Pre-trained Multiple Latent Variable Generative Models are good defenders against Adversarial Attacks

Supplementary Material

In completion of the main paper, we provide additional information regarding experimental content and qualitative results. This Supplementay Material is organized as follows: Appendix A contains the pseudocode of the purification procedure; Appendix B a qualitative analysis of images generated following the **learned** set of α hyperparameters; Appendix C experimental details that can help in reproducing the results; Appendix D the ablation studies and details on preprocessing operations; Appendix E shows some qualitative purification examples of our method, in comparison to A-VAE [78] and ND-VAE [28]; and Appendix F contains additional defense results on the Autoattack method [14].

A. Pseudocode

In Algorithm 1 we show the pseudocode for the purification procedure, as explained in Section 3 of the main paper.

Algorithm 1 Purification Procedure

```
Require: Input Image $\mathbf{x}$; Preprocessing Function $\mathcal{P}$; Encoder and MLVGM $\mathcal{E}, \mathcal{G}$; Generator's Prior $\mathcal{N}$; Hyperparameters $\{\alpha_0, \alpha_1, \dots, \alpha_{N-1}\}$.
1: $\mathbf{x}_\mathrm{p} = \mathcal{P}(\mathbf{x})$                                  ▷ optional preprocessing
2: $\mathbf{z}^\mathrm{e}_0, \mathbf{z}^\mathrm{e}_1, \dots, \mathbf{z}^\mathrm{e}_{N-1} = \mathcal{E}(\mathbf{x}_\mathrm{p})$          ▷ encoding
3: for $i = 0$ to $N-1$ do
4:     $\mathbf{z}^\mathrm{s}_i \sim \mathcal{N}$                                    ▷ sampling
5:     $\mathbf{z}_i = (1-\alpha_i)\mathbf{z}^\mathrm{e}_i + \alpha_i \mathbf{z}^\mathrm{s}_i$             ▷ interpolation
6: end for
7: $\hat{\mathbf{x}} = \mathcal{G}(\mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_{N-1})$                 ▷ decoding
8: return Purified Image $\hat{\mathbf{x}}$
```

B. Qualitative Analysis of Bayesian Optimization

In Figures 3 to 5 (b) of the main paper we show the hyperparameter values **learned** by Bayesian Optimization, compared to the **linear** and **cosine** case. For the identity classification task, the combination is monotonic and matches the idea that MLVGMs represent features in global-to-local manner, with class relevant information mainly contained in the first latent levels, and gradually decreasing. However, for the coarse-grained tasks (gender and car types classification), the **learned** combinations tend to assign low α values to specific intermediate latents. These are $i = 2, 5, 6$ and $i = 1, 2, 4, 5, 9, 13$ for the two tasks, respectively. In other terms, Bayesian Optimization suggests to maintain the original information of these latent levels, and to discard and re-sample the remaining codes. This means that the optimization process defines two types of codes: "class codes" which should maintain original information, and "detail codes" that can be re-sampled. In Figures 9 and 10 we qualitatively verify this aspect, by mixing the class and detail codes given by BO of two samples (A and B), of *different* classes. The second and third column in the figures shows what happens when mixing class codes of one sample with detail codes of the other. As visible, it is true that class-relevant and irrelevant information is disentangled, allowing to change the details of an image without altering its label. This qualitative analysis suggests that Bayesian Optimization is effective in distinguishing class-relevant features, assigning low α values to the corresponding latent levels. Furthermore, the visualization supports the hypothesis made in the main paper: in coarse-grained classification tasks, the class-relevant information is contained in a few, intermediate latents.

C. Experimental Details

Obtaining the pre-trained MLVGMs. For the Celeb-A HQ and Cars tasks we use the official StyleGan-2 checkpoints, coupled with the corresponding encoders. The public repositories are https://github.com/omertov/encoder4editing for faces and https://github.com/sapphire497/style-transformer for cars. These allow to download the pre-trained Autoencoders, based on StyleGan-2.
Regarding NVAE, we train our own model on the Celeb-A dataset at resolution 64×64. The model has 3 scales of 8 groups, for a total of 24 latent variables. We trained the model for 600 epochs, with a cumulative batch size of 256. The learning rate has been decayed from $1e-3$ to $5e-4$. The total number of parameters is 705.678 M. The pre-trained model, code and complete training configuration are available at https://github.com/SerezD/NVAE-from-scratch.

Creating the Datasets. The instructions on how to download the gender classification dataset are available at: https://github.com/ndb796/CelebA-HQ-Face-Identity-and-Attributes-Recognition-PyTorch. For Celeba-64 identity classification, we obtain the original data and annotated identities from the Celeb-A project page: https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html. The dataset comprises 10.177 identities. First, we center crop and resize each image to achieve the correct resolution of 64×64. Then, we select a random subset of 100 identities, between these that have at least 15 samples per class. For Stanford Cars, we download the original dataset from Kaggle,

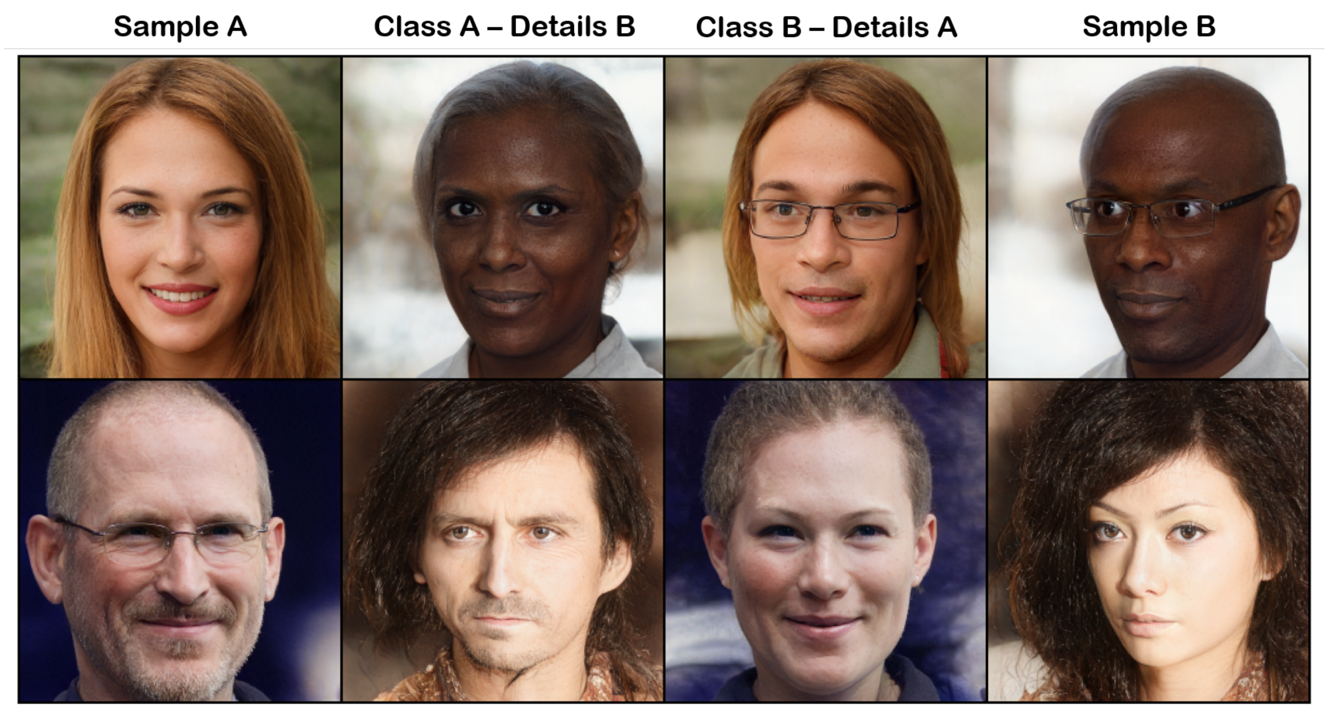

Figure 9. Mixing samples of different classes following what has been learned by Bayesian Optimization on the Celeb-A HQ Gender task. From left to right: Sample A (female or male), Class features of sample A mixed with Details of sample B, Class features of sample B mixed with Details of sample A, Sample B (male or female).

at: https://www.kaggle.com/datasets/jessicali9530/stanford-cars-dataset. We use the provided class names to filter those containing the words "coupe", "hatchback", "suv" or "van" in the original label. Then, we regroup them into 4 classes and manually remove outliers, keeping the same number of images per class. Lastly, each image is squared, adding black padding at the top and bottom sides, and resized at 128×128 resolution.

Training of classifiers. We use `pytorch.models` classes for the base architecture of each classifier. These are: `resnet50`, `vgg11_bn`, and `resnext50_32x4d`, respectively. In all cases, we replace the final fully connected layer with an MLP composed of two linear layers, a batch normalization, and a ReLU activation function. Then, we train the full model (backbone + head) on each task.

Hyperparameters of attacks. In a preliminary phase, we tested several hyperparameter configurations for both attacks, ensuring a sufficient number of steps for convergence. In general, we observe better performance of DeepFool, requiring significantly fewer steps to achieve high success rates. We show the final values for each task in Tables 2 and 3.

Task	Class Tested	Overshoot	Steps
Gender	2	0.01	1024
Identities	8	0.02	128
Cars	4	0.02	256

Table 2. Final hyperparameters for DeepFool attack.

Task	Reg. Const.	Conf.	Steps	Restarts	LR
Gender	64	0.01	1024	8	$1e-3$
Identities	16	0.05	1024	8	$5e-3$
Cars	24	0.02	1024	8	$2e-3$

Table 3. Final hyperparameters for Carlini & Wagner attack.

Code and Training of competitors. We use the official code provided by each competitor to train models on our tasks. The code is available at https://github.com/yaodongyu/TRADES for TRADES, at https://github.com/nercms-mmap/A-VAE for A-VAE , and at https://github.com/shayan223/ND-VAE for ND-VAE. For TRADES, we fine-tune each classifier for 50 additional epochs, setting the beta regularization term to $1.5, 1.0, 8.0$ for gender, identities, and cars, respectively. We found these values to give the best trade-off between robustness and ac-

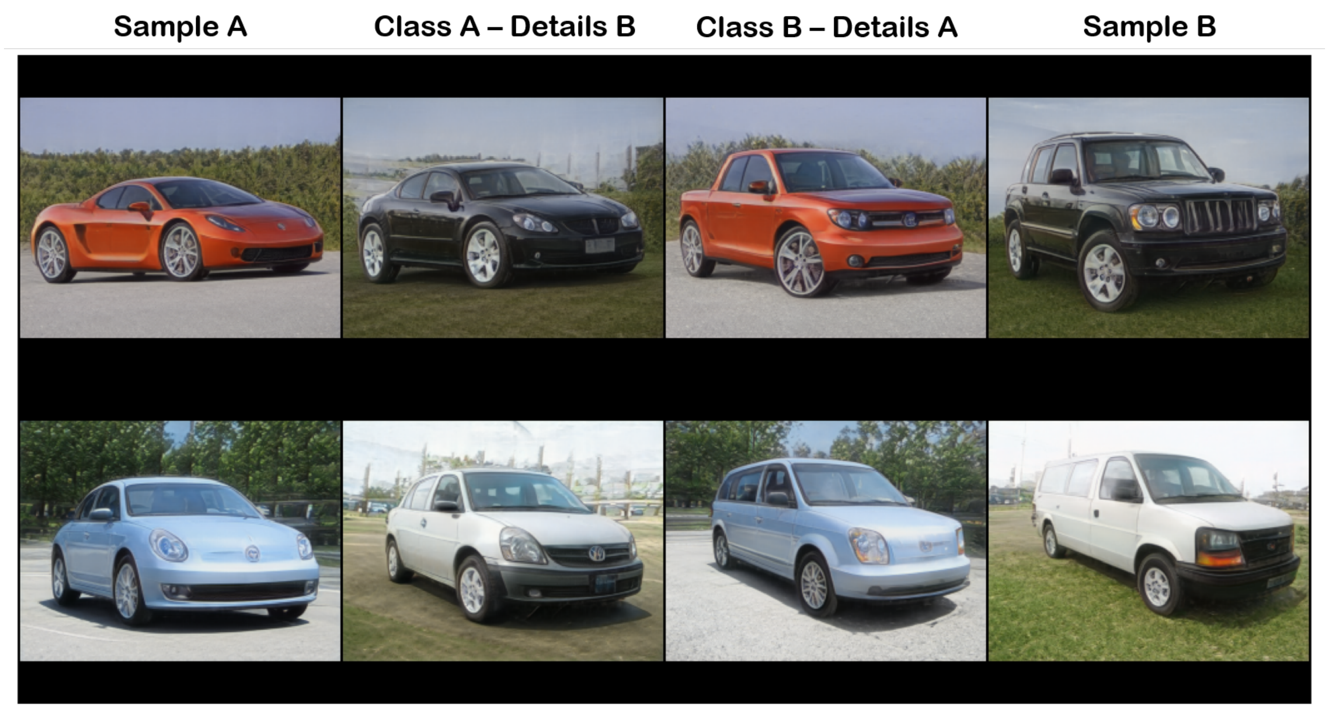

Figure 10. Mixing samples of different classes following what has been learned by Bayesian Optimization on the Stanford Cars Type Classification task. From left to right: Sample A (coupe or hatchback), Class features of sample A mixed with Details of sample B, Class features of sample B mixed with Details of sample A, Sample B (suv or van).

curacy. For A-VAE, we follow the indications of the original paper and downsample the initial image to 32×32, independently from the starting resolution. We continue training until convergence of the GAN model. For ND-VAE, we generate the adversarial training set with FGSM attack. The autoencoder model has 2 scales with 2 groups for the gender and cars tasks, and 1 scale only in the identity case, due to lower starting resolution. We train each model for 100 epochs.

D. Ablation Studies

In Figures 6 to 8 (c) of the main paper we test what type of α combination (**learned**, **linear** or **cosine**) performs best on each task. Here, we test the effect of preprocessing operations (adding gaussian noise or applying gaussian blur) when applied on the best configuration. For Gaussian noise, we add a random perturbation ν to each image, where $|\nu|_2 = 4$ in the gender and cars tasks, while $|\nu|_2 = 2$ on the identities task. For Gaussian blur we set the kernel size to $2^{\frac{R}{2}} - 1$, where R is the image resolution, and keep $\sigma = 1$ everywhere. Results for each task are visible in Figures 11 to 13. For better comparison, the attack success rate is computed also using Gaussian Noise or Gaussian Blur as a standalone purification mechanism. In the gender case, the combination with Blur is beneficial to the method, while adding Gaussian noise performs worse. On the remaininig tasks, the addition of Gaussian Noise allows an extra boost on the Carlini & Wagner attack, while all runs achieve similar performances on DeepFool. For computing the final results, shown in Figures 6 to 8 of the main paper, we use the best α + preprocessing combination found after these ablation studies.

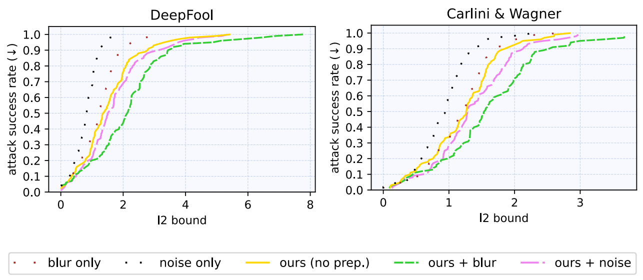

Figure 11. Attack succes rate (the lower the better) for increasing L_2 bounds for different attacks and preprocessing operations, on the Celeb-A HQ gender task. Dotted lines show the lower bound of applying only blur or noise as a defense mechanism.

E. Qualitative Results

We show one success and one failure case of purification on each task in Figures 14 to 16, respectively. For additional

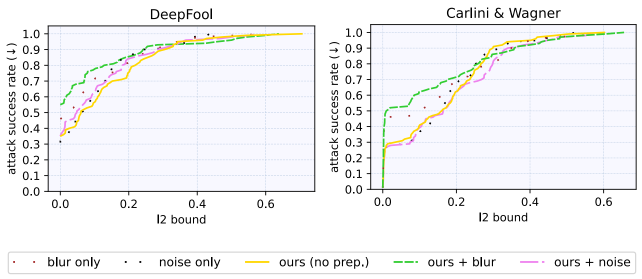

Figure 12. Attack succes rate (the lower the better) for increasing L_2 bounds for different attacks and preprocessing operations, on the Celeb-A 64 identities task. Dotted lines show the lower bound of applying only blur or noise as a defense mechanism.

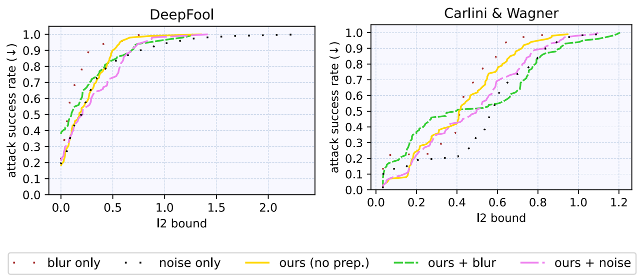

Figure 13. Attack succes rate (the lower the better) for increasing L_2 bounds for different attacks and preprocessing operations, on the Stanford Cars 128 task. Dotted lines show the lower bound of applying only blur or noise as a defense mechanism.

comparison, the images also show the same examples when purified with ND-VAE and A-VAE methods. As visible, the former mainly acts as a denoiser, attempting to remove adversarial noise while reconstructing the clean sample. A-VAE tries instead to modify the details of the input sample, while maintaining the semantic content. However, the mechanism appears limited, and the final purified image highly resambles the initial image. On the opposite, our method based on MLVGMs aims at maintaining only the relevan class information, while re-sampling all remaining irrelevant details. In coarse-grained classification (like male/female, with only two classes), this means the the final purified samples are highly different from the initial image. Conversely, in fine-grained tasks such as identity classification, the combination of α hyperparameters is properly tuned to alter only a few details, since it is easier to alter the class label. With this approach, MLVGMs can be effective both in coarse and fine-grained classification tasks, while limiting the freedom of the attacker, which is forced to act on the initial image by changing only imperceptible details.

F. Additional Results on Autoattack

In completion of the results reported in Section 4, we analyze the performance of our method also with Autoattack [14], a well-known evaluation pipeline which attempts to fool defense mechanisms by combining different techniques. The results for the three tasks are shown in Figures 17 to 19. In general, changing the hyperparameters combination (**learned**, **linear** or **cosine**) does not seem to have a significant effect in this scenario, while adding gaussian blur or noise is beneficial in Celeb-A HQ and Standford Cars tasks, respectively. When comparing with other methods, the general trend observed in Section 4 is maintained, with ND-VAE performing best on ids and cars classification and our MLVGMs-based mechanism achieving comparable results with other methods. Interestingly, TRADES [75] fails against this attack in the Celeb-A HQ gender classification task, showing the effectiveness of AutoAttack even when compared to state-of-the-art methods such as C&W or DeepFool.

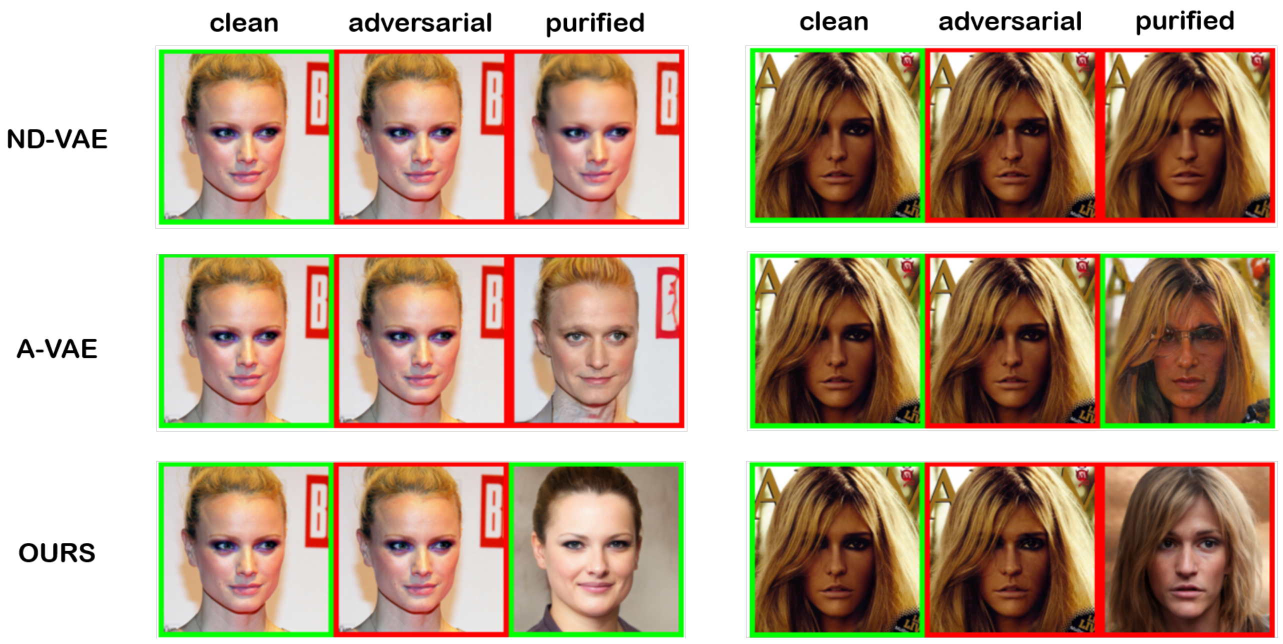

Figure 14. Qualitative examples of purification via MLVGMs on the Celeb-A HQ gender task. We show one success case (left) and failure case (Right), comparing the purification with the one of ND-VAE and A-VAE. In each example, we additionaly show the clean and adversarial images.

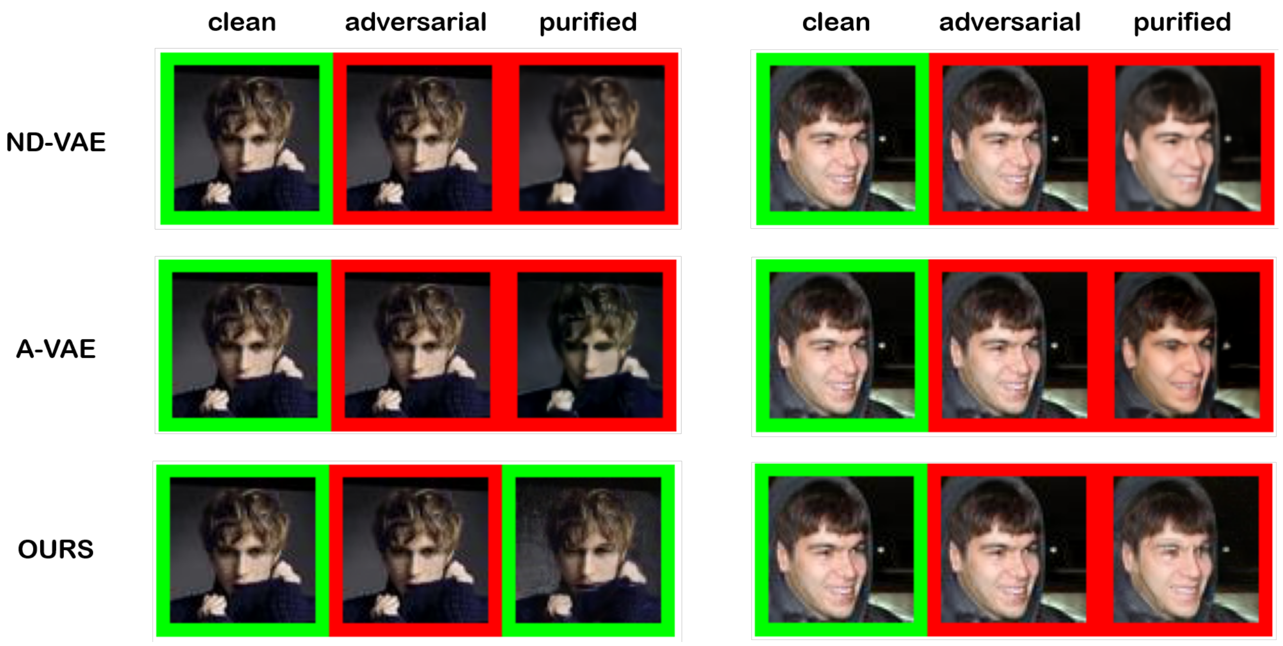

Figure 15. Qualitative examples of purification via MLVGMs on the Celeb-A 64 identities task. We show one success case (left) and failure case (Right), comparing the purification with the one of ND-VAE and A-VAE. In each example, we additionaly show the clean and adversarial images.

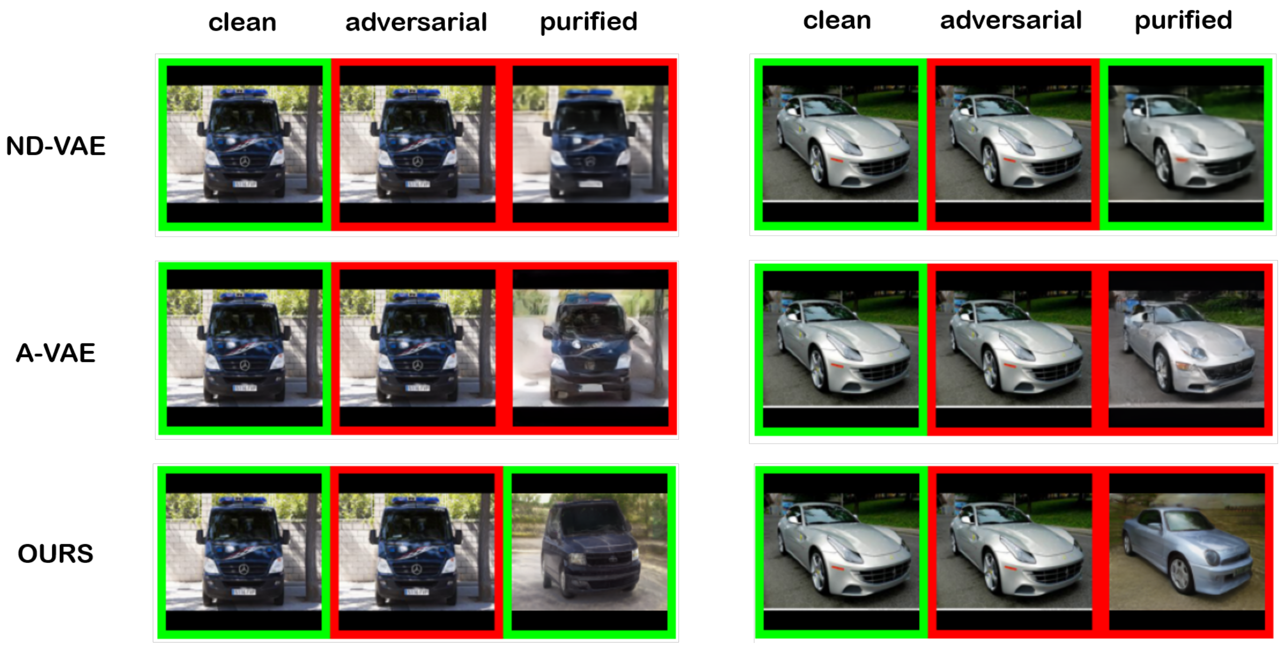

Figure 16. Qualitative examples of purification via MLVGMs on the Stanford Cars 128 task. We show one success case (left) and failure case (Right), comparing the purification with the one of ND-VAE and A-VAE. In each example, we additionaly show the clean and adversarial images.

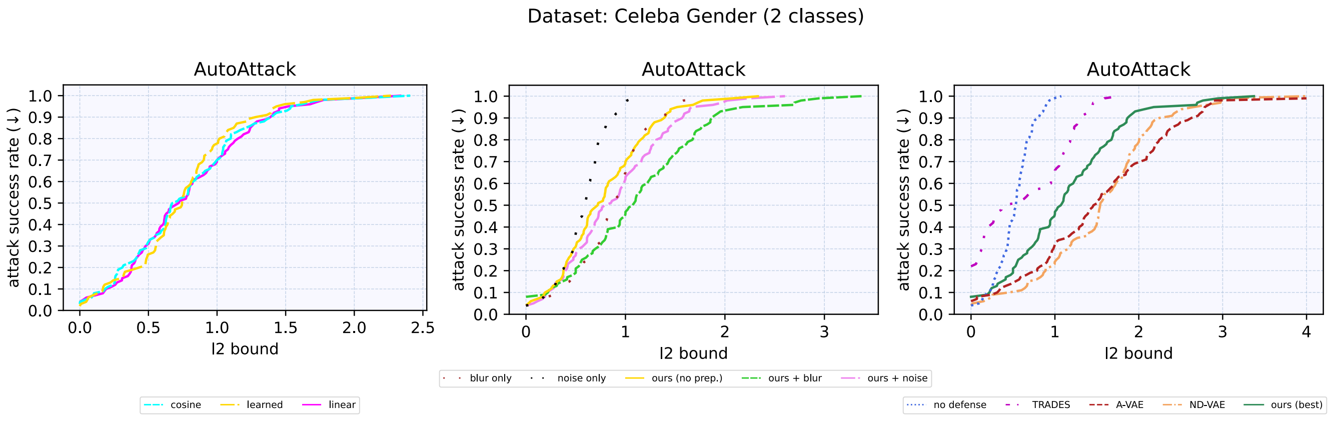

Figure 17. Success rates of Autoattack [14] for 100 samples on the Celeb-A HQ gender classification task. From left to right: comparison of the tested combinations (**learned**, **linear** and **cosine**); ablations on the introduction of random noise and blur; and comparison with other methods.

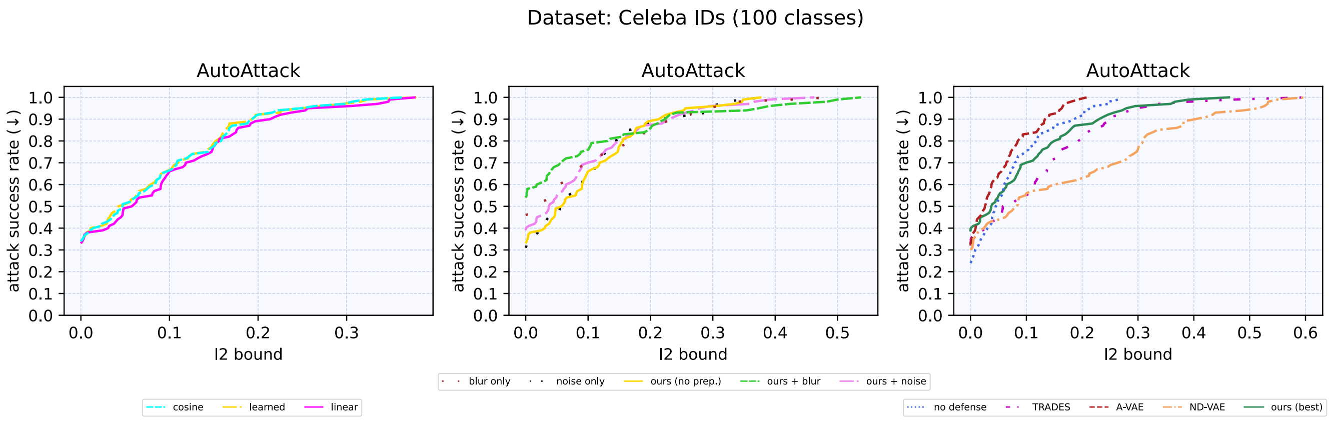

Figure 18. Success rates of Autoattack [14] for 100 samples on the Celeb-A ids classification task. From left to right: comparison of the tested combinations (**learned**, **linear** and **cosine**); ablations on the introduction of random noise and blur; and comparison with other methods.

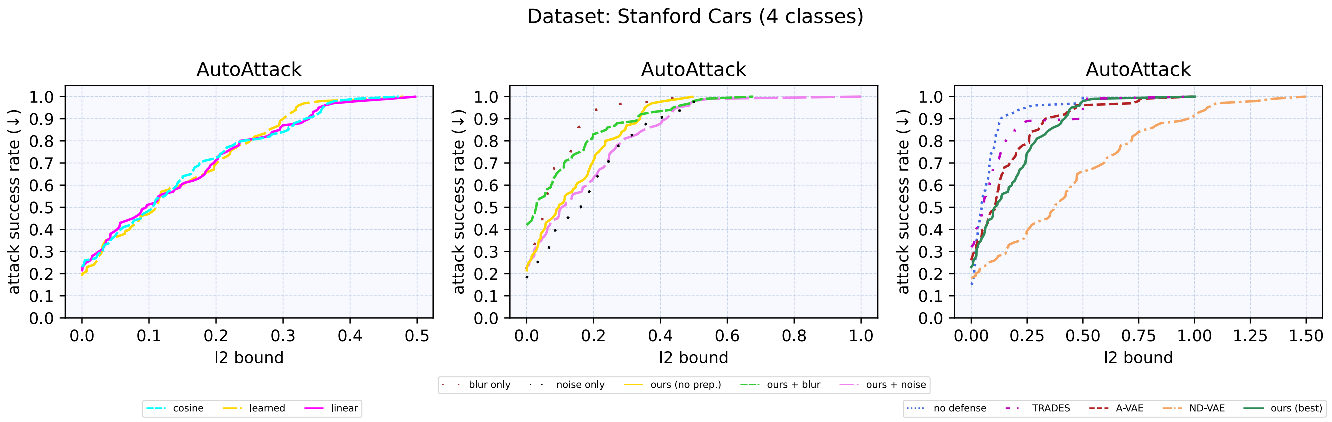

Figure 19. Success rates of Autoattack [14] for 100 samples on the Stanford Cars classification task. From left to right: comparison of the tested combinations (**learned**, **linear** and **cosine**); ablations on the introduction of random noise and blur; and comparison with other methods.