion

In this paper, we presented *cowboy*, a novel GAN-based method for successful detection and purification of adversarial samples. The method is based on the hypothesis that adversarial samples lie outside of the data manifold that is learned by a Generative Adversarial Network, irrespective of the underlying attack mechanism. We provide empirical evidence for this hypothesis and show that the discriminator acts as a good tool for the detection of adversarially perturbed samples. The defense strategy is based on projecting the detected adversarial samples back to the data manifold in order to clean the adversarial noise. Various experiments show the effectiveness and the generalization of the method across different attacks and datasets.



(a) Real samples (b) Adversarially perturbed samples (c) Cleaned adversarial samples

Figure 5. Here we show a few images from CIFAR-10 (a), then a few adversarial attacks performed on these images (b), and finally what they look like after cleaning by our proposed method (c).

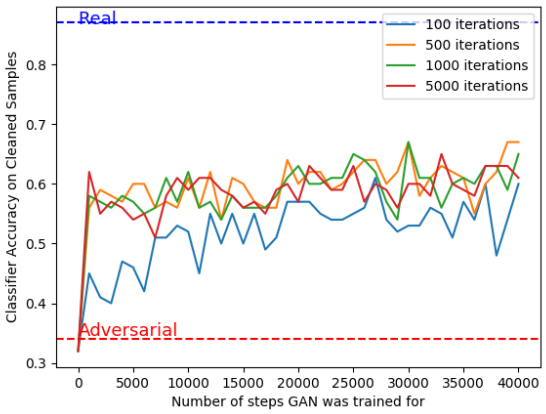


Figure 6. Effect of GAN quality on adversarial sample cleaning (FGSM (l_∞) on SVHN): The classifier accuracy improves as the training of the GAN progresses. Different colors represent results for different number of cleaning steps.

Figure 7. Effect of GAN quality on adversarial sample detection (FGSM (l_∞) on SVHN): The average discriminator score for all the real and adversarial samples is given with the orange and blue curve, respectively. The discriminator is able to distinguish adversarial from real samples even early on throughout the training.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *ArXiv e-prints*, March 2016.

Berthelot, David, Schumm, Tom, and Metz, Luke. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

Dong, Y., Liao, F., Pang, T., Su, H., Hu, X., Li, J., and Zhu, J. Boosting Adversarial Attacks with Momentum. *ArXiv e-prints*, October 2017.

Glorot, Xavier and Bengio, Yoshua. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

Gong, Z., Wang, W., and Ku, W.-S. Adversarial and Clean Data Are Not Twins. *ArXiv e-prints*, April 2017.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Networks. *ArXiv e-prints*, June 2014a.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and Harnessing Adversarial Examples. *ArXiv e-prints*, December 2014b.

Grosse, K., Manoharan, P., Papernot, N., Backes, M., and McDaniel, P. On the (Statistical) Detection of Adversarial Examples. *ArXiv e-prints*, February 2017.

Gu, S. and Rigazio, L. Towards Deep Neural Network Architectures Robust to Adversarial Examples. *ArXiv e-prints*, December 2014.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456, 2015.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *ArXiv e-prints*, July 2016.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. *ArXiv e-prints*, June 2017.

Meng, Dongyu and Chen, Hao. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147. ACM, 2017.

Miyato, T., Maeda, S.-i., Koyama, M., Nakae, K., and Ishii, S. Distributional Smoothing with Virtual Adversarial Training. *ArXiv e-prints*, July 2015.

Papernot, Nicolas, McDaniel, Patrick, Wu, Xi, Jha, Somesh, and Swami, Ananthram. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pp. 582–597. IEEE, 2016.

Papernot, Nicolas, Carlini, Nicholas, Goodfellow, Ian, Feinman, Reuben, Faghri, Fartash, Matyasko, Alexander, Hambardzumyan, Karen, Juang, Yi-Lin, Kurakin, Alexey, Sheatsley, Ryan, Garg, Abhibhav, and Lin, Yen-Chen. cleverhans v2.0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 2017.

Pouya Samangouei, Maya Kabkab, Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BkJ3ibb0-.

Qian, Ning. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Roth, Kevin, Lucchi, Aurelien, Nowozin, Sebastian, and Hofmann, Thomas. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems*, pp. 2015–2025, 2017.

Shen, S., Jin, G., Gao, K., and Zhang, Y. APE-GAN: Adversarial Perturbation Elimination with GAN. *ArXiv e-prints*, July 2017.

Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples. *ArXiv e-prints*, October 2017.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *ArXiv e-prints*, December 2013.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel Recurrent Neural Networks. *ArXiv e-prints*, January 2016.

Warde-Farley, David and Goodfellow, Ian. 11 adversarial perturbations of deep neural networks. *Perturbations, Optimization, and Statistics*, pp. 311, 2016.

Xiao, Han, Rasul, Kashif, and Vollgraf, Roland. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747.