Supplementary Material

Yanzhi Chen [1] **Renjie Xie** [2] **Zhanxing Zhu** [3]

A. Proof of Theorem 1

Theorem 1. *Let T' be the number of backtracking steps. Assuming that (a) the solved $\mathbf{z}^*$ is optimal in each $\mathbf{z}$-step (b) the backtracking learning rate η' satisfies $\eta'\|\mathbf{H}\| \leq 1$ where $\mathbf{H} = \nabla^2_{\mathbf{z}^*\mathbf{z}^*} l_G$ is the Hessian of l_G at $(\mathbf{z}^*, \mathbf{x}')$ and (c) $\mathbf{z}^0_{new}$ is not far away from $\mathbf{z}^*$ and l_G is smooth* w.r.t $\mathbf{z}$. *Then $\nabla_{\mathbf{x}'} Q(\mathbf{x}')$ is approximately a T'-order approximation to $\nabla_{\mathbf{x}'}\mathbf{z}^*$:*

$$\nabla_{\mathbf{x}'}\mathbf{z}^* = \sum_{t=0}^{\infty} \mathbf{B}\mathbf{A}^t, \qquad \nabla_{\mathbf{x}'} Q(\mathbf{x}') \approx \sum_{t=0}^{T'} \mathbf{B}\mathbf{A}^t$$

where $\mathbf{A}$ and $\mathbf{B}$ is some matrix with $\|\mathbf{A}\| \leq 1$.

Proof: We first prove $\nabla_{\mathbf{x}'}\mathbf{z}^* = \sum_{t=0}^{\infty} \mathbf{B}\mathbf{A}^t$, then $\nabla_{\mathbf{x}'} Q(\mathbf{x}') \approx \sum_{t=0}^{T'} \mathbf{B}\mathbf{A}^t$. This proof is inspired by (Shaban et al., 2019).

Lemma 1. *Given conditions (a)(b) in Theorem 1, we have*

$$\nabla_{\mathbf{x}'}\mathbf{z}^* = \sum_{t=0}^{\infty} \mathbf{B}\mathbf{A}^t$$

where

$$\mathbf{A} = \mathbf{I} - \eta'\nabla^2_{\mathbf{z}^*,\mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*), \qquad \mathbf{B} = -\eta'\nabla_{\mathbf{x}',\mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*)$$

Proof: it can be easily shown by the technique of implicit differentiation that (Gould et al., 2016):

$$\nabla_{\mathbf{x}'}\mathbf{z}^* = -\nabla_{\mathbf{x}',\mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*) \underbrace{\nabla^2_{\mathbf{z}^*,\mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*)}_{\mathbf{H}^{-1}}{}^{-1} \tag{1}$$

Remark that for a matrix $\mathbf{M}$ that $\|\mathbf{M}\| \leq 1$, we have (Horn & Johnson, 2012):

$$(\mathbf{I} - \mathbf{M})^{-1} = \sum_{t=0}^{\infty} \mathbf{M}^t \tag{2}$$

then, since $\eta'\|\mathbf{H}\| \leq 1$,

$$\mathbf{H}^{-1} = \eta' \cdot (\mathbf{I} - (\mathbf{I} - \eta'\mathbf{H}))^{-1} = \eta' \sum_{t=0}^{\infty} (\mathbf{I} - \eta'\mathbf{H})^t \tag{3}$$

substituting this back to (1):

$$\nabla_{\mathbf{x}'}\mathbf{z}^* = \underbrace{-\eta'\nabla_{\mathbf{x}',\mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*)}_{\mathbf{B}} \sum_{t=0}^{\infty} \underbrace{(\mathbf{I} - \eta'\mathbf{H})}_{\mathbf{A}}{}^t \tag{4}$$

which completes the proof. □

*Equal contribution [1]School of Informatics, The University of Edinburgh, UK [2]School of Information Engineering, Southeast University, China [3]School of Mathematical Sciences, Peking University, China. Correspondence to: Zhanxing Zhu <zhanxing.zhu@pku.edu.cn>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

Lemma 2. *Given conditions (a)(c) in Theorem 1, we have*

$$\nabla_{\mathbf{x}'} Q(\mathbf{x}') \approx \sum_{t=0}^{T'} \mathbf{B}\mathbf{A}^t$$

where $\mathbf{A}$ *and* $\mathbf{B}$ *are the same matrices as in Lemma 1.*

Proof: remark that

$$\begin{aligned} Q(\mathbf{x}') := Q_{\mathbf{x}'}(\mathbf{z}^0) &= \underbrace{q_{\mathbf{x}'} \circ q_{\mathbf{x}'} \ \ldots \ \circ q_{\mathbf{x}'}(\mathbf{z}_0)}_{T'} \\ q_{\mathbf{x}'}(\mathbf{z}) &= \mathbf{z} - \eta' \cdot \nabla_{\mathbf{z}} l_G(\mathbf{z}; \mathbf{x}') \end{aligned} \tag{5}$$

For clarity let us write $q_{\mathbf{x}'}(\mathbf{z}) = q(\mathbf{z}, \mathbf{x}')$ from now on. The total derivative *w.r.t* $\mathbf{x}'$ in $q(\mathbf{z}, \mathbf{x}')$ is then

$$\nabla_{\mathbf{x}'} q(\mathbf{z}, \mathbf{x}') = \nabla_{\mathbf{z}} q(\mathbf{z}, \mathbf{x}')\Big|_{\mathbf{x}'} \nabla_{\mathbf{x}'} \mathbf{z} + \nabla_{\mathbf{x}'} q(\mathbf{z}, \mathbf{x}')\Big|_{\mathbf{z}} \tag{6}$$

Remark that $\mathbf{z}^{t+1} = q_{\mathbf{x}'}(\mathbf{z}^t) = q(\mathbf{z}^t, \mathbf{x}')$. Then

$$\nabla_{\mathbf{x}'} \mathbf{z}^{t+1} = \underbrace{\nabla_{\mathbf{z}^t} \mathbf{z}^{t+1}\Big|_{\mathbf{x}'}}_{\alpha_t} \nabla_{\mathbf{x}'} \mathbf{z}^t + \underbrace{\nabla_{\mathbf{x}'} \mathbf{z}^{t+1}\Big|_{\mathbf{z}^t}}_{\beta_t} \tag{7}$$

Unfolding this recursive expression, we obtain

$$\nabla_{\mathbf{x}'} \mathbf{z}^{T'} = \sum_{t=0}^{T'} \beta_t \alpha_{t+1} \alpha_{t+2} \ \ldots \ \alpha_{T'} \tag{8}$$

Now, since $\mathbf{z}^{T'} = \mathbf{z}^*$ (condition (a)), we have

$$\begin{aligned} \alpha_{T'} = \nabla_{\mathbf{z}^{T'}} \mathbf{z}^{T'+1}\Big|_{\mathbf{x}'} &= \nabla_{\mathbf{z}^{T'}} \mathbf{z}^{T'} - \eta' \nabla_{\mathbf{z}^{T'}} \nabla_{\mathbf{z}^{T'}} l_G(\mathbf{z}^{T'}; \mathbf{x}') = \mathbf{I} - \eta' \nabla^2_{\mathbf{z}^*, \mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*) = \mathbf{A} \\ \beta_{T'} = \nabla_{\mathbf{x}'} \mathbf{z}^{T'+1}\Big|_{\mathbf{z}^{T'}} &= \mathbf{0} - \eta' \nabla_{\mathbf{x}'} \nabla_{\mathbf{z}^{T'}} l_G(\mathbf{z}^{T'}; \mathbf{x}') = -\eta' \nabla_{\mathbf{x}', \mathbf{z}^*} l_G(\mathbf{x}', \mathbf{z}^*) = \mathbf{B} \end{aligned} \tag{9}$$

Importantly, since $\mathbf{z}^0_{\text{new}}$ is close to $\mathbf{z}^*$ and l_G is smoothed enough (condition (c)), we have

$$\alpha_0 \approx \alpha_1 ... \approx \alpha_{T'} = \mathbf{A}, \qquad \beta_0 \approx \beta_1 ... \approx \beta_{T'} = \mathbf{B} \tag{10}$$

which yields

$$\nabla_{\mathbf{x}'} \mathbf{z}^{T'} \approx \sum_{t=0}^{T'} \mathbf{B}\mathbf{A}^t \tag{11}$$

substituting $\mathbf{z}^* = \mathbf{z}^{T'}$ completes the proof. □

Unifying Lemma 1 and Lemma 2 immediately yields Theorem 1. □

B. Proof of Theorem 2

Theorem 2. *Let L be the number of layers in G and it only contains four types of operation: (a) ReLU, (b) transposed convolution, (c) max-pooling and (d) full connection. If the weights w^l_{ij} in the l-th layer satisfy $\sum |w^l_{ij}| < \omega$, then*

$$\|G(\mathbf{z}^*) - G(E(\mathbf{x}))\|_2 \leq \omega^L \|\mathbf{z}^* - E(\mathbf{x})\|_2.$$

Proof. We begin by considering two input $\mathbf{x}, \tilde{\mathbf{x}}$ whose latent representation at layer l are $\mathbf{a}^l$ and $\mathbf{a}^{l+1}$ respectively. We below discuss the four cases of feedforward propagation operations in G.

Transposed convolution. The transposed convolution can mathematically expressed as $\mathbf{a}_j^{l+1} = \sum_i \mathbf{w}_{i,j}^l \cdot \text{pad}(\mathbf{a}_i^l)$ with $\text{pad}(\mathbf{a}^l)$ the 'zero padded' version of $\mathbf{a}^l$ and $\mathbf{w}_j^l$ the incoming weights for neuron j. This can be equivalently seen as $\mathbf{a}_j^{l+1} = \sum_i \mathbf{v}_{i,j}^l \mathbf{a}_i^l$ where $\mathbf{v}_j^l$ is a copy of $\mathbf{w}_j^l$ with some of the elements set to be zeros. Then by Cauchy-Schwarz inequality:

$$\|\mathbf{a}_j^{l+1} - \tilde{\mathbf{a}}_j^{l+1}\|_2 \leq (\sum_i (\mathbf{v}_{i,j}^l)^2)^{\frac{1}{2}} \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \tag{12}$$

then

$$\begin{aligned} \|\mathbf{a}^{l+1} - \tilde{\mathbf{a}}^{l+1}\| &\leq (\sum_{i,j} (\mathbf{v}_{i,j}^l)^2)^{\frac{1}{2}} \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \\ &\leq \sum_{i,j} |\mathbf{v}_{i,j}^l| \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \\ &= \sum_{i,j} |\mathbf{w}_{i,j}^l| \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \\ &\leq \omega \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \end{aligned} \tag{13}$$

Full connection. The fully connected layer performs feedfoward computation as $\mathbf{a}_j^{l+1} = \sum_i w_{i,j}^l \mathbf{a}_i^l$. Therefore, similar to the case of transposed convolution,

$$\|\mathbf{a}_j^{l+1} - \tilde{\mathbf{a}}_j^{l+1}\|_2 \leq (\sum_i (\mathbf{w}_{i,j}^l)^2)^{\frac{1}{2}} \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \tag{14}$$

which yields

$$\begin{aligned} \|\mathbf{a}^{l+1} - \tilde{\mathbf{a}}^{l+1}\|_2 &\leq (\sum_{i,j} (\mathbf{w}_{i,j}^l)^2)^{\frac{1}{2}} \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \\ &\leq \sum_{i,j} |\mathbf{w}_{i,j}^l| \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \\ &= \omega \cdot \|\mathbf{a}_i^l - \mathbf{a}_i^l\|_2 \end{aligned} \tag{15}$$

ReLU. Notice that $|max(0, x) - max(0, y)| \leq |x - y|$ for $\forall x, y \in \mathbb{R}$. Therefore:

$$\|\text{ReLU}(\mathbf{a}^l) - \text{ReLU}(\tilde{\mathbf{a}}^l)\|_2 \leq \|\mathbf{a}^l - \tilde{\mathbf{a}}^l\|_2 \tag{16}$$

Max pooling. The max pooling operation can be viewed as a special kind of convolution operation with virtual weight $\mathbf{w}^l$ whose elements are defined as:

$$\mathbf{w}_{ij}^l = \mathbf{1}(i, j = \arg\max_{i',j'} \mathbf{a}_{i',j'}^l) \tag{17}$$

Then

$$\|\text{MaxPool}(\mathbf{a}^l) - \text{MaxPool}(\tilde{\mathbf{a}}^l)\|_2 \leq \|\mathbf{w}^l\|_2 \cdot \|\mathbf{a}^l - \tilde{\mathbf{a}}^l\|_2 = \|\mathbf{a}^l - \tilde{\mathbf{a}}^l\|_2 \tag{18}$$

Unifying (13)(15)(16)(18), for a network with L-layer (remark that ReLU and max-pooling operations do not contribute to the number of layer), we have:

$$\begin{aligned} \|\mathbf{a}^L - \tilde{\mathbf{a}}^L\|_2 &\leq \omega \cdot \|\mathbf{a}^{L-1} - \tilde{\mathbf{a}}^{L-1}\|_2 \\ &\leq \omega^2 \cdot \|\mathbf{a}^{L-2} - \tilde{\mathbf{a}}^{L-2}\|_2 \\ &... \\ &\leq \omega^L \|\mathbf{a}^0 - \tilde{\mathbf{a}}^0\|_2 \\ &= \omega^L \|\mathbf{x} - \tilde{\mathbf{x}}\|_2 \end{aligned} \tag{19}$$

which completes the proof. □

C. Detailed defense settings

The detailed network architectures for the deep generative models used in DefenseGAN and ABS are summarized in Table 1. Here, C is the number of channels in the image (e.g $C = 3$) and d is the dimensionality of the latent representation.

Table 1. Discriminator / Encoder

Stage1	Stage2	Stage3	Stage4	out
conv(C, 256, 4, 2, 1) LeakyReLU(0.2)	conv(256, 512, 4, 2, 1) LeakyReLU(0.2)	conv(512, 1024, 4, 2, 1) LeakyReLU(0.2)	conv(1024, 128, 4, 1, 0) LeakyReLU(0.2)	FC(d)

Table 2. Generator / Decoder

Stage1	Stage2	Stage3	Stage4	out
conv$^\top$(d, 1024, 4, 1, 0) BatchNorm ReLU	conv$^\top$(1024, 512, 4, 2, 1) BatchNorm ReLU	conv$^\top$(512, 256, 4, 2, 1) BatchNorm ReLU	conv$^\top$(256, C, 4, 2, 1)	Tanh

- conv(c, m, k, s, p) refers to convolution with c input channels, m feature maps, filter size $k \times k$, stride s and padding p;
- conv$^\top(c, m, k, s, p)$ refers to the transpose of convolution (sometimes called deconvolution) (Radford et al., 2015) with c input channels, m feature maps, kernel $k \times k$, stride s and padding p;
- LeaklyReLU(r) refers to the leaky version of ReLU unit (Maas et al., 2013) with parameter r;
- FC(d) denotes the fully connected layer with d outputs.
- BatchNorm refers to the batch normalization operation (Ioffe & Szegedy, 2015).

All the GANs are realized as Wasserstein GAN (Arjovsky et al., 2017) trained with gradient penalty (Gulrajani et al., 2017).

D. Detailed attack settings

NES. We estimate the gradient from a population of $n = 50$ samples. These samples are drawn from a search distribution with variance σ^2 set to be $\sigma = 0.5$ in DefenseGAN and $\sigma = 0.1$ in ABS (we discover that the default $\sigma = 0.1$ is not enough to break DefenseGAN in the experiments we consider). We then update the adversarial sample $\mathbf{x}'$ with the estimated gradient by the CW objective:

$$\nabla_{\mathbf{x}'}\mathcal{L}(\mathbf{x}') \approx \nabla_{\mathbf{x}'}\|\mathbf{x}' - \mathbf{x}\|_2^2 + \lambda \cdot \hat{\nabla}^{\text{NES}}_{\mathbf{x}'} F(\mathbf{x}') \tag{20}$$

BPDA. For DefenseGAN (Samangouei et al., 2018), we adopt the same set up as in the original obfuscated gradient literature (Athalye et al., 2018). A CW-like objective is applied so as to achieve minimal distortion. More specifically, it updates the adversarial sample $\mathbf{x}'$ in two steps: (a) forward step, where we find $\mathbf{z}^* = \arg\min_{\mathbf{z}} l_G(\mathbf{z}; \mathbf{x}') = \|G(\mathbf{z}) - \mathbf{x}'\|_2$; (b) backward step, where we update $\mathbf{x}' \leftarrow \mathbf{x}' - \eta \cdot \text{sign}(\nabla_{\mathbf{x}'}\mathcal{L}(\mathbf{x}'))$ with

$$\nabla_{\mathbf{x}'}\mathcal{L}(\mathbf{x}') \approx \nabla_{\mathbf{x}'}\|\mathbf{x}' - \mathbf{x}\|_2^2 + \lambda \cdot \nabla_{\mathbf{x}''} F(\mathbf{x}'')\Big|_{\mathbf{x}'=G(\mathbf{z}^*)} \tag{21}$$

the value of λ are determined by a 5-steps binary search. The value of η is set to be $\eta = 0.04 \cdot t/T$ where t is the current iteration and T is the number of maximum iteration.

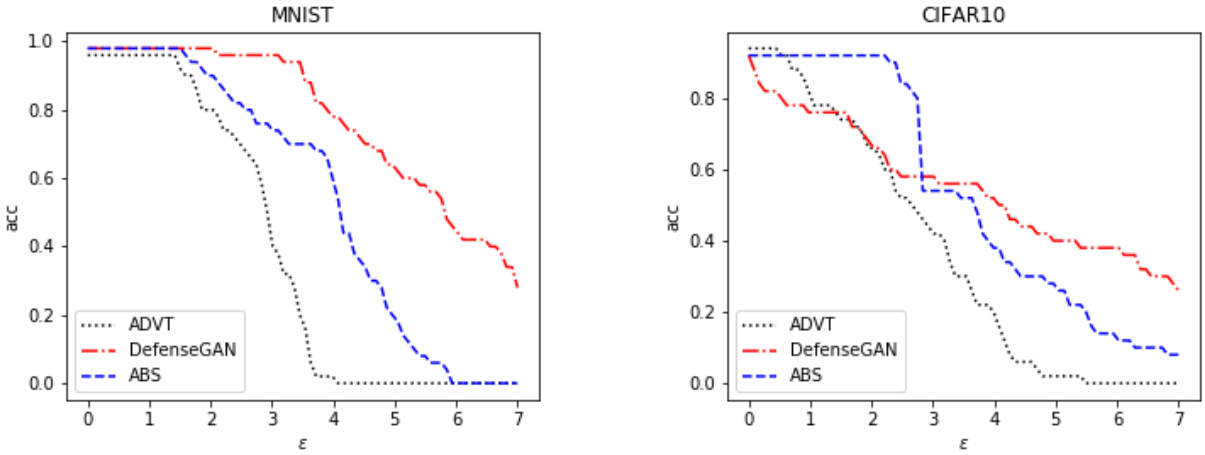
For ABS (Schott et al., 2019), we adopt the same 'latent space attack' as in the ABS literature (Schott et al., 2019), which is variant of BPDA adapted to this defense. It updates the adversarial sample $\mathbf{x}'$ in two steps: (a) forward step, where we find $\mathbf{z}^* = \arg\min_{\mathbf{z}} l_{G_c}(\mathbf{z}; \mathbf{x}') = \|G_c(\mathbf{z}) - \mathbf{x}'\|_2$ with $c \neq y$ the wrong class; (b) backward step, where we update $\mathbf{x}'$ by

$$\mathbf{x}' \leftarrow (1 - \eta) \cdot \mathbf{x}' + \eta \cdot G_c(\mathbf{z}^*) \tag{22}$$

that is, we make a step towards the best reconstruction in the wrong class. We set $\eta = 0.02$ and iterate until we have found $\mathbf{x}'$. To ensure that we can find an adversarial sample within the 75 update steps, we increase the value of η every 25 steps. After we have found $\mathbf{x}'$, a 5-step binary search between $\mathbf{x}$ and $\mathbf{x}'$ is applied to reduce the distortion as much as possible.

E. Comparison among different defenses

We also compares the robustness of DefenseGAN (Samangouei et al., 2018), ABS (Schott et al., 2019) and adversarial training (ADVT) (Madry et al., 2018). Note that when applied to ADVT our attack degenerates to the original CW attack.



(a) MNIST, $d = 32$ (b) CIFAR10, $d = 64$

Figure 1. Comparing the robustness of different defenses in the presence of the proposed attack. The horizontal axis denotes the l_2 distortion and the vertical axis denotes the classification accuracy. Theses results are reported on 100 images only.

Figure 1 shows the accuracy-distortion curve of the various defenses under our attack. Interestingly, we see that despite of our empowered attack, the two deep generative model-based defenses (DefenseGAN, ABS) do seem to be more robust than adversarial training given the experimental setting. A possible explanation is that although DefenseGAN and ABS do not realize the on-manifold conjecture perfectly as analyzed in the paper, they do to some extent exclude some out-off-the-distribution samples, which yield a good protection. We conclude that this kind of defense is still very promising.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pp. 214–223, 2017.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pp. 274–283, 2018.

Gould, S., Fernando, B., Cherian, A., Anderson, P., Cruz, R. S., and Guo, E. On differentiating parameterized argmin and argmax problems with application to bi-level optimization. *arXiv preprint arXiv:1607.05447*, 2016.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pp. 5767–5777, 2017.

Horn, R. A. and Johnson, C. R. *Matrix analysis*. Cambridge university press, 2012.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, pp. 3, 2013.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Samangouei, P., Kabkab, M., and Chellappa, R. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *ICLR*, 2018.

Schott, L., Rauber, J., Bethge, M., and Brendel, W. Towards the first adversarially robust neural network model on mnist. In *ICLR*, pp. 1–16, 2019.

Shaban, A., Cheng, C.-A., Hatch, N., and Boots, B. Truncated back-propagation for bilevel optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1723–1732, 2019.