Ensembling with Deep Generative Views
Supplementary Material

Lucy Chai[12] Jun-Yan Zhu[23] Eli Shechtman[2] Phillip Isola[1] Richard Zhang[2]
[1]MIT [2]Adobe Research [3]CMU
{lrchai, phillipi}@mit.edu {elishe, rizhang}@adobe.com junyanz@cs.cmu.edu

In supplementary material, we provide additional details on dataset preparation and classifier training methods for each classification task. We show additional qualitative examples of the GAN reconstructions and the perturbation methods investigated in the main text, at both fine and coarse layers of the latent code. Finally, we provide additional results investigating different experiment settings and classifier training distributions under each type of latent perturbation method.

1. Supplementary Methods

1.1. Pretrained generators

Unconditional GANs learn to mimic the image manifold by transforming low dimensional latent codes to image outputs. A number of interesting properties emerge in these generator networks, such as learning to model degrees of variation in real data. We use pretrained StyleGAN2 generators [8] for our experiments. As class labels for images are not required during GAN training, the generators are trained on larger datasets than we would otherwise use for classification – 5,520,756 images for LSUN Cars [15], 1,657,266 images for LSUN Cats [15], and 70,000 for FFHQ faces [7] for the 512×384 resolution car, 256×256 resolution cat, and 1024×1024 resolution face generators respectively.

1.2. Datasets

Face attribute classification We used the labeled CelebA-HQ [12, 5] dataset containing 30000 faces with 40 labelled attributes. All face images are aligned by facial landmarks and cropped to square at 1024×1024 resolution. We follow the training, validation, and test splits used in [12] on the 30,000 HQ images to obtain 24,183 images for training, 2993 for validation, and 2824 for testing. Due to the alignment and square cropping, we do not perform any additional resizing or shifting operations prior to projection into the GAN, i.e., the GAN reconstructs the full input image.

Car Shape Classification We derive our cars dataset from [10], which in total contains 16,185 car images at the granularity of Make, Model, and Year for each image. As the GAN cannot recover fine-grained details for each image, instead we take a subset of the labelled images and group them into super-classes of "SUV", "Sedan", and "Cab" by parsing the provided class name. Using this subset of three super-classes, we divide the images into 2007 images for training, 1007 for validation, and 1049 for testing; by splitting each fine-level class according to a 50%/25%/25% ratio. Prior to projection into the GAN's latent code, we first rescale the width of each image to 512px, shift the image to center the car using the provided bounding box, and then perform a center crop on the shifted image to fit the GAN's aspect ratio (512×384 pixels). As the shifting operation may introduce unknown pixels around the edges of the image, the encoder and optimization step are both performed with a masking input to account for these missing pixels [1]. Note that due to the GAN's aspect ratio and the aspect ratio of cars, there are parts of the image that may be cut out; therefore at test time, we find that adding multiple random image crops to ensemble improves classification.

Cat breeds classification The pets dataset from [13] contains in total 37 breeds of cats and dogs, including 12 cat breeds with 200 images per class. We subdivide the 200 images of each class into 100 images for training, 50 for validation, and 50 for testing; this yields a total of 1200 images for training, 600 for validation and 600 for testing. To preprocess the dataset, we align each image using face attributes: we apply a face landmark detector on the images[1], align the landmarks to a canonical pose, and crop to 256px; empirically we find that this improves both classification performance and GAN reconstruction. Note that in a few cases, the cat face is not correctly detected, resulting in a poorly aligned image; even though these instances will negatively impact classification, we do not remove them but rather retain the full dataset. Similar to before, for the GAN

[1] https://github.com/zylamarek/frederic

reconstruction process, the encoder and optimization steps both use a masking input to account for missing pixels that may occur during the alignment operation.

1.3. Classifier Training

Binary facial attribute classification For classification of binary face attributes we follow the setup of Karras et al. [8], which uses the GAN discriminator architecture as the model for the attribute classification task. We also follow the corresponding downsampling step which performs a 4×4 average pooling operation prior to classification. We train a classifier for each attribute from scratch (we also experimented with finetuning classifiers, and obtained similar results). Random horizontal flipping is applied during training. We use the Adam optimizer [9] with default parameters (learning rate 10^{-3}, $\beta_1 = 0.9$), and train until validation accuracy does not increase for five epochs (most attributes will finish training by 20 epochs). We use the same setup when training from the image dataset as training from the GAN-generated reconstructions: depending on the setting, we simply replace the image x with the reconstructed image $G(w^*)$, or the perturbed latent $G(\tilde{w})$ before sending to the classifier for training. We use the checkpoint with the highest validation accuracy for further experiments.

Multi-class classification For the cat and car classification tasks, we use a ResNet-18 [4] backbone with ImageNet [14] pretrained weights for the feature extractor. We modify the final linear layer to output the appropriate number of logits for each class – three classes for car classification, and 12 classes for cat breeds. First, we finetune this model on the respective datasets. We use the Adam optimizer [9] with initial learning rate 10^{-4} for the feature backbone and 10^{-3} for the linear classification layer and $\beta_1 = 0.9$. We then decay learning rate by 10× if the validation accuracy does not increase for 10 epochs, up to a minimum learning rate of 10^{-6}. We use a maximum of 500 epochs for training, and record the checkpoint with the highest validation accuracy for further experiments. Next, to finetune a model on GAN-generated samples, we start with this previous model finetuned on the appropriate dataset and train with a reduced learning rate 10^{-6}; with 50% probability for each batch, the model is finetuned on the GAN reconstructions or real image samples. During the classifier training procedure we apply a random resized crop with scale=[0.8, 1.0] and random horizontal flipping. All images are cropped to square prior to classification, at a resolution of 256×256 for cars and 224×224 for cats.

1.4. Perturbations in GAN Latent Code

For the isotropic and PCA direction perturbation methods, an additional hyperparameter is the extent of perturbation allowed. For the isotropic perturbation, we scale the variance σ of the added noise to ensure that the modified latent does not deviate too far from the starting latent. For the PCA directions, we randomly sample a multiplier $\beta \sim \mathcal{U}[-\sigma, \sigma]$ that we use to scale the selected principle component direction. We try a few values for each hyperparameter, selected so that the GAN-modified outputs are similar to the input but with small distortions; we take the best setting from validation data to apply on the test partition. Values for these hyperparameters are listed in Table 1.

Table 1: Hyperparameter values for isotropic and PCA latent perturbation methods. For the isotropic perturbation, we use a σ hyperparameter to scale the variance of the random noise added to each optimized latent code. For the PCA perturbation method, we use σ to denote a maximum magnitude for each principle component applied, and sample a magnitude β randomly from $\beta \sim \mathcal{U}[-\sigma, \sigma]$. For each model, we select a few discrete values for each hyperparameter to create small variations on the input image without deviating too far from the input. We select the best hyperparameter setting on validation data and use the same value at test time.

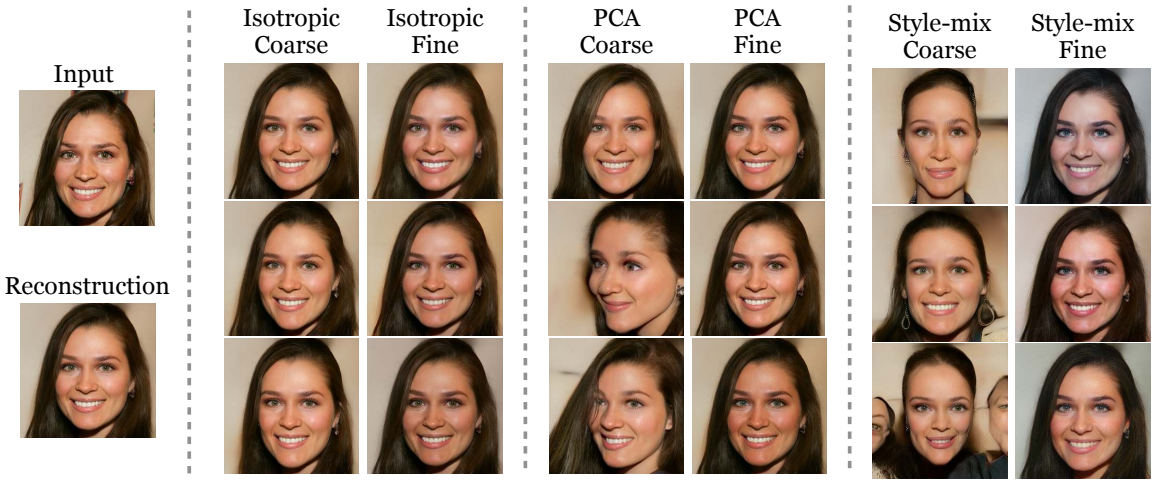
Model	Isotropic Coarse	Isotropic Fine	PCA Coarse	PCA Fine
Car	{1.0, 1.5, 2.0}	{0.3, 0.5, 0.7}	{1.0, 2.0, 3.0}	{ 1.0, 2.0, 3.0}
Cat	{0.5, 0.7, 1.0}	{0.1, 0.2, 0.3}	{0.5, 0.7, 1.0}	{0.5, 0.7, 1.0}
Face	{0.1, 0.2, 0.3}	{0.1, 0.2, 0.3}	{1.0, 2.0, 3.0}	{1.0, 2.0, 3.0}

2. Supplementary Results

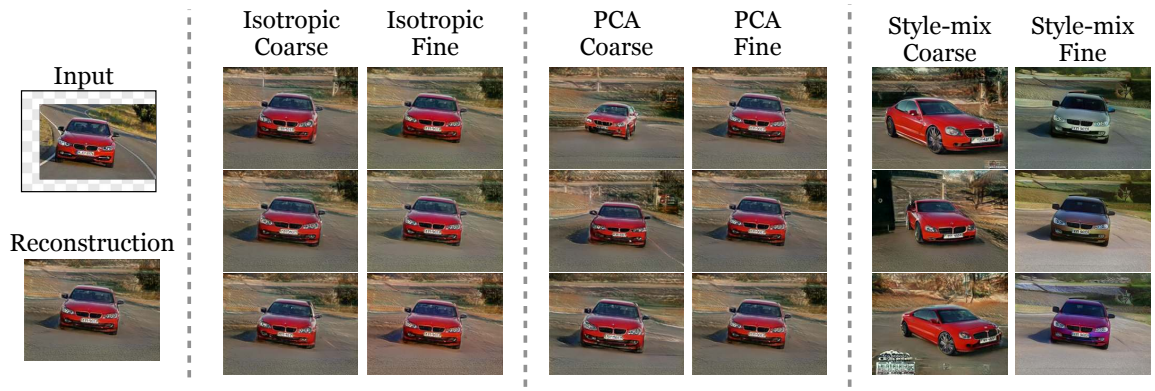
2.1. Additional Qualitative Examples

In the main text, we define three methods for perturbations in the latent code of a GAN: 1) adding isotropic Gaussian noise, 2) moving along principle component axes [2], and 3) style-mixing the optimized latent code with a random latent code. We apply each type of perturbation respectively to the coarse layers (first four style layers) or fine layers (tenth and higher style layers) of the optimized latent code w. In Fig. 1, we show qualitative examples of each type of perturbation applied to the same base image in each domain. Note that the coarse layers correspond to positional or shape changes, while the fine layers correspond to coloring changes. Furthermore, the style-mixing operation, which swaps in an entirely *random* latent code rather than adding some offset to the optimized latent, achieves qualitatively larger changes than the isotropic or PCA methods.

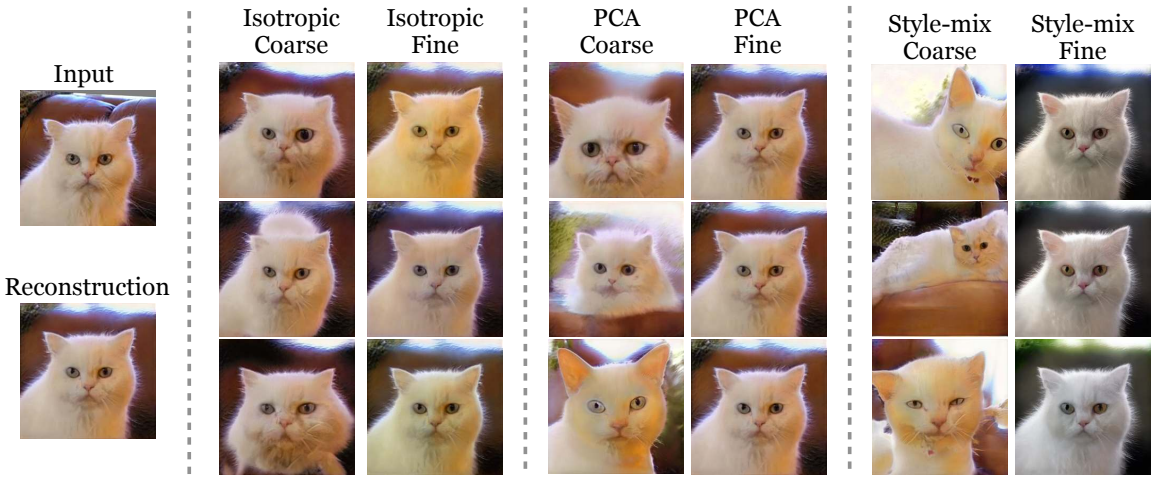
Reconstruction via GAN inversion is easier for images in canonical poses and plain backgrounds that do not contain uncommonly seen details. Fig. 2 visualizes the four best reconstructed and the four worst reconstruction images in the test split of each dataset, measured using the LPIPS perceptual distance metric [16]. The hardest images to reconstruct contain text, large accessories on the head, or non-facial ob-



(a) Face domain



(b) Car domain



(c) Cat domain

Figure 1: Visualizing GAN perturbations. For the (a) Face, (b) Car, and (c) Cat domains, we show qualitative samples of an input image (Input), which is centered, if necessary, prior to reconstruction by the GAN (Reconstruction). Once the latent code is optimized to obtained the best reconstruction of the input, we perform three types of latent code modifications: isotropic, PCA, and style-mixing perturbations at both coarse and fine layers of the latent code, where coarse-level manipulations alter pose and size, while fine-level manipulations alter color.

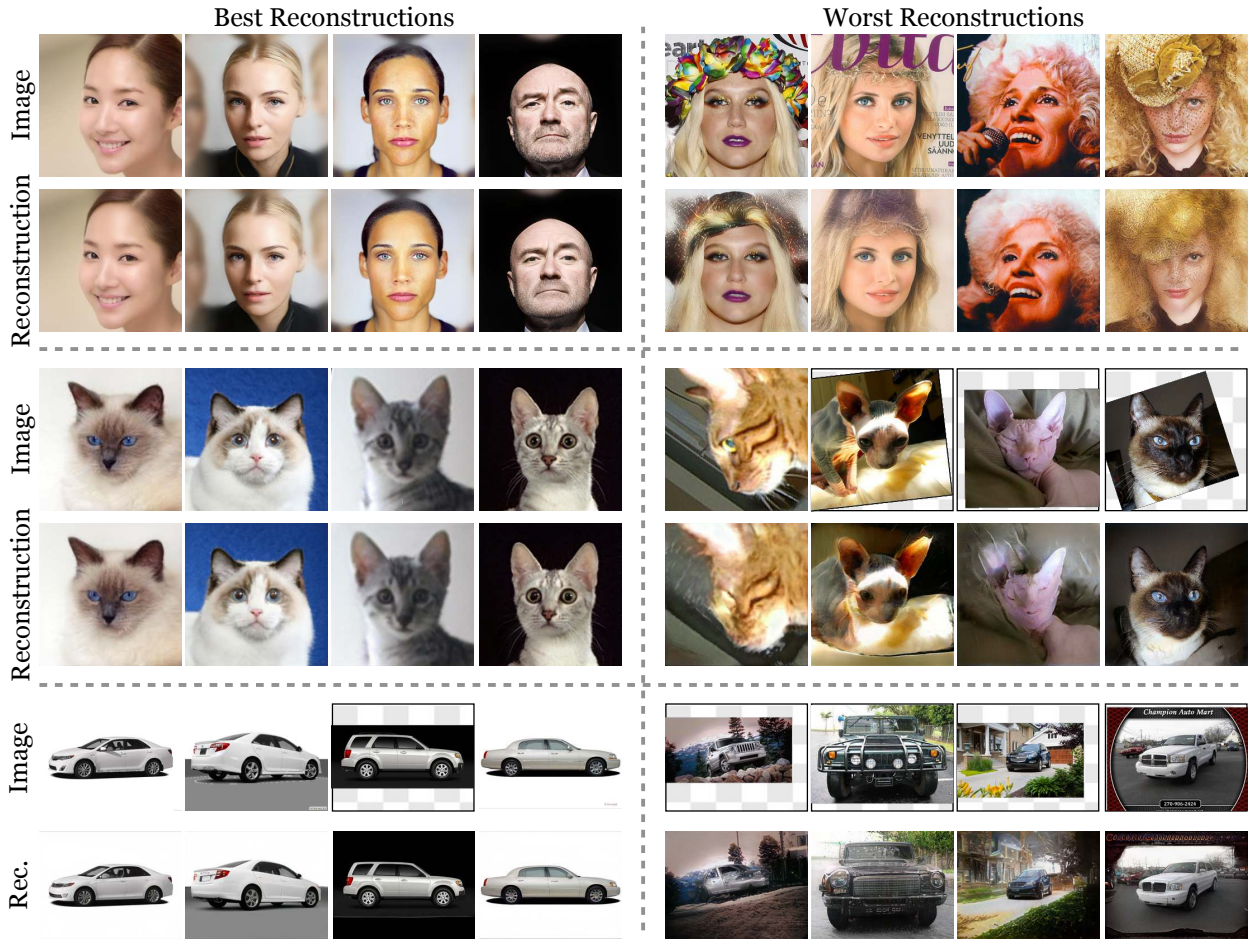


Figure 2: Visualizing the four best and worst reconstructions in the test split measured using LPIPS perceptual distance [16]. On each domain, The best reconstructions tend to be in canonical poses with simple backgrounds, and the worst reconstructions have complex backgrounds or textural details that the GAN cannot accurately recreate.

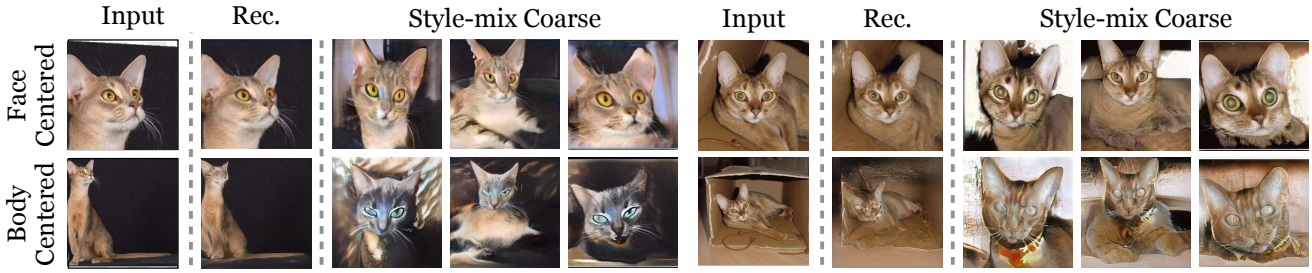


Figure 3: The StyleGAN2 generator exhibits a bias towards cat faces, thus we find that projecting face-centered images using the GAN yields better reconstructions than body-centered images. Furthermore, when the style-mixing operation is performed, the face-centered images better preserve the identity of the cat on the modified images.

jects for the face domain. On the cat and car domains, the difficult cases are detailed textures, complex backgrounds, or unusual poses.

On the cat dataset, we preprocess all images by aligning and cropping the face, as we find that the GAN has a facial bias in reconstruction. We show examples in Fig. 3. Centering the same image on the face, rather than the body (we use a MaskRCNN object detector [3] to obtain a bounding

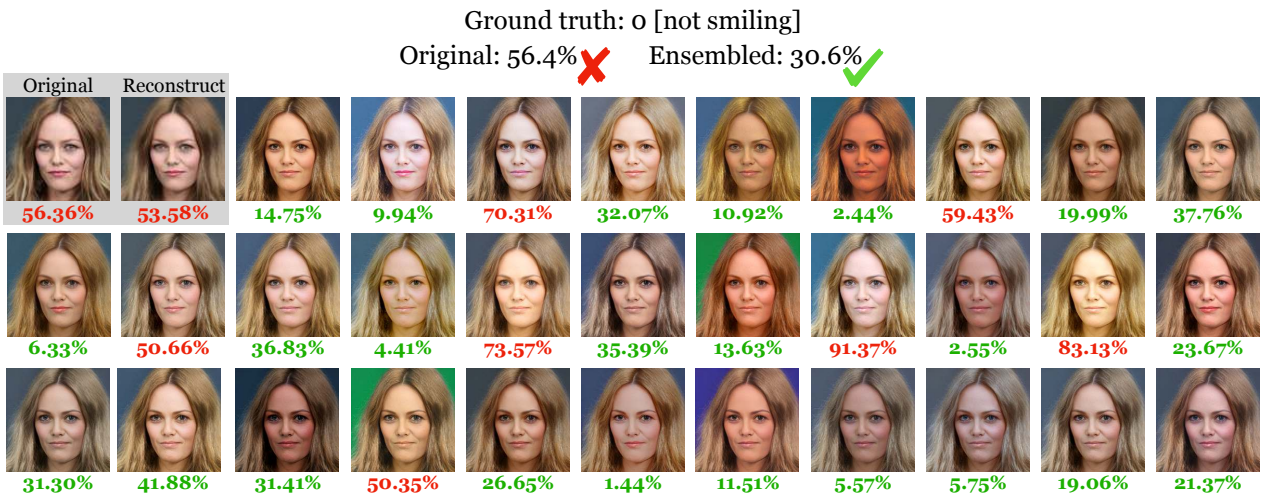


Figure 4: A selected example in which the original image is predicted incorrectly ($P(\text{Smiling}) > 0.5$), but ensembling the classifier predictions with style-mixing on fine layers recovers the ground-truth label ($P(\text{Smiling}) < 0.5$).

box for the cat), improves the GAN's reconstruction. Furthermore, we find that style-mixing in coarse layers better preserves the identity of the cat when it is face-centered, as opposed to body-centered.

In Fig. 4 we show an example of the fine layer style-mixing augmentation in latent space where the original image is misclassified but the ensemble recovers the correct prediction. The classifier is sensitive to the variations introduced by the style-mixing operation, which changes the classifier's incorrect prediction of the original image to the correct one when averaging the predictions on the GAN-generated views.

2.2. Additional Experiments: CelebA-HQ

Distribution of classification accuracies The CelebA-HQ dataset [12] contains attribute labels for 40 binary classification tasks spanning a wide range of difficulty. The hardest attribute to classify is "Big Lips" with a test of 53.58%, while 23 out of 40 attributes can attain a test accuracy of over 90%. We show a distribution of the test accuracy over all 40 attributes in Fig. 5.

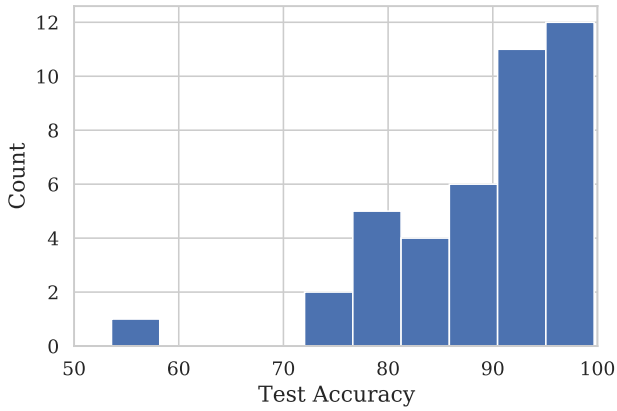
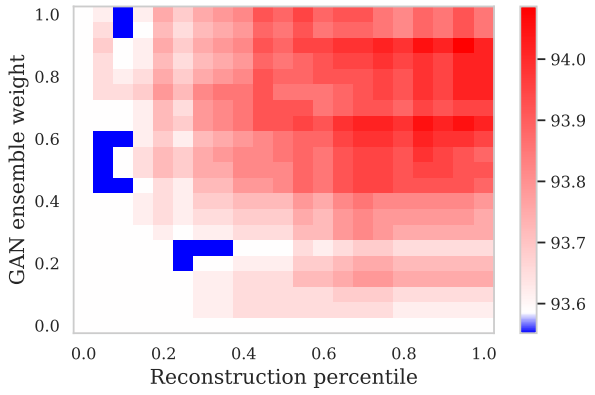


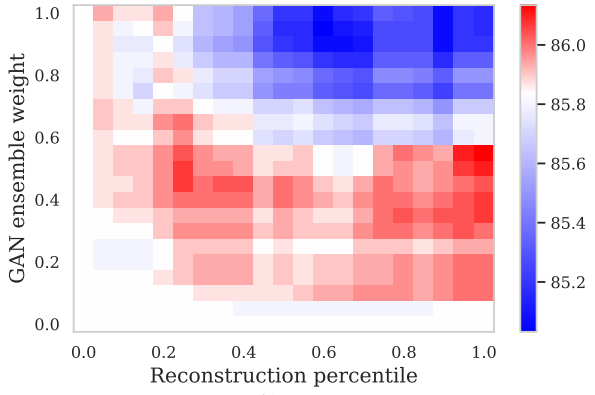
Figure 5: Distribution of test accuracies for the 40 attribute classification tasks in the CelebA-HQ dataset. Most attributes (23 out of 40) attain over 90% accuracy on the test partition, while the Big Lips attributes has the test lowest accuracy at 53.58%.

Variations on ensemble weighting In the main text, we introduce a weighting hyper-parameter α which balances between using the original image for classification and the GAN-generated variants. However, not all dataset images can be reconstructed with the same fidelity. We investigate an alternative ensembling approach, in which we also discard the GAN-generated variants whose reconstruction error is greater than a certain percentile cutoff; this 2D space is visualized in Fig. 6. In the main paper, we retain the GAN-generated variants on all images and only use the ensemble weighting α, which corresponds to a search over the right-most column in the 2D plot. However, we find that using the 2D search over ensemble weighting and reconstruction on the validation split performs similarly to the simpler 1D hyper-parameter search.

Optimization time vs. accuracy For the classifier to behave similarly on the GAN reconstruction and a given real image, we desire the GAN's version to be as similar to the original image as possible. However, obtaining a close reconstruction via optimization is slow; we must balance between reconstruction quality and a computationally feasible optimization budget over the dataset. As such we use an encoder model to initialize the starting latent code, and then optimize for 500 steps to improve the reconstruction. In



(a) Smiling



(b) Young

Figure 6: Visualizing classification accuracy as a function of ensemble weight and reconstruction similarity. In the main text, we cross validate for an ensembling weighting parameter α between the original test image, and it's GAN-reconstructed variants. Here, we also explore reconstruction quality as an additional axis: we discard the GAN-reconstructed ensemble if the reconstruction similarity of the image is below a certain cutoff (0 corresponds to using no reconstructed images, and 1 indicates using the GAN ensemble for all images; the experiments in the main text correspond to the right-most column of each grid.) White corresponds to standard classification accuracy, shades of red indicate increases in accuracy, and shades of blue indicate decreases in accuracy. We find that the classification is more sensitive to the ensemble weighting α rather than reconstruction similarity, and using the 2D grid search performs similarly to the simpler 1D search over ensemble weight α.

Tab. 2, we compare the L1 and LPIPS reconstruction distance and classification accuracy of the reconstructed images as a function of the number of optimization steps. While the reconstruction improves as more optimization is performed, the accuracy plateaus after 250 optimization steps, suggesting that a reduction in optimization time can possible while obtaining similar classification results.

Ensembled classification accuracy: 40 attributes When training the classifier, the standard approach is to train on the image dataset – in this case, if GAN-generated images are added as part of the ensemble at test time, there is potentially a domain gap as the classifier has never seen GAN images during training. However, the face domain is fairly simple for the generator to reconstruct; adding fine layer style-mixing of images at test time, even without a classifier trained on GAN images, outperforms the baseline of testing on a single image when averaged over 40 attributes. Adding additional color and spatial jitter to the image at test time offers a further boost. Using this setting, we sort the attributes based on how much ensembling at test time helps in Table 3; the highest difference between ensembled test accuracy and standard test accuracy (classifying a single image) indicates the attribute where ensembling increases accuracy the most, while the lowest difference indicates where ensembling helps the least and can harm classification. There are a few attributes that notably do not benefit from ensembling: some accessories (Wearing Hat, Wearing Necklace) which can be difficult for the GAN to reconstruct, while several color-based attributes (Black Hair, Brown Hair, Gray Hair) also do not benefit from the fine layer style-mixing operation, which can cause color changes.

Table 2: Reconstruction similarity (L1, LPIPS) and accuracy for the 'Smiling' attribute, vs. optimization steps. Classification accuracy on images is 93.6%.

Opt. Steps	0	50	100	150	200	250	500
L1	0.172	0.104	0.092	0.086	0.083	0.080	**0.073**
LPIPS	0.443	0.252	0.219	0.201	0.188	0.179	**0.152**
Acc	92.2	93.2	93.2	93.2	93.5	**93.6**	93.4

Alternative projection algorithms Rather than using the 1024px pre-trained StyleGAN2 [8], we also run the same experiments using a 256px StyleGAN and the In-domain inversion algorithm [17]. This method is designed for fast image projection, and combined with a smaller resolution GAN, *reduces the computational cost of* inverting real images and creating the GAN-generated variations. Over the 40 facial attributes, the results are correlated (Fig. 7). On average over 40 attributes, using the optimization procedure in the main text achieves 0.07 accuracy gain using fine layer style-mix augmentation only at test time, and 0.13 gain using fine style-mix combined with image augmentations at test time; with the In-domain inverter, the average accuracy gain is 0.10 for both fine layer style-mix augmentation only and fine style-mix combined with image augmentations at test time. Thus, we are able to obtain a similar result, but with a lower computational overhead.

Training distributions Although we find that there are improvements when using GAN-generated views at test time, even when the classifier is only trained on real face

Table 3: Comparison of standard test accuracy and ensembled test accuracy on classifiers trained on images, using the combined GAN augmentation at test time on 40 facial classification attributes. This augmentation consists of style-mixing at fine layers, and small color and spatial jittering. Attributes are sorted in order from highest difference (ensembling helps the most), to lowest difference (ensembling harms classification).

Attribute	Standard Test Accuracy	Ensembled Test Accuracy	Difference	Attribute	Standard Test Accuracy	Ensembled Test Accuracy	Difference
1: Wearing Lipstick	93.17	94.47	1.30	21: Wearing Earrings	85.20	85.21	0.02
2: Wavy Hair	74.29	75.26	0.96	22: Bald	98.26	98.28	0.01
3: High Cheekbones	85.48	86.40	0.92	23: Mustache	95.89	95.90	0.00
4: No Beard	94.83	95.47	0.64	24: Blurry	99.68	99.68	0.00
5: Goatee	95.82	96.23	0.41	25: Bushy Eyebrows	91.50	91.50	-0.00
6: Arched Eyebrows	81.59	81.95	0.36	26: Double Chin	94.48	94.47	-0.00
7: Male	97.31	97.65	0.34	27: Attractive	78.82	78.82	-0.01
8: Mouth Slightly Open	93.52	93.83	0.31	28: Chubby	94.37	94.36	-0.01
9: Smiling	93.59	93.89	0.30	29: Pale Skin	97.17	97.15	-0.01
10: Young	87.50	87.74	0.24	30: Rosy Cheeks	91.64	91.61	-0.03
11: Eyeglasses	99.29	99.50	0.21	31: Bangs	95.15	95.11	-0.04
12: Bags Under Eyes	81.98	82.15	0.17	32: Pointy Nose	72.24	72.18	-0.06
13: Sideburns	96.67	96.83	0.16	33: Gray Hair	98.37	98.28	-0.09
14: 5 o Clock Shadow	92.71	92.84	0.13	34: Receding Hairline	92.63	92.55	-0.09
15: Big Nose	76.95	77.08	0.13	35: Wearing Necklace	81.02	80.92	-0.10
16: Heavy Makeup	89.09	89.22	0.13	36: Brown Hair	86.61	86.49	-0.13
17: Blond Hair	94.30	94.37	0.07	37: Narrow Eyes	86.15	86.02	-0.13
18: Oval Face	79.32	79.36	0.04	38: Black Hair	89.41	89.14	-0.27
19: Wearing Necktie	95.47	95.50	0.04	39: Wearing Hat	98.58	98.29	-0.29
20: Straight Hair	80.38	80.42	0.04	40: Big Lips	53.58	53.05	-0.52

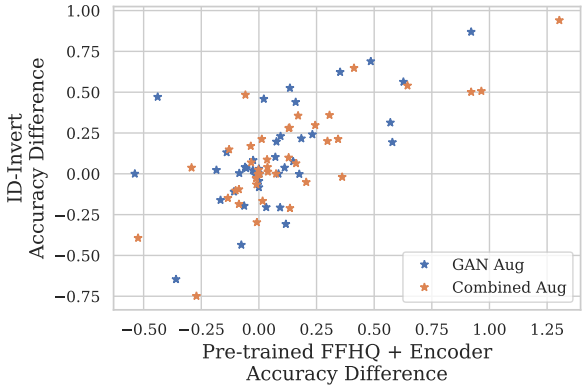


Figure 7: Comparison of StyleGAN2 [8] generator with a separately trained encoder, and the In-domain inversion method [17]. We plot the accuracy difference between using GAN-based ensembling at test time and standard test accuracy for both methods, where style-mixing augmentation is shown in blue, and combined style-mixing and image augmentations is shown in orange. Over the 40 attribute classification tasks, the accuracy gains of the two methods are similar (Pearson $r = 0.69$, $p < 0.001$), but the smaller resolution of the In-domain GAN allows for faster inversion and inference.

images, we also investigate different training variations for the face attribute classifiers (Fig. 8), such as training the classifier using GAN-reconstructed images and perturbed reconstructions. On average over 40 attributes, we find that adding the style-mixing ensemble is more beneficial when the classifier is also trained on the GAN reconstructions, compared to when these classifiers are trained only on images. The results of applying fine style-mixing while training these classifiers are mixed and less beneficial than training on the GAN reconstructions alone. This is likely due to the inability of fine-level changes to preserve the the classifier boundaries for certain attributes, such as those based on color. In Fig. 9, we show examples of two attributes where training with fine-level style-mixing improves classification (Wavy Hair and Young), and two attributes where such an adjustment in the latent code is harmful (Black Hair and Brown Hair). In the latter case, we find that training the classifier with coarse-level isotropic jittering outperforms training the classifier with fine-level style-mixing, as fine-level adjustments in color may create samples inconsistent with the original class label.

Additional attributes: ensemble size When ensembling on face attributes, we use an ensemble size of 32 images. In Fig. 10, we plot the classification accuracy as a function of the number of ensembled images on four attributes that benefit from test-time ensembling with GAN-generated views (Smiling, Arched Eyebrows, Wavy Hair, and Young). Generally, increasing the number of images in the ensemble improves performance up to a certain point, saturating around an ensemble size of 16 GAN samples.

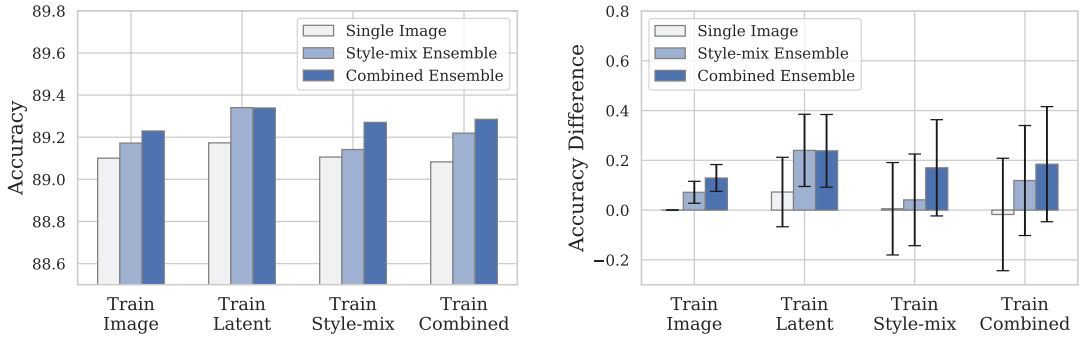


Figure 8: Classifier training variations, averaged over 40 facial attributes. (Left) We plot the average classification accuracy when the classifier is trained only on images (Train Image), trained on optimized latent codes corresponding to each image (Train Latent), trained with fine-layer style-mixing (Train Style-mix) and trained with style-mixing and a combination of color and spatial jitter (Train Combined). We evaluate using an ensemble of style-mixed samples, or the combined augmentation (different colored bars). (Right) As there is large variation in the classification accuracy of individual attributes (see Fig. 5), we also plot the difference in classification accuracy for each setting, compared to a classifier trained on images and evaluated on a single image. While training on the optimized latent codes outperforms training on images, we find that results on style-mixing during training are mixed, as some attributes are sensitive to the style-mixing operation (we show examples in Fig. 9). Error bars indicate standard error over all 40 attributes.

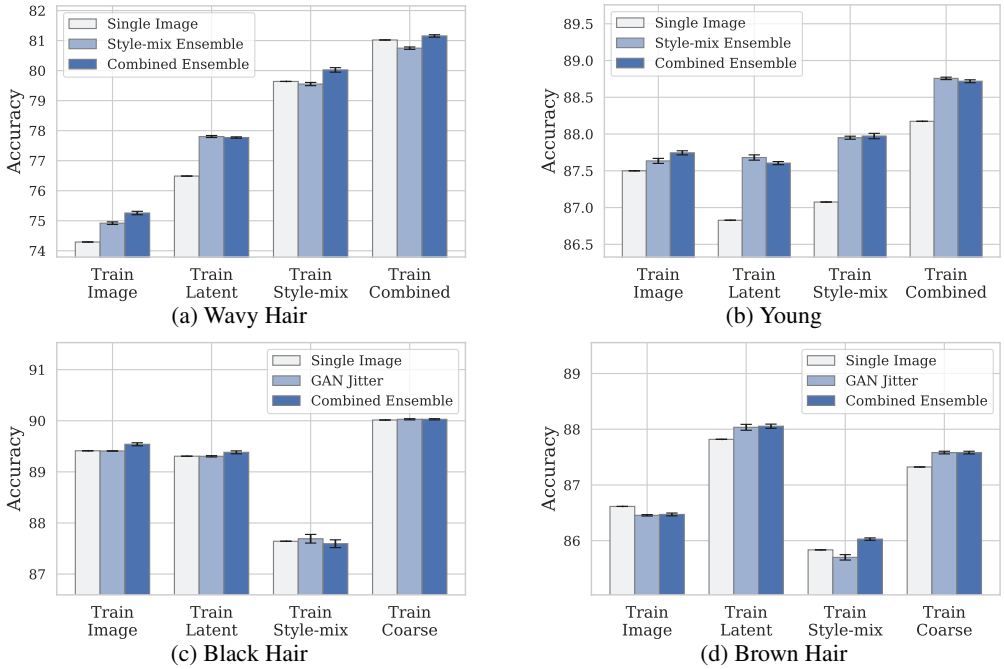


Figure 9: Classifier training variations on individual attributes. (Top) We show two attributes (Wavy Hair and Young) where training with the fine-level style-mixing outperforms training on the original images and the optimized latent code. (Bottom) However, style-mixing during training is harmful for some attributes, such as Black Hair and Brown Hair, where the color adjustment introduced by changing the latent code may create inconsistencies with the image label. For the Black Hair attribute, training with isotropic jittering in coarse layers performs best, while for Brown Hair, training on the latent codes, without additional GAN-based augmentations, is better. For the Black Hair and Brown Hair attributes, we plot test accuracy using coarse-level isotropic jittering as the type of GAN augmentation, which performs better than fine-level style-mixing, except in the case when the classifier is trained with the fine-layer style-mixing augmentation (Train Style-mix).

Additional attributes: image corruptions We show results on the same four attributes (Smiling, Arched Eyebrows, Wavy Hair, and Young), when the input image is corrupted prior to classification (Fig. 11). We project the *corrupted* image through the GAN to obtain the reconstructed image, and perform style-mixing on the fine layer to create

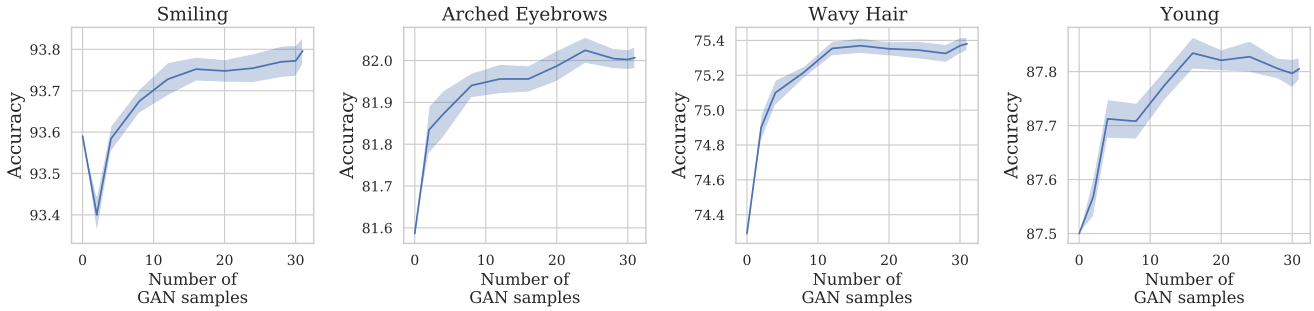


Figure 10: Effects of ensemble size. Classification accuracy as a function of the number of ensembled deep generative views, on four facial attributes that benefit from GAN-augmented views at test time. Zero views corresponds to using the original input image and adding more views increases accuracy up to a certain point. We use a total of 32 images (1 dataset image and 31 GAN views) in our experiments, as performance saturates. The shaded region corresponds to standard error over random draws of the ensemble elements.

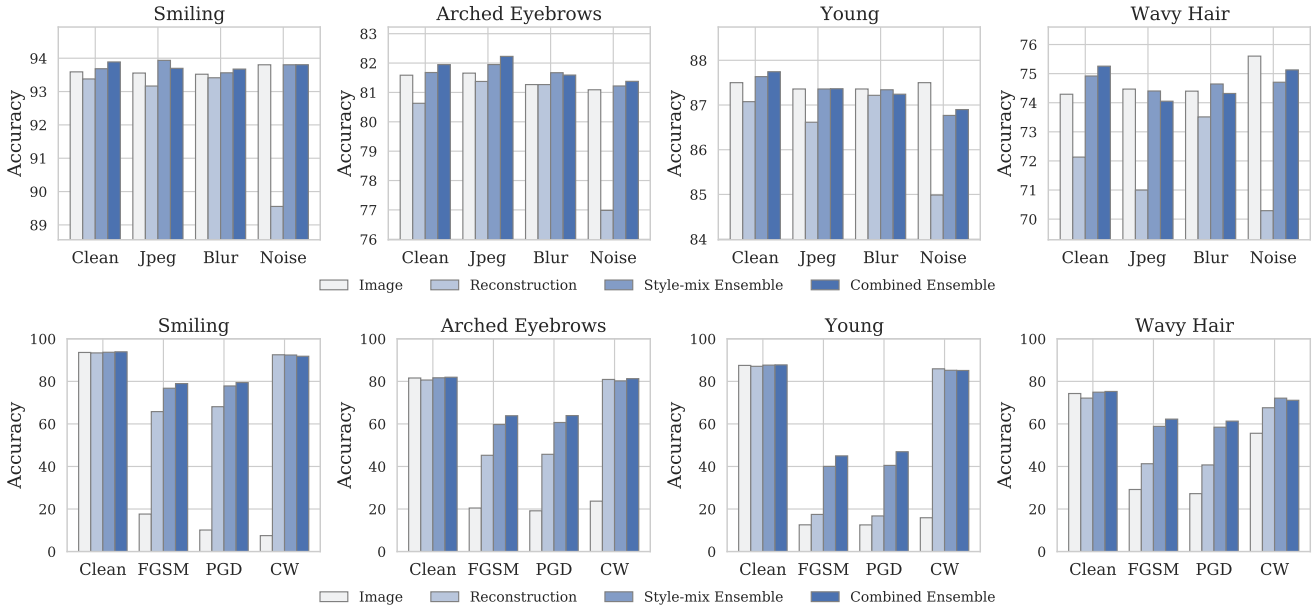


Figure 11: Robustness to corruptions. We show accuracy on a corrupted image (Image), the GAN reconstruction (Reconstruction), ensembling with GAN style-mixing (Style-mix Ensemble), and ensembling over both traditional and GAN views (Combined Ensemble). (top) On clean images, deep views increase performance across the 4 facial attributes. We test against different types of corruptions: jpeg, Gaussian blur, and Gaussian noise. The results on untargeted corruptions are mixed; in 6 of 12 cases, ensembling improves performance. (bottom) Adversarial attacks (FGSM, PGD, CW) greatly reduce accuracy. On all cases, just GAN reconstruction recovers significant performance. In the majority of cases, ensembling further improves performance.

an ensemble (Style-mix Ensemble). We also ensemble by combining style-mixing and traditional crop and color jittering (Combined Ensemble). On untargeted corruptions, the result of ensembling using GAN augmentations is mixed. Accuracy improves on JPEG and Blur corruptions for the Smiling and Arched Eyebrows attributes, but it does not improve for the Wavy Hair/Young attributes (in fact, Wavy Hair accuracy is higher on the corrupted images than clean images as the classifier is not impaired by the corruption). In the case of Gaussian Noise, classification of the reconstructed image drops, as the projection procedure fails to find a good reconstruction of the noisy image; thus adding GAN perturbations at test time only helps in one out of four attributes (Arched Eyebrows). Note that the classifier is not greatly sensitive to these types of corruptions: classification accuracy between the clean images and the corrupted ones are largely similar, this may be due to the initial downsampling operation on the attribute classifiers following [7], which may reduce the effect of these corruptions. In the targeted corruption setting, the benefits of GAN reconstruction and ensembling is more pronounced. Using the GAN reconstruction rather than the corrupted image increases classifi-

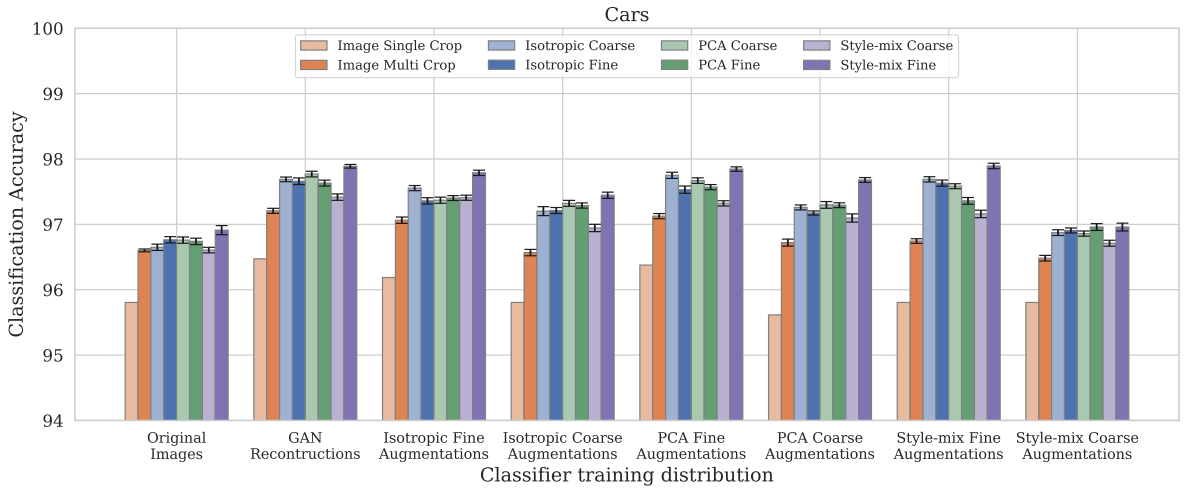


Figure 12: Classifier training variations: cars. Starting with a classifier trained on image crops (Original Images), we finetune the classifier using different types of images projected through the generator's latent code – either GAN Reconstructions or manipulated latent codes using the isotropic, PCA, or style-mixing augmentations at coarse or fine layers during training (along the x-axis). At test time, we then evaluate with an ensemble of different types of GAN perturbations and image crops (different colored bars). On the car domain, we find that adding GAN augmentations only at test time, when the classifier is only trained on dataset images, offers a small increase in accuracy, but there are greater benefits when the classifier is further finetuned on GAN images. In particular, using the fine layer style-mixing augmentation is most beneficial at training and test time. Error bars indicate standard error over 20 bootstrapped samples from 32 ensemble elements.

cation accuracy on these corrupted images, and in the majority of cases, ensembling multiple views from the GAN further improves performance.

2.3. Additional experiment settings: cars

Training distributions In the main text, we focus on the fine-level style-mixing augmentation when finetuning the classifier on generated samples, which corresponds to small color changes, such as changing the color of the car. In Fig. 12 we show results on finetuning the classifier with the remaining isotropic, PCA, and style-mixing augmentations, at both coarse and fine layers. On the car domain, we find that training on fine layer perturbations tend to outperform training on coarse layer perturbations, and at test time, ensembling with fine layer style-mixing augmentations is best. As using multiple image crops in the ensemble consistently increases classification accuracy, we use a combination of 16 image crops and 16 cropped and perturbed GAN reconstructions when ensembling using the GAN output.

Effect of image augmentations When training the classifier, we use standard random resize, crop, and horizontal flip following the transformations used in ImageNet training[2]. Here, we also investigate the effect of an additional image rotation augmentation, at both training and test time (we use a random rotation between -10 and 10 degrees prior to the previous transformations of resizing, cropping, and flipping). When trained on resize, crop, and flip transformations on images, ensembling GAN augmentations outperforms image multi-crop classification using the crop and scale+crop augmentations at test time. Next, we train a classifier using rotate, resize, crop, and flip transformations on images; in this case, the using the scale+crop test-time augmentation on images attains the highest accuracy. We then finetune these classifiers by also training on the GAN generated variants: here, combining image and GAN augmentations at test time outperform test-time image augmentations alone, but the accuracy of these finetuned classifiers is lower than the highest accuracy attained by the classifier trained on images with the rotate augmentation (Fig. 13). Given the current limitations in GAN reconstruction ability, this suggests that carefully chosen image augmentations *during training* can slightly outperform the benefits of GAN-based augmentations *at test-time*.

2.4. Additional experiment settings: cats

Training distributions We use a similar setup for the cat classification task as the cars task. Fig 14 shows the classification result, trained on the Original Images (left), and finetuned on GAN reconstructions or each type of perturbation method. In this domain, we find that *training* with the coarse layer style-mix augmentation offers the largest benefit image classification accuracy, and furthermore, ensembling with this same augmentation offers an additional increase in accuracy. Unlike the car domain, the cat im-

[2] https://github.com/pytorch/examples/blob/master/imagenet/main.py

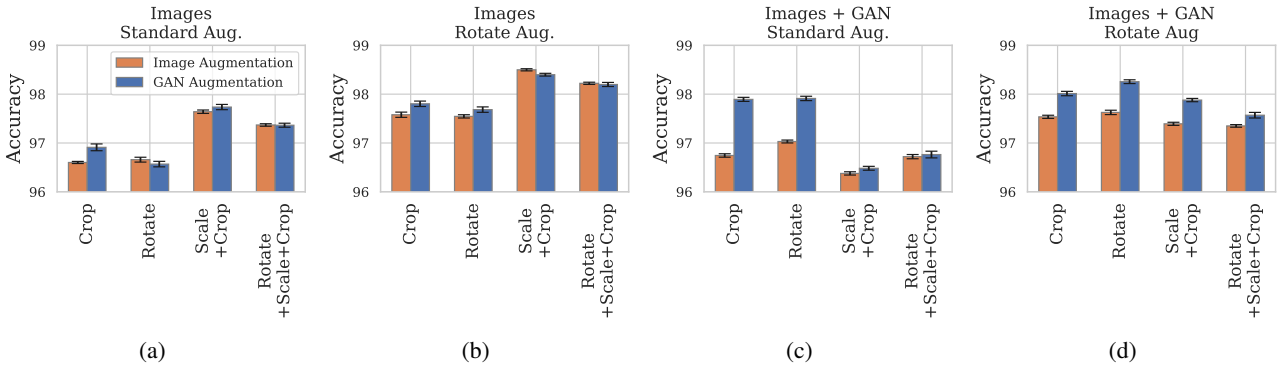


Figure 13: Comparing image and GAN augmentations during training: cars. We investigate the effect of various image augmentations during classifier training and at test time. (a) We first train a classifier on the cars dataset following the ImageNet training transformations, which include random resize, crop, and horizontal flip; applying GAN augmentations at test time can slightly outperform using only image augmentations. (b) Next, we add a random rotate transformation in addition to the previous image transformations during classifier training, and again train on the image dataset; this classifier outperforms the previous classifier, and in this case ensembling images with the scale+crop augmentation at test time is the best. When the classifiers are trained with GAN images, either with the standard training transformations (c) or the additional rotate transformation (d), adding GAN augmentations outperforms using image augmentations at test time, but the classifier's overall accuracy is lower. This suggests that carefully chosen image augmentations during classifier training can still slightly outperform the benefits of GAN-based augmentations at test-time, due to current limitations in image reconstruction using GANs.

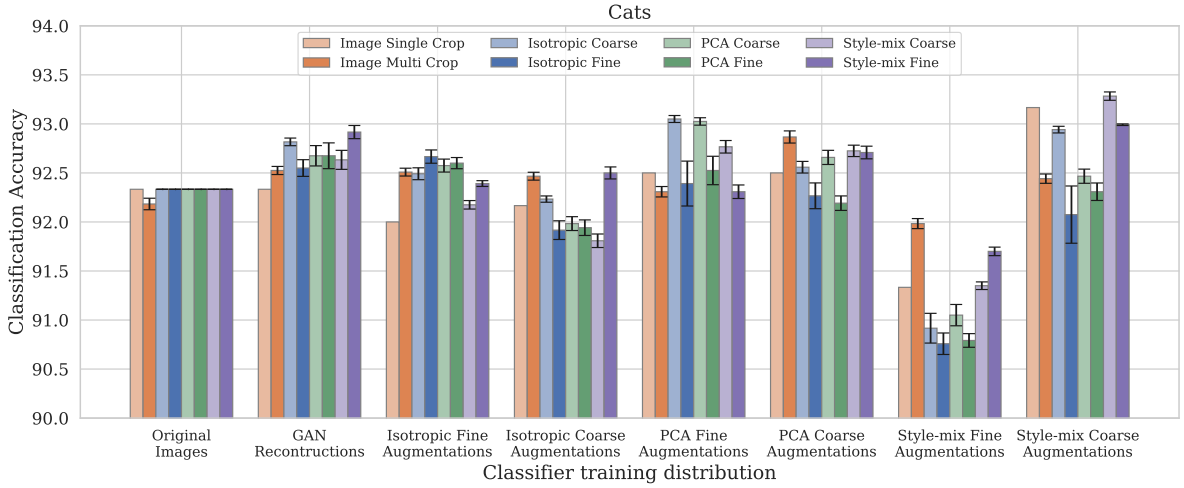


Figure 14: Classifier training variations: cats. Similar to the car domain, we start with a classifier trained on image crops (Original Images) and then finetune the classifier using different types of images projected through the generator's latent code – either GAN Reconstructions or manipulated latent codes using the isotropic, PCA, or style-mixing augmentations at coarse or fine layers during training (along the x-axis). At test time, we then evaluate with an ensemble of different types of GAN perturbations and image crops (different colored bars). On the cat domain, we find that using coarse layer style-mixing offers the highest classification accuracy; it increases image classification accuracy on the test set compared to training the classifier only on image crops, and ensembling using the GAN outputs offers a small additional increase. Error bars indicate standard error over 20 bootstrapped samples from 32 ensemble elements.

ages are preprocessed to align and center the face, and so empiricially the benefits of ensembling multiple image crops are less consistent. When ensembling with GAN augmentations, we take the original center-cropped image, and 31 perturbed views from the GAN that are averaged and weighted with ensemble weight hyperparameter α.

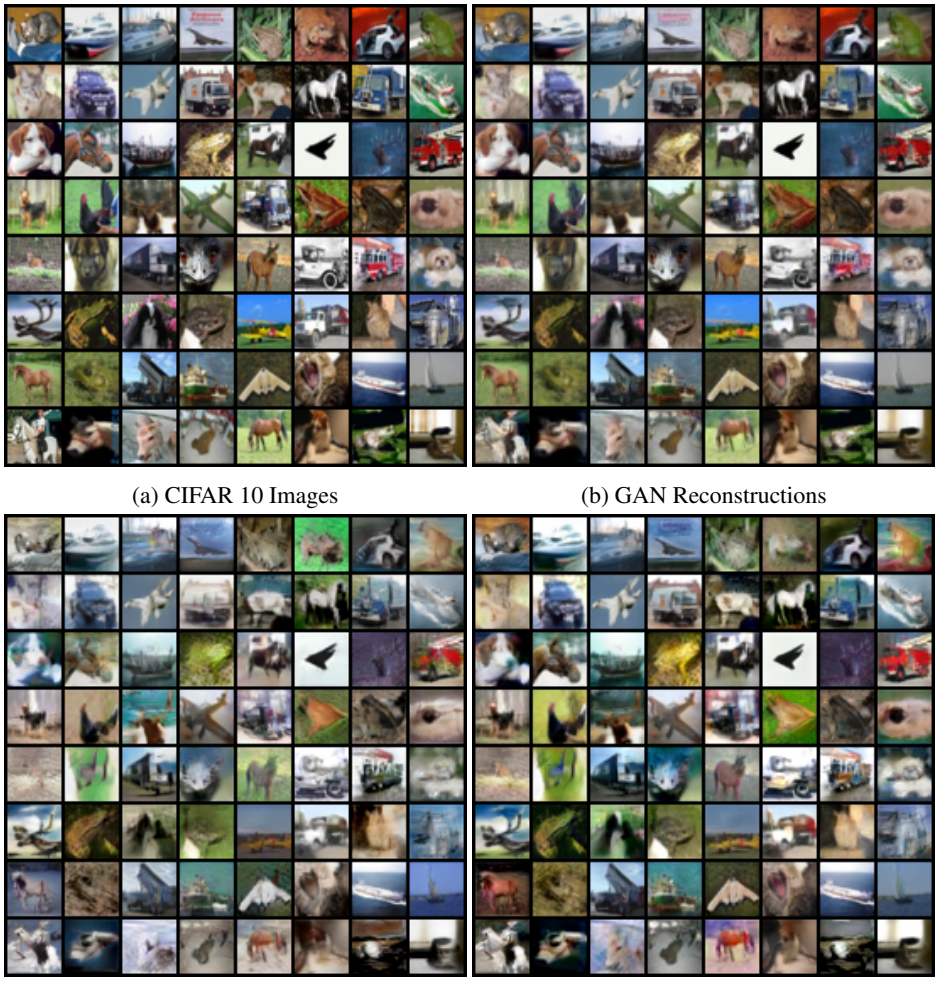
2.5. GAN augmentations with CIFAR10

We use a class-conditional StyleGAN2 [6] to conduct preliminary experiments on the CIFAR10 dataset [11] (we also tried the unconditional CIFAR StyleGAN, but obtained poorer reconstructions). We first reserve the final 5000

Figure 15: Classifier training variations: CIFAR10. With a classifier trained on the CIFAR10 training split, standard test classification accuracy and adding GAN augmentations at test time perform similarly. When the classifier is then finetuned on GAN reconstructions or perturbed GAN reconstructions, adding GAN augmentations at test time offers a small improvement over standard image classification, but the overall classification accuracy is lower, suggesting that the GAN reconstructions cannot perserve the true class well enough to match the classification performanance of the CIFAR10 dataset simages.

training images for validation, and train a Resnet-18 classifier on the remaining 45000 training images, which achieves accuracy of 95.04% on the CIFAR10 test set[3]. To project the 32x32 images into the GAN latent space, we first predict the class of each image, use the average W latent of the predicted class to initialize optimization, and optimize for 200 steps, taking about eight seconds per image. We show qualitative examples of CIFAR10 test images, and their GAN reconstructions, and the result of swapping a random latent code, i.e. style-mixing, corresponding to the same predicted class at the seventh and eighth layers in Fig. 16 (the GAN has a total of eight layers). When the classifier is trained only on real images, we find that classifying the reconstructed test images is harder than the real images (accuracy drops from 95.04% to 84.68%). With the ensemble weighting hyperparameter α, we find that while the validation split has a small increase in classification accuracy of 0.04%, although when applying this same ensemble weight to the test split, the accuracy increase is only 0.01%. However, using the *optimal* α weight on the test split increases accuracy by 0.05%. We then finetune the classifier on GAN reconstructions of the training set, and additionally perform style-mixing in the seventh and eighth layers. When trained with style-mixing, adding GAN-generated views at test time can outperform standard image classification, but note that the overall accuracy of the classifier is lower (Fig. 15). These initial results suggest that the GAN reconstructions currently not perserve the true class well enough to attain the same performance as classification of CIFAR10 images, and due to the smaller resolution of the CIFAR10 StyleGAN, the style-mixing operation in later layers may not be sufficiently disentangled from class identity to offer benefits when ensembling at test time, comparing to classifying images directly.

[3] https://github.com/kuangliu/pytorch-cifar



(a) CIFAR 10 Images

(b) GAN Reconstructions

(c) Style-mix layer 7

(d) Style-mix layer 8

Figure 16: Qualitative examples of CIFAR10 GAN reconstructions. (a) CIFAR10 images from the test set, (b) the GAN reconstructions of the test images (c) swapping the reconstructed latent code with a random latent code from the same predicted class (style-mixing) at layer 7, and (d) style-mixing at the final layer.

References

[1] Lucy Chai, Jonas Wulff, and Phillip Isola. Using latent space regression to analyze and leverage compositionality in gans. In *Int. Conf. Learn. Represent.*, 2021. 1

[2] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. In *Adv. Neural Inform. Process. Syst.*, 2020. 2

[3] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Int. Conf. Comput. Vis.*, pages 2961–2969, 2017. 4

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. 2

[5] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *Int. Conf. Learn. Represent.*, 2018. 1

[6] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Adv. Neural Inform. Process. Syst.*, 2020. 11

[7] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1, 9

[8] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 1, 2, 6, 7

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[10] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 1

[11] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 11

[12] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Int. Conf. Comput. Vis.*, 2015. 1, 5

[13] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2012. 1

[14] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 2015. 2

[15] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 1

[16] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2, 4

[17] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *Eur. Conf. Comput. Vis.*, 2020. 6, 7