Instant Adversarial Purification with Adversarial Consistency Distillation

Supplementary Material

1. Appendix

1.1. Additional implementation details

We implement our method with Pytorch [3] and Diffusers. we fix the random seed of PyTorch's generator as 100 for reproducibility [4]. For all implementations, we use the standard version of AutoAttack, which is the same in both the main paper and the appendix.

We leverage LCM-LoRA [2] and TCD-LoRA [8] from their HuggingFace repositories.

In terms of training with our proposed GAND, we use the Stable Diffusion v1.5 (SD15) as the teacher model.

We borrow a part of code and pretrained weights from AMT-GAN when we do experiments on CelebA-HQ. We also conduct a series of IQAs to evaluate the quality of purified images. To be specific, we leverage:

- PSNR: Range set to 1, aligning with PyTorch's image transformation.
- SSIM [6]: Gaussian kernel size set to 11.
- LPIPS [7]: Utilizing VGG [5] as the surrogate model.

Experimenting our method in defending Fog [1]: We set the number of iterations to 10, ϵ to 128, and step size to 0.002236. Snow [1]: We set the number of iterations to 10, ϵ to 0.0625, and step size to 0.002236. L_2-PGD: We set the number of iterations to 100, ϵ to 0.5, and step size to 0.1.

It is worth noting that we borrow the implementation of PSNR from TorchEval. Unless mentioned, all reproducibility-related things follow the above.

1.2. Proofs

We are going to provide some simple proofs for things we have claimed, including

1. $\boldsymbol{z}_t^* \to \boldsymbol{z}$ when $t \to 0$,
2. $\boldsymbol{z}_t^* \to \boldsymbol{\epsilon} + \boldsymbol{\delta}_{adv}$ when $t \to T$
3. $f_\theta(\boldsymbol{z}_{adv}(t), \varnothing, t) \to \boldsymbol{z}_{adv}$ when $t \to 0$
4. $f_\theta(\boldsymbol{z}_{adv}(t), \varnothing, t) - f_\theta(\boldsymbol{z}(t), \varnothing, t) \to \boldsymbol{\delta}_{adv}$ when $t \to 0$

Lemma. If $X \sim \mathcal{N}(\mu, \sigma^2)$ and $\sigma^2 \to 0$, then $X \to \mu$

Proof. For any $\epsilon > 0$,

$$\begin{aligned} P(\|X - \mu\| \geq \epsilon) &= P(\|Z\| \geq \epsilon) \quad Z \sim \mathcal{N}(0, \sigma^2) \\ &\leq \frac{E(X^2)}{\epsilon^2} \\ &= \frac{Var(X) + (E(X))^2}{\epsilon^2} \\ &= \frac{\sigma^2}{\epsilon^2} \\ &\to 0. \end{aligned} \tag{1}$$

The first line to the second line is true by Markov's inequality. Hence, we prove that normal distribution with a vanishing variance will converge in probability to its mean.

Proof 1. β_t is increasing sequence in $t \in \{0, 1, \cdots, T-1, T\}$ in range $(0, 1)$, then we have $\alpha_t = 1 - \beta_t$ is decreasing sequence in $t \in \{0, 1, \cdots, T-1, T\}$ in range $(0, 1)$, $\bar{\alpha}_t$ is decreasing step function in range $(0, 1)$, further assume β_0 is a arbitrarily small number:

$$\begin{aligned} \lim_{t \to 0} \boldsymbol{z}_t^* &= \lim_{t \to 0} \sqrt{\bar{\alpha}_t}\boldsymbol{z} + \sqrt{1 - \bar{\alpha}_t}(\boldsymbol{\epsilon} + \boldsymbol{\delta}_{adv}) \\ &= \sqrt{\alpha_0}\boldsymbol{z} + \sqrt{1 - \alpha_0}(\boldsymbol{\epsilon} + \boldsymbol{\delta}_{adv}) \\ &= \sqrt{1 - \beta_0}\boldsymbol{z} + \sqrt{\beta_0}\boldsymbol{\epsilon} + \sqrt{\beta_0}\boldsymbol{\delta}_{adv} \\ &\to \boldsymbol{z} + \sqrt{\beta_0}\boldsymbol{\epsilon}. \end{aligned} \tag{2}$$

The third line to the fourth line is true by assumption on β_0, and since we know $\boldsymbol{z} + \sqrt{\beta_0}\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{z}, \beta_0 \boldsymbol{I})$ and β_0 is vanishing. By lemma, we have $\boldsymbol{z}_t^* \to \boldsymbol{z}$.

Proof 2. β_t is increasing sequence in $t \in \{0, 1, \cdots, T-1, T\}$ in range $(0, 1)$, then we have α_t is decreasing sequence in $t \in \{0, 1, \cdots, T-1, T\}$ in range $(0, 1)$, $\bar{\alpha}_t$ is decreasing step function in range $(0, 1)$, further assume T is a arbitrarily large number and $\beta_{T/2} > c$ where constant $c \in (0, 1)$, consider $\boldsymbol{\delta}_{adv}$ as constant.

$$\begin{aligned} \bar{\alpha}_T &= \prod_{t=0}^{T} \alpha_t \\ &= \prod_{t=0}^{T} (1 - \beta_t) \\ &= \prod_{t=0}^{T/2} (1 - \beta_t) \prod_{t'=T/2+1}^{T} (1 - \beta_{t'}) \\ &\leq (1 - c)^{T/2} \prod_{t=0}^{T/2} (1 - \beta_t) \\ &\to 0, \\ \lim_{t \to T} \boldsymbol{z}_t^* &= \lim_{t \to T} \sqrt{\bar{\alpha}_t}\boldsymbol{z} + \sqrt{1 - \bar{\alpha}_t}(\boldsymbol{\epsilon} + \boldsymbol{\delta}_{adv}) \\ &= \boldsymbol{\epsilon} + \boldsymbol{\delta}_{adv} \sim \mathcal{N}(\boldsymbol{\delta}_{adv}, \boldsymbol{I}). \end{aligned} \tag{3}$$

$(1-\beta_0)^{T+1}$ converge to 0 by assuming T is arbitrarily large.

Proof 3. β_t is increasing sequence in $t \in \{0, 1, \cdots, T-1, T\}$ in range $(0, 1)$, then we have α_t is linear decreasing sequence in $t \in \{0, 1, \cdots, T-1, T\}$ in range $(0, 1)$, $\bar{\alpha}_t$ is decreasing step function in range $(0, 1)$, further assume $\hat{\epsilon}$ has standard Gaussian distribution and β_0 is a arbitrarily small number,

$$
\begin{aligned}
&\lim_{t\to 0} f_\theta(\boldsymbol{z}_{adv}(t), \varnothing, t) \\
=&\lim_{t\to 0} \frac{\sigma^2}{t^2+\sigma^2}\boldsymbol{z}_{adv}(t) \\
&+ \frac{t^2}{\sqrt{t^2+\sigma^2}}\left(\frac{\boldsymbol{z}_{adv}(t) - \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_{adv}(t), \boldsymbol{c}, t)}{\sqrt{\bar{\alpha}_t}}\right) \\
=&\lim_{t\to 0} \frac{\sigma^2}{t^2+\sigma^2}\boldsymbol{z}_{adv}(t) \\
&+ \lim_{t\to 0}\frac{t^2}{\sqrt{t^2+\sigma^2}}\left(\frac{\boldsymbol{z}_{adv}(t) - \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\epsilon}}}{\sqrt{\bar{\alpha}_t}}\right) \\
=&\lim_{t\to 0}\sqrt{\bar{\alpha}_t}\boldsymbol{z}_{adv} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} - \lim_{t\to 0}\frac{t^2}{\sqrt{t^2+\sigma^2}}\left(\frac{\sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\epsilon}}}{\sqrt{\bar{\alpha}_t}}\right) \\
=&\sqrt{\alpha_0}\boldsymbol{z}_{adv} + \sqrt{1-\alpha_0}\boldsymbol{\epsilon} - \lim_{t\to 0}\frac{t^2}{\sqrt{t^2+\sigma^2}}\left(\frac{\sqrt{1-\alpha_0}\hat{\boldsymbol{\epsilon}}}{\sqrt{\alpha_0}}\right) \\
=&\boldsymbol{z}_{adv} + \sqrt{\beta_0}\boldsymbol{\epsilon} - \lim_{t\to 0}\frac{t^2}{\sqrt{t^2+\sigma^2}}\left(\sqrt{\beta_0}\right)\hat{\boldsymbol{\epsilon}} \\
=&\boldsymbol{z}_{adv} + \lim_{t\to 0}\sqrt{\left(1+\frac{t^4}{t^2+\sigma^2}\right)\beta_0}\boldsymbol{\epsilon}.
\end{aligned}
\tag{4}
$$

The second last line is from assumption on β_0 and the last line is from the property of normal distribution and assumption on $\hat{\boldsymbol{\epsilon}}$. Finally, by the fact that $\lim_{t\to 0}\sqrt{\left(1+\frac{t^4}{t^2+\sigma^2}\right)\beta_0} = 0$ and lemma we have proved, we prove $f_\theta(\boldsymbol{z}_{adv}(t), \varnothing, t) \to \boldsymbol{z}_{adv}$.

Proof 4. Following a similar way in proof 3, we have

$$
\begin{aligned}
&\lim_{t\to 0} f_\theta(\boldsymbol{z}(t), \varnothing, t) \\
=&\boldsymbol{z} + \lim_{t\to 0}\sqrt{\left(1+\frac{t^4}{t^2+\sigma^2}\right)\beta_0}\boldsymbol{\epsilon}, \\
&\lim_{t\to 0} f_\theta(\boldsymbol{z}_{adv}(t), \varnothing, t) - f_\theta(\boldsymbol{z}(t), \varnothing, t) \\
=&\boldsymbol{z}_{adv} - \boldsymbol{z} + \lim_{t\to 0}\sqrt{2\left(1+\frac{t^4}{t^2+\sigma^2}\right)\beta_0}\boldsymbol{\epsilon} \\
\to&\boldsymbol{z}_{adv} - \boldsymbol{z} = \boldsymbol{\delta}_{adv}.
\end{aligned}
\tag{5}
$$

The last line uses the definition of δ_{adv}.

1.3. Experiment

In Fig. 1, we test the robustness of another model that can do few steps generation, Trajectory consistency distillation (TCD) [8], which can also generate an image in one step. We can see that using LCM as a purification model is generally more robust than using TCD. Also, the standard accuracy of the two models is similar. Therefore, we choose LCM as our backbone model for purification instead of TCD.

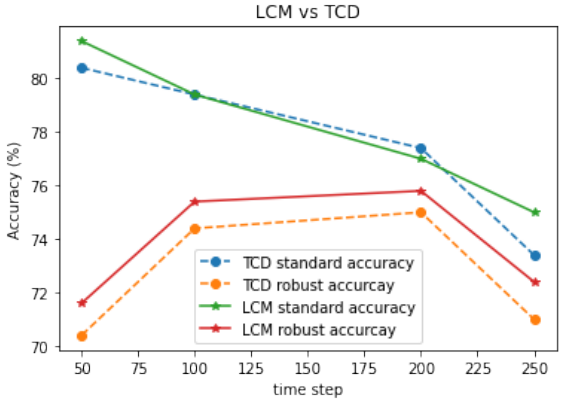

Figure 1. Accuracy (%) using LCM and TCD model as defense model under L_∞-PGD attack on ImageNet.

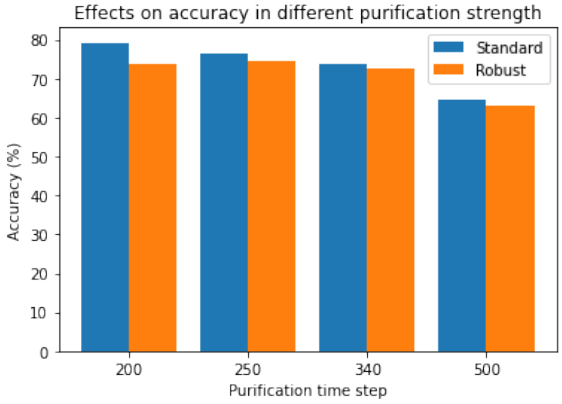

Figure 2. Accuracy (%) on different purification time step t^* on our method. Two figures have the same attack setting, PGD-100 $L_\infty\gamma$ ($\gamma = 4/255$), step size 0.01 * 4/255, where both we evaluate on ResNet50 on 500 images subset of ImageNet.

Table 1. Inference time (s) of our method on different inference time and resolution

Method	Resolution	$t^*=20$	$t^*=60$	$t^*=100$	$t^*=250$	$t^*=500$
DiffPure	256×256	~ 2	~ 6	~ 10	~ 24	~ 45
DiffPure	512×512	~ 6	~ 17	~ 27	~ 70	~ 140
DiffPure	1024×1024	~ 30	~ 85	~ 145	~ 360	~ 720
Ours	256×256	~ 0.05	~ 0.05	~ 0.05	~ 0.05	~ 0.05
Ours	512×512	~ 0.1	~ 0.1	~ 0.1	~ 0.1	~ 0.1
Ours	1024×1024	~ 0.5	~ 0.5	~ 0.5	~ 0.5	~ 0.5

In Fig. 2, we show the experiment for choosing the purification time step on ImageNet. Our model achieved the best performance at $t^* = 200$ in both standard accuracy and robust accuracy. Hence, we decide to choose $t^* = 200$ for our method on ImageNet.

In Tab. 1, we further test the inference time of our method on three resolutions without resizing on an NVIDIA F40 GPU. This result showcases that if our GAND weights

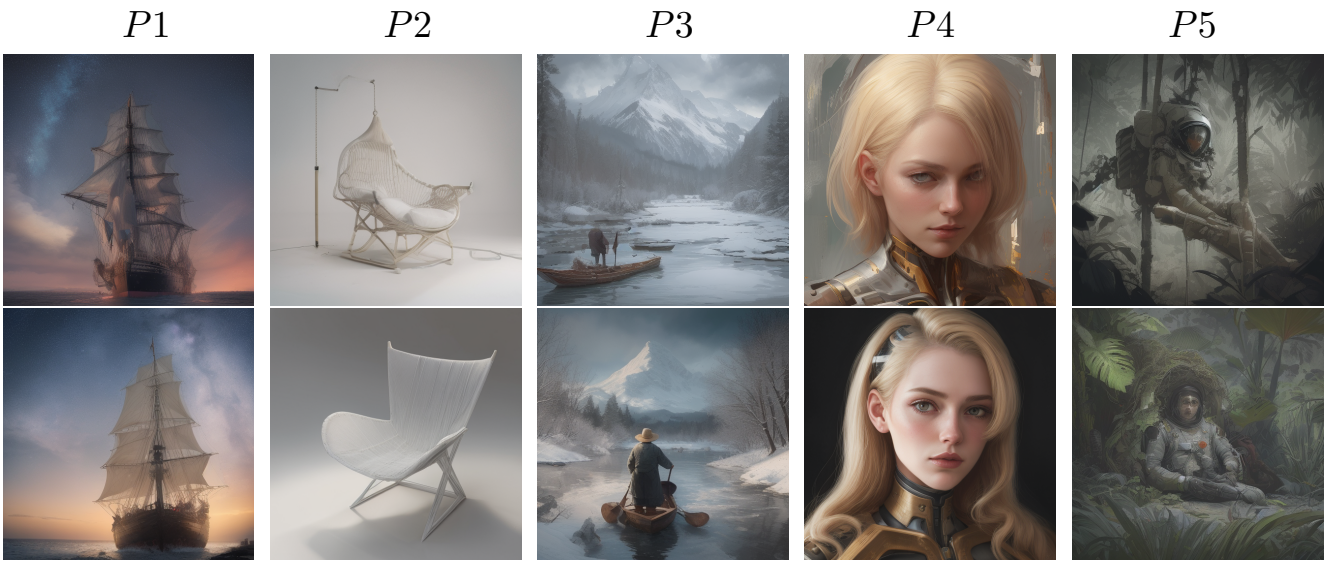

Figure 3. Visualization of T2I (text to image) generation in different text prompts making use of our GAND weight. P1: A gorgeous ship sails under a beautiful starry sky. P2: Cradle chair, a huge feather, sandy beach background, minimalism, product design, white background, studio light, 3d, iso 100, 8k. P3: In a serene, snowy landscape, an elderly man in a straw raincoat and hat fishes alone on a small wooden boat in a calm, cold river. Surrounded by snow-covered trees and mountains, the tranquil scene conveys harmony with nature and quiet isolation. Masterpiece. Chinese art. extreme details. P4: Self-portrait oil painting, a beautiful cyborg with golden hair, 8k. P5: Astronauts in a jungle, cold color palette, muted colors, detailed, 8k

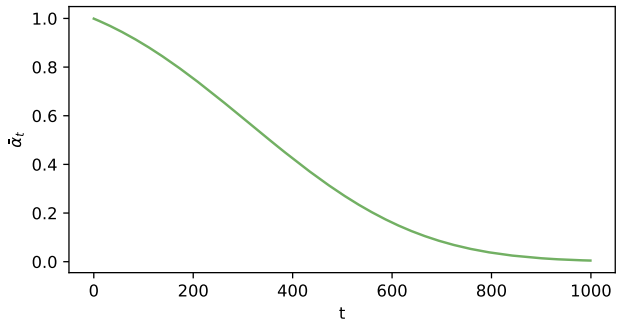

Figure 4. Visualization of how the term $\bar{\alpha}_t$ change by t

are trained on those resolutions, what will the inference time of our method be. As shown in Tab. 1, DiffPure takes around 12 minutes to purify a 1024×1024 image when t^* is chosen to be 500. Meanwhile, our method only takes 0.5 seconds to purify a 1024×1024 image on any t^*, showing the potential of our method for purification in high-resolution images since our method does the purification in the latent space, which is much more efficient than pixel space.

In Fig. 4, we show how $\bar{\alpha}_t$ variate by t in the LCM scheduler. We can see that the value of $\bar{\alpha}_t$ starts at almost 1 ($t = 0$) and decreases to almost 0 ($t = 1000$), which meets with the assumptions we have made in our proofs.

In Tab. 2, we conducted four more attack methods on 500 images subset of ImageNet; we can see that our method is generally more robust than DiffPure, which is 1% and 2.6% higher than DiffPure when defending L_2-PGD and Snow. A slightly higher robust accuracy in defending StAdv and the same robust accuracy in defending Fog. Also, our method has a higher standard of accuracy, which is 1.6% higher than DiffPure, showing that our method prevents semantic loss in the purification process. In Fig. 5, we provide more visual examples.

Table 2. Standard accuracy and robust accuracies against unseen threat models on ResNet-50, L_2-PGD, StAdv, Snow, Fog

Method	Standard	L_2-PGD	StAdv	Snow	Fog
DiffPure	80.8%	80.6%	69.4%	77.2%	**77.8%**
Ours	**82.4%**	**81.6%**	**69.8%**	**79.8%**	**77.8%**

Although our goal is purification, it is still interesting for us to visualize the image generated by our GAND weight. We use LCM-LoRA and change their LoRA to our GAND LoRA weight. Then, we generate pictures by text prompt P1 (ship), P2 (chair), P3 (elderly man), P4 (Girl), and P5 (Astronauts); the full-text prompt is shown in the caption of Fig. 3. We can see that our LCM-LoRA (GAND) still maintains generability; this is a surprising result and shows that our objective might be useful for some img2img tasks if people edit our objective correctly for their task.

(a) (b) (c) (d)

Figure 5. Visualization of the experiment L_2-PGD, StAdv, Snow, Fog respectively, which compares with DiffPure and the proposed method. (a) Input image. (b) Adversarial image. (c) DiffPure. (d) Ours. The last row presents the proposed method (a) Input image (b) Adversarial image (c) Edge image (d) Purified Image

References

[1] Max Kaufmann, Daniel Kang, Yi Sun, Steven Basart, Xuwang Yin, Mantas Mazeika, Akul Arora, Adam Dziedzic, Franziska Boenisch, Tom Brown, et al. Testing robustness against unforeseen adversaries. *arXiv preprint arXiv:1908.08016*, 2019. 1

[2] Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. Lcm-lora: A universal stable-diffusion acceleration module. *arXiv e-prints*, pages arXiv–2311, 2023. 1

[3] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[4] David Picard. Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *arXiv preprint arXiv:2109.08203*, 2021. 1

[5] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 1

[6] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 1

[7] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[8] Jianbin Zheng, Minghui Hu, Zhongyi Fan, Chaoyue Wang, Changxing Ding, Dacheng Tao, and Tat-Jen Cham. Trajectory consistency distillation. *arXiv preprint arXiv:2402.19159*, 2024. 1, 2