APPENDIX OVERVIEW

This appendix provides additional details and analyses to complement the main paper. It is organized as follows:

- **Section A. Use of Large Language Models.** We clarify the extent to how LLMs were used during the writing and proofreading process, ensuring transparency in compliance with conference policies.
- **Section B. Background on Adversarial Attacks and Defenses.** We review standard adversarial attacks (e.g., PGD, AutoAttack, BPDA) and defense paradigms (adversarial training, purification), offering context for how our method relates to existing approaches.
- **Section C. Theoretical Supplement.** We provide a more complete derivation of diffusion models, present a unified mathematical framework for adversarial purification, and analyze the computational complexity and stability of different approaches.
- **Section D. Experimental Settings.** We detail the hyperparameter choices for both attacks and diffusion models, including perturbation budgets, iteration numbers, noise schedules, and pretrained checkpoints, ensuring reproducibility of all results.
- **Section E. Additional Experimental Results.** We extend the evaluations beyond the main text. This includes: (i) a step-by-step algorithmic workflow of our framework. (ii) classification with alternative backbones (CLIP-RN101, WRN-28-10,RN-50), (iii) plug-and-play integration under ℓ_2 attacks, (iv) analysis of PGD iteration numbers, and
- **Section F. Visualization.** We provide additional qualitative results, showing purified versus adversarial samples across multiple datasets, highlighting the semantic preservation and noise suppression of our method.

A STATEMENT ON THE USE OF LLMS

This study employed LLMs to assist in writing. LLMs were primarily utilized for language refinement, grammatical corrections, and enhancing academic tone. It is crucial to emphasize that all viewpoints, theoretical frameworks, experimental results, and final conclusions were independently developed by human authors. LLMs served solely as auxiliary tools for manuscript refinement, with all final drafts thoroughly reviewed and approved by the authors.

B SUPPLEMENT RELATED WORK

Adversarial Attacks & Robustness. Adversarial attacks have long revealed the fragility of neural networks, beginning with the discovery of imperceptible perturbations by Szegedy et al. (2013) and the efficient one-step FGSM attack (Goodfellow et al., 2014). Iterative methods such as PGD (Madry et al., 2017) established strong benchmarks for robustness evaluation, later extended by efficient variants like FreeAT (Shafahi et al., 2019) and AutoAttack (Croce & Hein, 2020). The use of EOT (Expectation over Transformation) (Athalye et al., 2018) was further emphasized to mitigate randomness and non-differentiability in gradients, ensuring accurate robustness assessment. On the defense side, adversarial training (Schlarmann et al., 2024; Mao et al., 2023) remains the most widely used strategy. By incorporating adversarial examples into the training process, AT explicitly improves the decision boundary against perturbations, thereby enhancing robustness. However, AT requires significant computational resources and often generalizes poorly to unseen attacks, motivating research into alternative approaches.AP emerged in response to this situation.

C THEORETICAL SUPPLEMENT

C.1 UNIFIED FRAMEWORK FOR ADVERSARIAL PURIFICATION

We can unify diffusion-based adversarial purification methods into the following generalized formulation:

$$x_t = f(x_0; \bar{\alpha}_t) + g(\epsilon; \mathbf{W}), \tag{15}$$

where $f(x_0; \bar{\alpha}_t) = \sqrt{\bar{\alpha}_t}\, x_0$ denotes the signal decay term, $g(\epsilon; \mathbf{W})$ represents noise injection, and $\mathbf{W}$ is a weighting or transformation operator.

- **Adversarial Training:** robustness stems from model parameters; no explicit $g(\cdot)$ is introduced.
- **DiffPure:** $g(\epsilon; \mathbf{W}) = \sqrt{1 - \bar{\alpha}_t}\, \epsilon$, where $\mathbf{W} = I$.
- **MANI-Pure:** $g(\epsilon; \mathbf{W}) = \sqrt{1 - \bar{\alpha}_t}(\mathbf{W} \odot \epsilon)$, where $\mathbf{W}$ is derived from frequency magnitudes.
- **FreqPure:** constraints are imposed in the *reverse* step, by spectral recombination rather than forward-side weighting.

This unified framework highlights a key dichotomy: *forward-side approaches* redesign $g(\cdot)$ to better mimic adversarial distributions, while *reverse-side approaches* constrain the reconstruction trajectory. MANI-Pure naturally combines both perspectives, explaining its superior performance.

C.2 Complexity and Stability Analysis

Time Complexity:

- **DiffPure:** $O(T \cdot HW)$ per reverse trajectory, dominated by neural network inference.
- **MANI-Pure:** adds DFT/IDFT operations of $O(HW \log(HW))$ per step, negligible compared to network cost.
- **FreqPure:** incurs extra spectral recombination and projection, but all operations are element-wise or FFT-based, remaining parallelizable on GPUs.
- **Hybrid methods (e.g., MANI+FreqPure):** maintain linear scaling in T and near-constant overhead relative to the diffusion backbone.

Space Complexity:

- All methods store $O(HW)$ activations per step.
- Frequency-based approaches require one additional complex-valued copy of the spectrum, i.e., $O(2HW)$, which is marginal compared with feature maps inside the denoiser.

Numerical Stability:

- FFT and inverse FFT are unitary transforms, introducing no instability.
- MANI's band-wise weighting may amplify small magnitudes, but normalization with ϵ ensures bounded variance.
- FreqPure's projection operator $\Pi(\cdot)$ restricts phase drift, effectively stabilizing the reverse trajectory under strong attacks.

Scalability. Since the extra overhead scales sub-linearly with resolution ($\log(HW)$), frequency-domain operations remain efficient even for high-resolution ImageNet-1K images. Therefore, the proposed MANI-Pure achieves robustness gains without sacrificing efficiency.

D Parameters and Settings

D.1 Attack setup

We adopt three types of strong adaptive attacks: PGD+EOT, AutoAttack, and BPDA+EOT. For PGD and BPDA, the number of iterations is set to 10 (the rationale for this choice is discussed in Appendix E.4), while the number of EOT samples is also set to 10. AutoAttack is executed in its standard version, which integrates `APGD-CE`, `APGD-DLR`, `FAB`, and `Square Attack`, with 100 update iterations. The perturbation budget is $\epsilon = 8/255$ for ℓ_∞ attacks on CIFAR-10 and $\epsilon = 4/255$ on ImageNet-1K, while ℓ_2 attacks use $\epsilon = 0.5$ for both datasets. Unless otherwise specified, the step size is set to 0.007 for all attacks.

D.2 DIFFUSION SETUP

Our purification framework is based on DDPM++ (Song et al., 2020) with a linear variance schedule, where the noise variance increases from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$ over $T = 1000$ steps (Ho et al., 2020). In all experiments, we set the forward noising steps to 100 and the reverse denoising steps to 5, unless otherwise specified. For DiffPure, we follow the original implementation and use 100 reverse steps. The pretrained diffusion weights are taken from public releases: the unconditional CIFAR-10 checkpoint of EDM (Karras et al., 2022) and the 256×256 unconditional diffusion checkpoint for ImageNet-1K, consistent with prior works.

D.3 NOISE DIFFERENCE HEATMAP COMPUTATION

To analyze the similarity between injected noise N_{inj} and adversarial noise N_{adv}, we compute their pixel-wise difference:

$$D = N_{\text{inj}} - N_{\text{adv}}. \tag{16}$$

Here D contains both positive and negative values, where the sign indicates whether the injected noise is larger or smaller than the adversarial noise at each pixel. For visualization, we normalize D and render it with a diverging colormap, where red/blue colors represent positive/negative differences, respectively.

E ADDITIONAL RESULTS

E.1 THE ALGORITHM WORKFLOW OF MANI-PURE

This section presents the **MANI-Pure** algorithm flowchart (Algorithm 1), which comprehensively illustrates the entire processing workflow. This contrasts with the section-by-section module introductions in Sec. 3.2 and the abstract representation in Figure 2.

Algorithm 1 Adversarial Purification with MANI and FreqPure

Require: Adversarial input x_{adv}, Diffusion steps T, Band number n, Weighting factor γ
Ensure: Purified image x_0

$(A_{\text{adv}}, \Phi_{\text{adv}}) = \mathcal{F}(x_{\text{adv}})$
Partition M_{adv} into n frequency bands $\{B_i\}$ // Forward Progress:MANI
for each band B_i **do**
  $M_i = \frac{1}{|B_i|} \sum_{(u,v)\in B_i} A_{\text{adv}}(u, v)$
  $w_i = (M_i + \epsilon_0)^{-\gamma}$
end for
Construct spatial weight map W via IDFT
$\epsilon_t = W \odot \epsilon_G$, with $\epsilon_G \sim \mathcal{N}(0, I)$
$x_t = \sqrt{\bar{\alpha}_t}\, x_{\text{adv}} + \sqrt{1 - \bar{\alpha}_t}\, \epsilon_t$
Initialize $x_T \sim \mathcal{N}(0, I)$ // Reverse Progress:FreqPure
for $t = T \to 1$ **do**
  $x_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}}\big(x_t - \sqrt{1 - \bar{\alpha}_t}\, \epsilon_\theta(x_t, t)\big)$
  $(A_t, \Phi_t) = \mathcal{F}(x_{0|t})$
  $A^{t-1} = \mathcal{H}(A_{\text{adv}}) + (1 - \mathcal{H})(A_t)$
  $\Phi^{t-1} = \mathcal{H}\big(\Pi(\Phi_t, \Phi_{\text{adv}}, \delta)\big) + (1 - \mathcal{H})(\Phi_t)$
  $x_{t-1} = \mathcal{F}^{-1}(A^{t-1}, \Phi^{t-1})$
end for
return x_0

E.2 ROBUSTNESS UNDER DIFFERENT BACKBONES

In this section, we further supplement classification experiments with CLIP (RN101), WRN-28-10 (Zagoruyko & Komodakis, 2016) and ResNet-50 (He et al., 2016), following the same settings as Sec. 4.1 in the main text. As shown in Table 1, Table 2, Table 7, Table 8 and Table 9, **MANI-Pure**

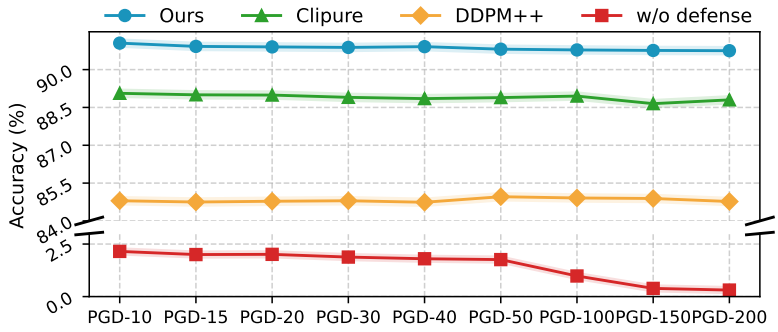

Figure 6: Robust accuracy of several purification methods across different PGD iteration counts (All attacks with EOT=10).

consistently achieves the best performance across different classifier architectures, demonstrating its versatility and robustness.

Table 7: Classification accuracy on CIFAR-10 under adversarial attacks using CLIP RN101. Zero-shot CLIP (w/o defense) is denoted by †; its standard accuracy as the upper bound. Only AP-based methods are included.

Algorithm	**Standard**	**PGD**		**AutoAttack**		**BPDA**
		ℓ_∞	ℓ_2	ℓ_∞	ℓ_2	
Zero-shot (w/o defense) †	78.32	0.00	26.56	0.20	0.20	2.73
+ *DiffPure* (Nie et al., 2022)	67.58	65.98	66.60	65.62	66.60	66.01
+ *DDPM++* (Song et al., 2020)	68.95	65.62	66.99	64.45	66.80	65.62
+ *REAP* (Lee & Kim, 2023)	62.30	61.33	61.72	61.91	61.13	61.91
+ *FreqPure* (Pei et al., 2025b)	70.70	68.55	68.95	67.97	68.75	66.80
+ *CLIPure* (Zhang et al., 2025b)	68.95	62.89	68.75	64.26	68.84	59.18
+*Ours*	**71.88**	**68.75**	**70.12**	**69.43**	**70.12**	**69.53**

E.3 Plug-and-Play Results under ℓ_2 Attacks

In addition to the ℓ_∞ setting reported in the main text, we also evaluate the plug-and-play integration of MANI with existing AP methods under ℓ_2 attacks. Following the same configurations as Sec. 4.1, we consider PGD+EOT and AutoAttack with perturbation budget $\epsilon = 0.5$. The results, summarized in Table 10, show that MANI consistently improves both clean and robust accuracy when combined with different AP backbones.

E.4 Effect of Attack Iterations

We also examine the impact of the number of PGD iterations on robust accuracy. In our main experiments, we set PGD iterations to 10. Since prior works adopt different iteration counts, we perform an ablation to validate this choice. As illustrated in Figure 6, the robust accuracy of undefended models decreases sharply with more iterations and converges near zero, while defense methods remain relatively stable with only minor fluctuations. Therefore, we adopt 10 iterations as a practical **balance between robustness evaluation and computational efficiency**. Additionally, for EOT iterations, we follow the setting in Nie et al. (2022), which shows that robustness converges once EOT exceeds 10.

Table 8: Classification accuracy on CIFAR-10 under adversarial attacks using WRN-28-10. WRN-28-10(w/o defense) is denoted by †; its standard accuracy as the upper bound. Results marked with ‡ are reported in Bai et al. (2024). Only AP-based methods are included.

Algorithm	Standard	PGD	AutoAttack
WRN-28-10 (w/o defense) †	96.48	0.00	0.00
+*Diffpure*(Nie et al., 2022)	90.07	56.84	63.30
+*REAP*(Lee & Kim, 2023)	90.16	55.82	70.47
+*CGDM*(Bai et al., 2024)‡	91.41	49.22	77.08
+*FreqPure*(Pei et al., 2025b)	92.19	59.39	77.35
+*Ours*	**92.57**	**61.32**	**78.69**

Table 9: Classification accuracy on CIFAR-10 under adversarial attacks using ResNet-50. ResNet-50(w/o defense) is denoted by †; its standard accuracy as the upper bound. Results marked with ‡ are reported in Bai et al. (2024). Only AP-based methods are included.

Algorithm	Standard	PGD	AutoAttack
ResNet-50 (w/o defense) †	76.01	0.00	0.00
+*Diffpure*(Nie et al., 2022)	67.84	42.58	41.53
+*REAP*(Lee & Kim, 2023)	68.72	43.19	44.67
+*CGDM*(Bai et al., 2024)‡	68.98	41.80	-
+*FreqPure*(Pei et al., 2025b)	69.53	59.77	**63.49**
+*Ours*	**70.31**	**60.03**	61.79

Table 10: **Plug-and-play validation of the MANI module under ℓ_2 attacks.** We integrated MANI into various diffusion-based purification frameworks and evaluated them on CIFAR-10. Results are reported both without MANI (**w/o**) and with MANI (**w/**).

Algorithm	PGD		AutoAttack	
	w/o	w/	w/o	w/
+ *DiffPure* (Nie et al., 2022)	85.74	87.08	85.55	87.50
+ *DDPM++* (Song et al., 2020)	85.16	86.72	85.74	87.11
+ *REAP* (Lee & Kim, 2023)	79.87	81.64	80.18	81.84
+ *FreqPure* (Pei et al., 2025b)	91.41	92.58	92.00	93.16

F VISUALIZATION

To intuitively illustrate the purification effect, we present qualitative results on randomly selected samples from CIFAR-10 (Figure 7, Figure 8, Figure 9) and ImageNet-1K (Figure 10, Figure 11, Figure 12), including clean images, adversarial images, and purified images.

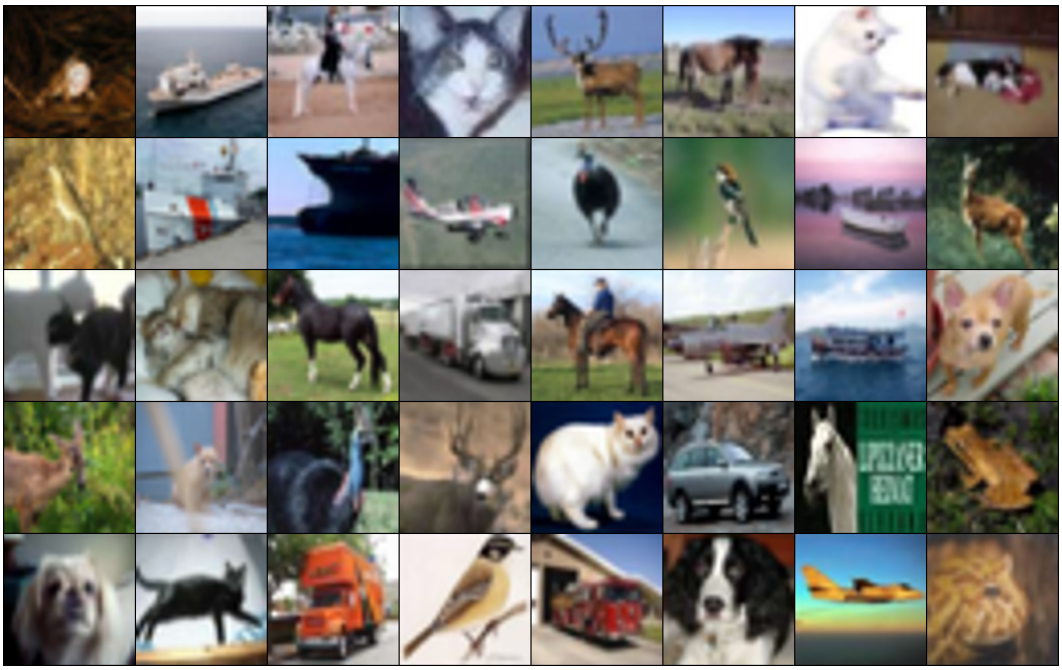

Figure 7: **Clean** CIFAR-10 images randomly selected for visualization

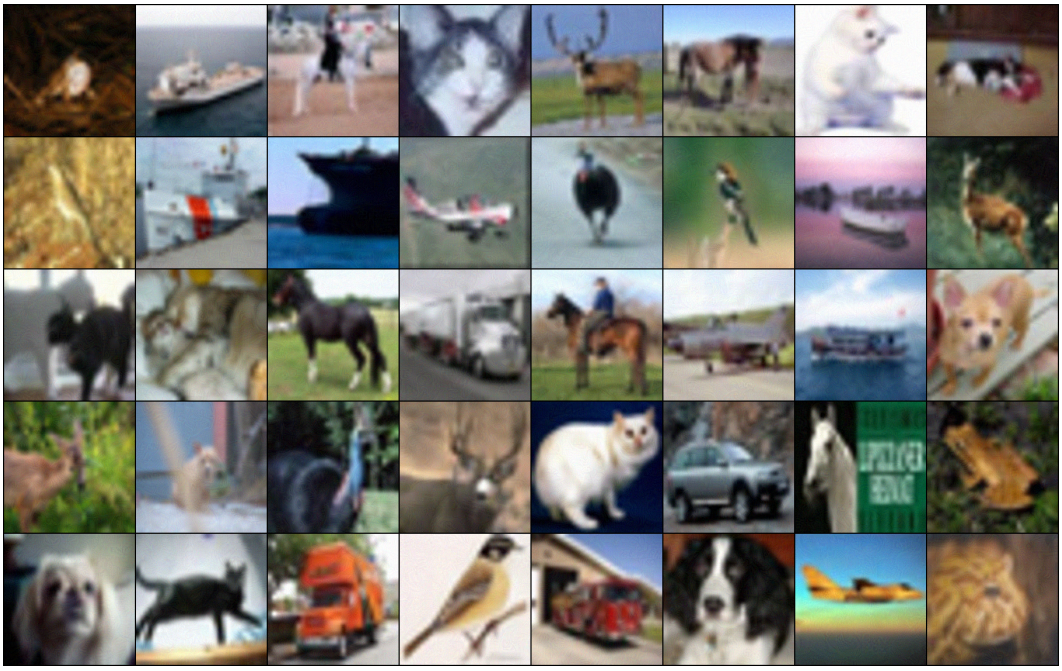

Figure 8: **Adversarial** CIFAR-10 images randomly selected for visualization

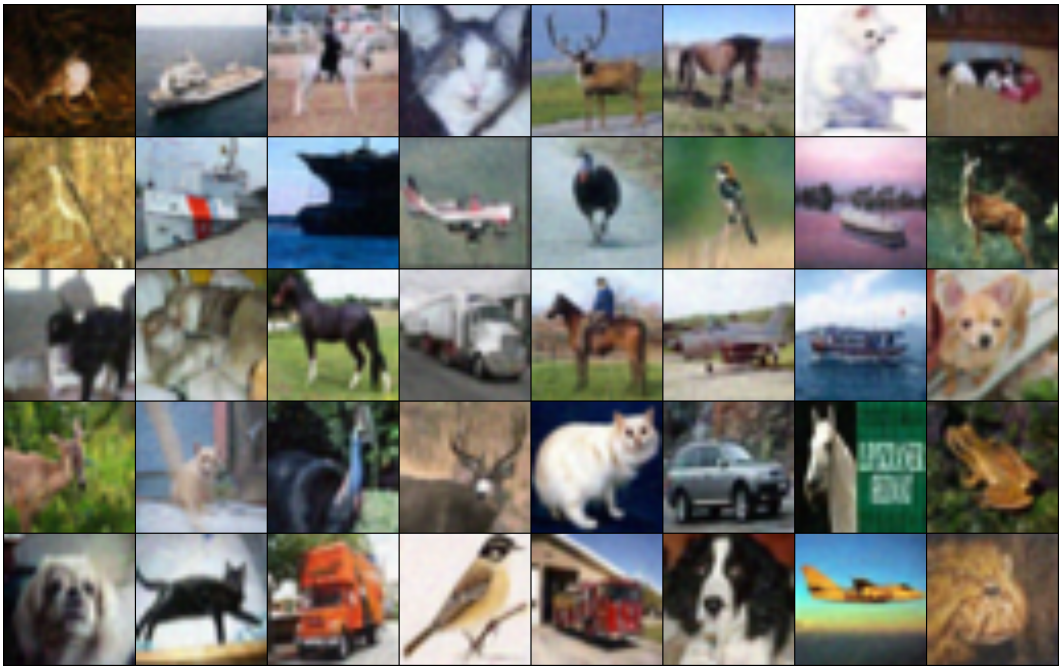

Figure 9: **Purified** CIFAR-10 images randomly selected for visualization

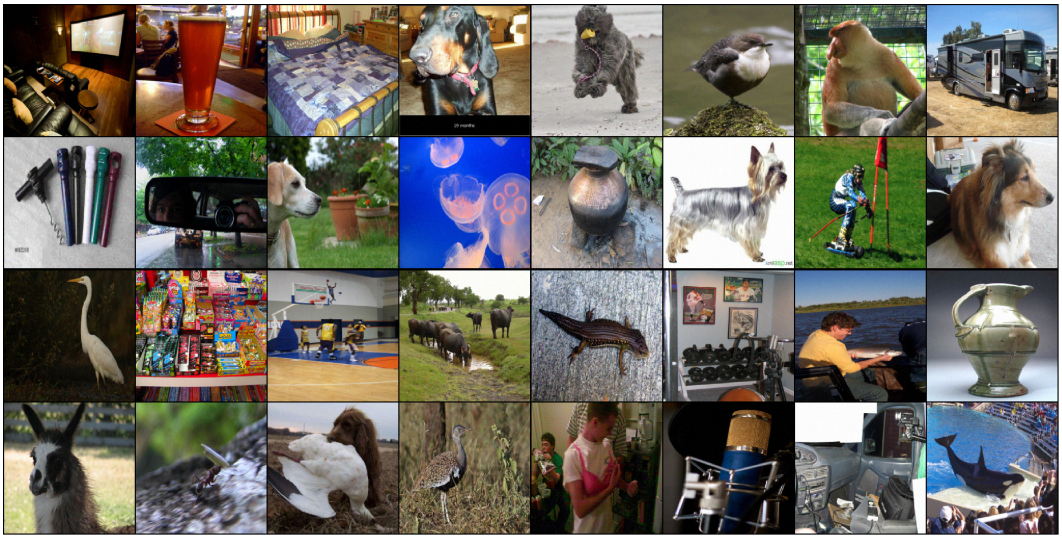

Figure 10: **Clean** ImageNet-1K images randomly selected for visualization

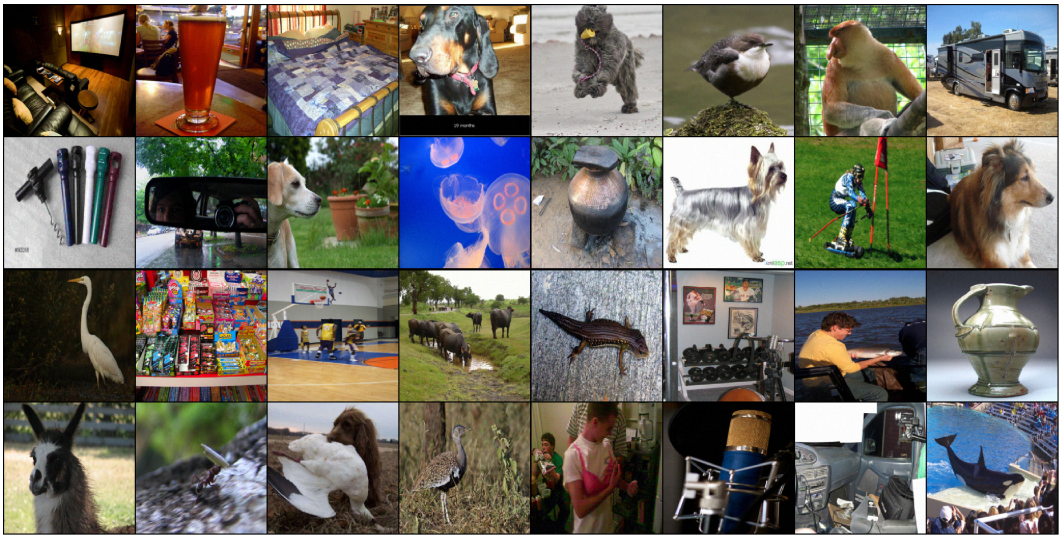

Figure 11: **Adversarial** ImageNet-1K images randomly selected for visualization

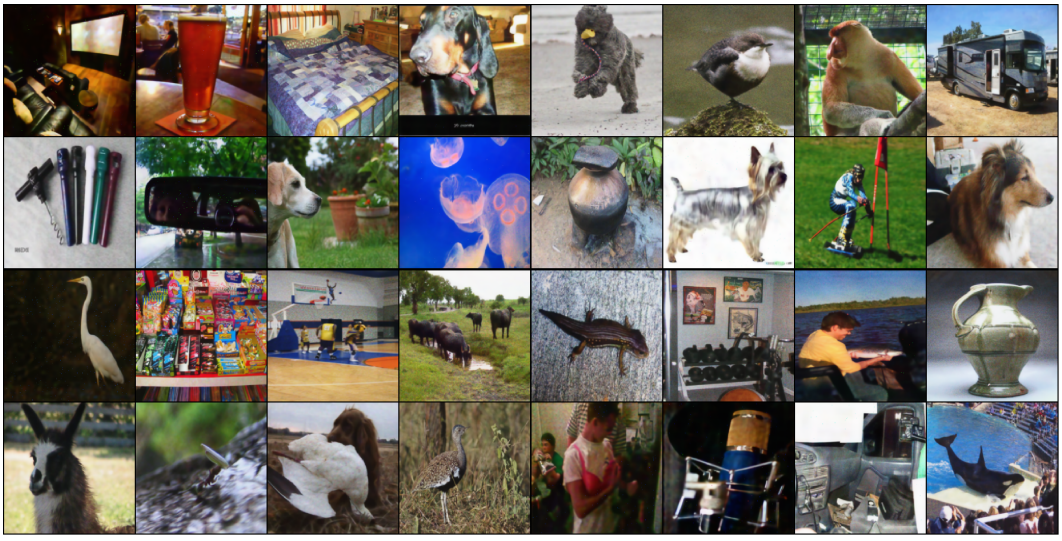

Figure 12: **Purified** ImageNet-1K images randomly selected for visualization