Appendix

6. GAN and InfoGAN

The Generative Adversarial Network [16] is a generative model trained via a minimax game performed between a generator G and a discriminator D:

$$\min_G \max_D V(G,D) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\text{noise}}}[\log(1 - D(G(\boldsymbol{z})))], \quad (13)$$

where $\boldsymbol{z}$ is a noise variable sampled from a prior distribution $p_{\text{noise}}(\boldsymbol{z})$, the generator G maps $\boldsymbol{z}$ to the pixel space to synthesize images, and the discriminator D predictes if an images is sampled from the real distribution or not. After convergence, the G should be able to synthesize realistic images and the D cannot tell if an image is fake or not.

The InfoGAN [8] augments the GAN loss (Eq. 13) with a regularization term:

$$\min_{G,Q} \max_D V_{\text{INFO}}(G,D) = V(G,D) - \beta I(\boldsymbol{c}; G(\boldsymbol{c},\boldsymbol{z})), \quad (14)$$

where $I(\boldsymbol{c}; G(\boldsymbol{c},\boldsymbol{z}))$ is the mutual information between a subset of latent codes $\boldsymbol{c} \in \mathbb{R}^d$ and the generated samples $G(\boldsymbol{c},\boldsymbol{z})$. By maximizing the mutual information, the latent codes $\boldsymbol{c}$ are able to represent a set of interpretable salient variations in data. In practice, the mutual information is approximated by a variational lower bound and is implemented with an auxiliary network Q, trained together with G by regression loss for predicting the latent codes $\boldsymbol{c}$.

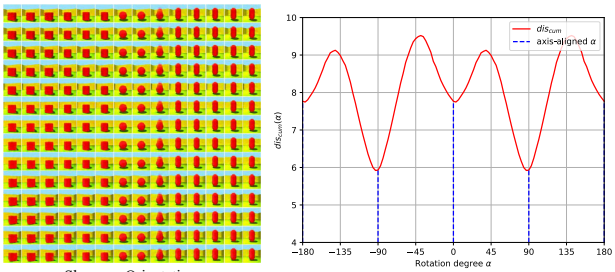
7. Datasets Introduction

CelebA This is a dataset of 202,599 images of cropped real-world human faces, containing various poses, backgrounds and facial expressions. We use the cropped center 128×128 area in this paper.

Shoes+Edges This dataset contains the commonly used image-to-image translation datasets Shoes and Edges, but mixes the 50,000 Shoes images and 50,000 Edges images together to form a 100,000-image dataset in 128×128.

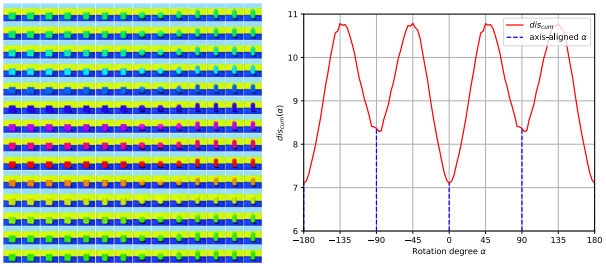
Clevr-Simple This dataset is a variant of the Clevr dataset, which contains an object featuring four factors of variation: object color, shape, and location (both horizontal and vertical). It contains 10,000 256×256 images.

Clevr-Complex This dataset retains all variations from Clevr-Simple but adds a second object and multiple sizes for a total of 10 factors of variation (5 per object). It contains 10,000 256×256 images.



Shape vs Orientation

Figure 11. Shape vs Orientation.



Shape vs Obj-Color

Figure 12. Shape vs Obj-Color.

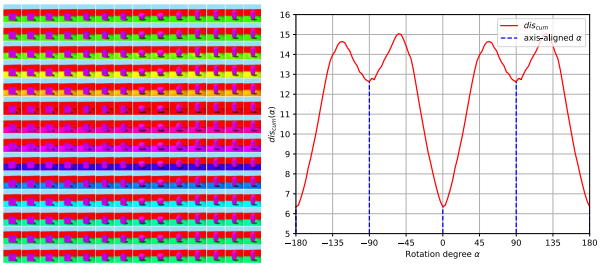
FFHQ This is a dataset of aligned images of human faces crawled from Flickr (70,000 in tatal). We use the 512×512 version in our paper. Link: https://github.com/NVlabs/ffhq-dataset.

DSprites This is a dataset of 2D shapes generated from 5 independent factors, which are *shape* (3 values), *scale* (6 values), *orientation* (40 values), *x position* (32 values), and *y position* (32 values). All combinations are present exactly once, with the total number of binary 64×64 images 737,280. Link: https://github.com/deepmind/dsprites-dataset.

3DShapes This is a dataset of 3D shapes generated from 6 independent factors, which are *floor color* (10 values), *wall color* (10 values), *object color* (10 values), *scale* (8 values), *shape* (4 values), *orientation* (15 values). All combinations are present exactly once, with the total number of 64×64 images 480,000. Link: https://github.com/deepmind/3d-shapes.

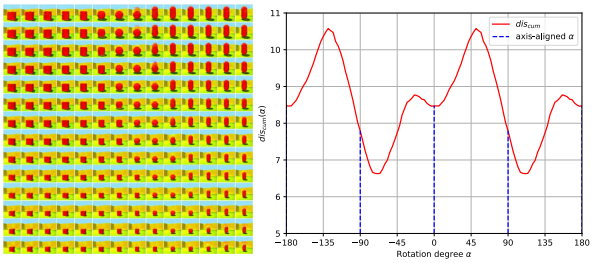
8. More Experiments of Rotating Latent Space

The plot for different semantic-pairs on 3DShapes are shown in Fig. 11 - 25. Similar results for CelebA are shown in Fig. 26 - 46. For most semantic-pairs, the corresponding α rotation plots agree with the assumption of Perceptual



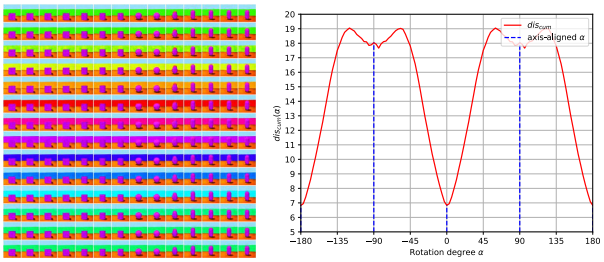
Shape vs Floor-Color

Figure 13. Shape vs Floor-Color.



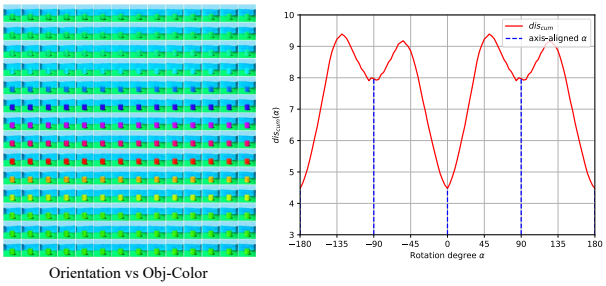
Shape vs Size

Figure 14. Shape vs Size.



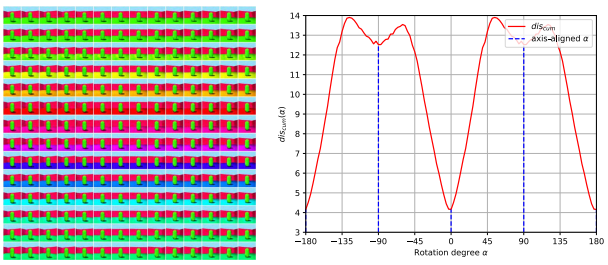
Shape vs Wall-Color

Figure 15. Shape vs Wall-Color.



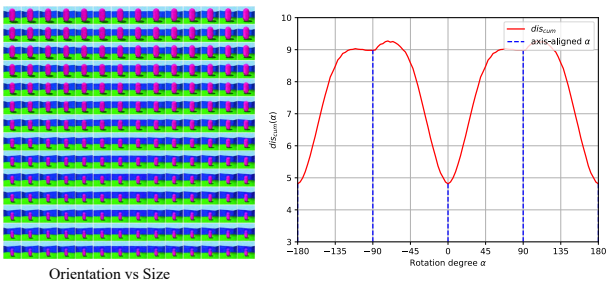
Orientation vs Obj-Color

Figure 16. Orientation vs Obj-Color.



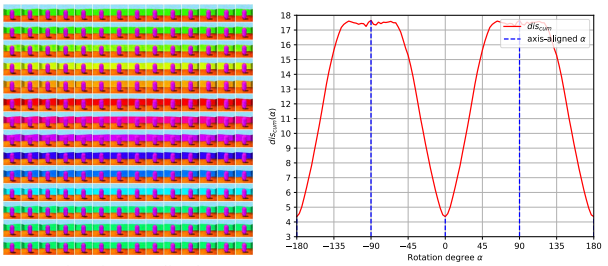
Orientation vs Floor-Color

Figure 17. Orientation vs Floor-Color.



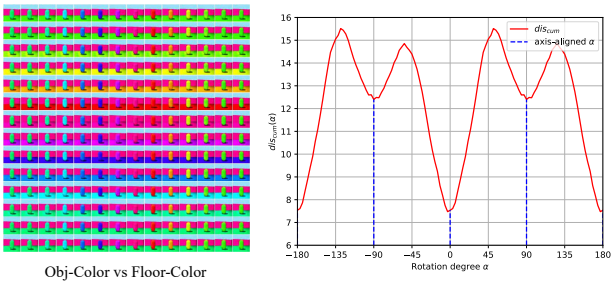
Orientation vs Size

Figure 18. Orientation vs Size.



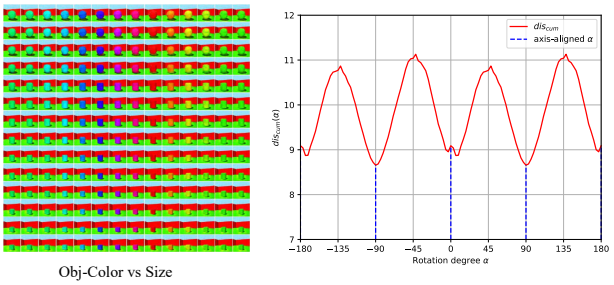
Orientation vs Wall-Color

Figure 19. Orientation vs Wall-Color.



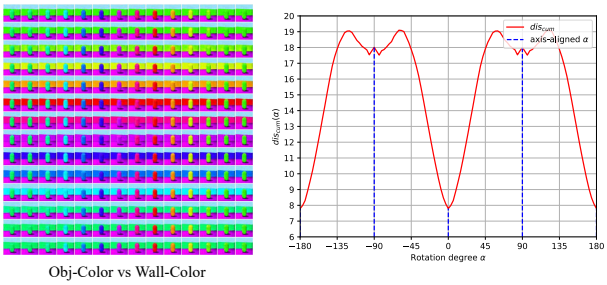
Obj-Color vs Floor-Color

Figure 20. Obj-Color vs Floor-Color.



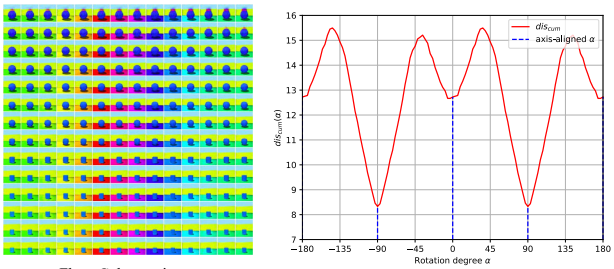
Obj-Color vs Size

Figure 21. Obj-Color vs Size.



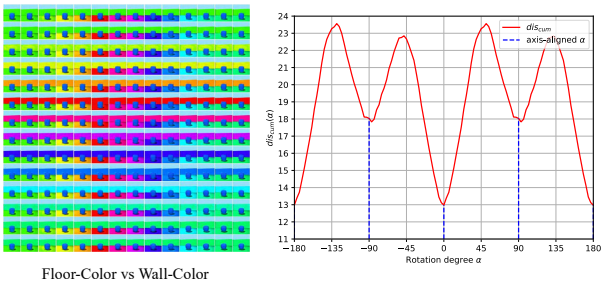
Obj-Color vs Wall-Color

Figure 22. Obj-Color vs Wall-Color.



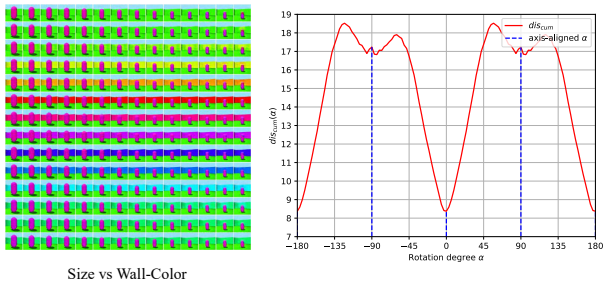
Floor-Color vs size

Figure 23. Floor-Color vs Size.



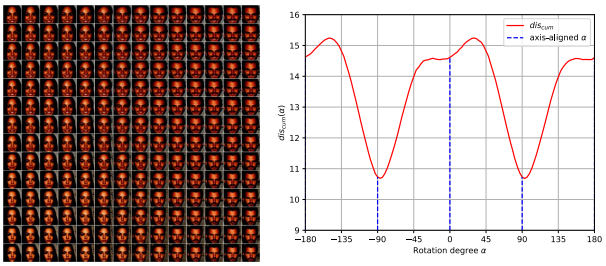
Floor-Color vs Wall-Color

Figure 24. Floor-Color vs Wall-Color.



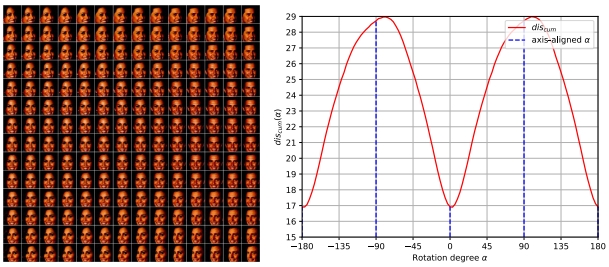
Size vs Wall-Color

Figure 25. Size vs Wall-Color.



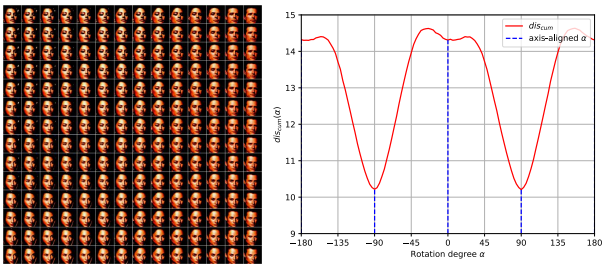
Gender vs Smile

Figure 26. Gender vs Smile.



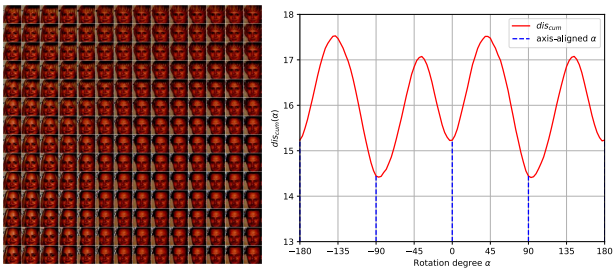
Gender vs Azimuth

Figure 27. Gender vs Azimuth.



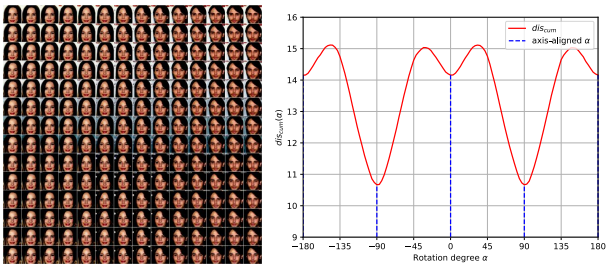
Gender vs Elevation

Figure 28. Gender vs Elevation.



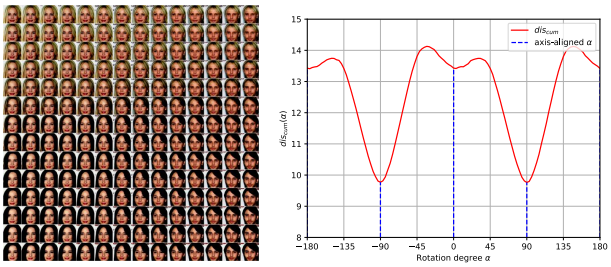
Gender vs Fringe

Figure 29. Gender vs Fringe.



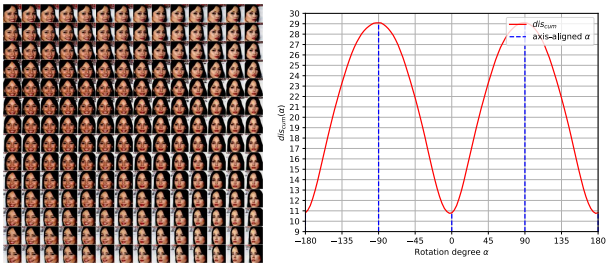
Gender vs Background

Figure 30. Gender vs Background.



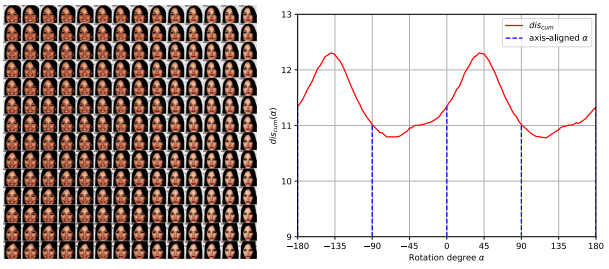
Gender vs Hair-Color

Figure 31. Gender vs Hair-Color.



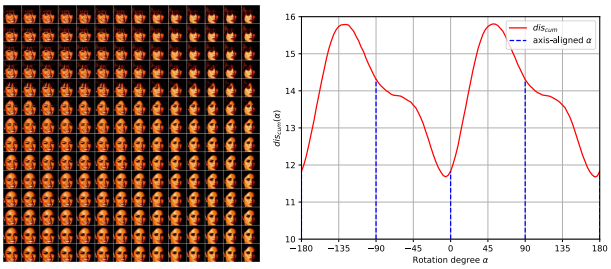
Smile vs Azimuth

Figure 32. Smile vs Azimuth.



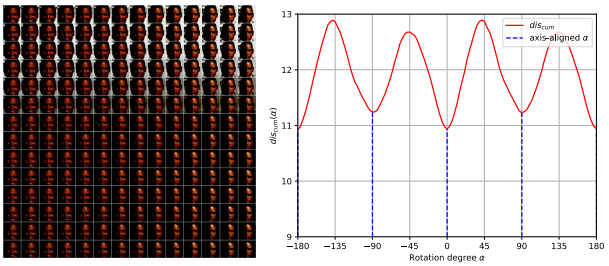
Smile vs Elevation

Figure 33. Smile vs Elevation.



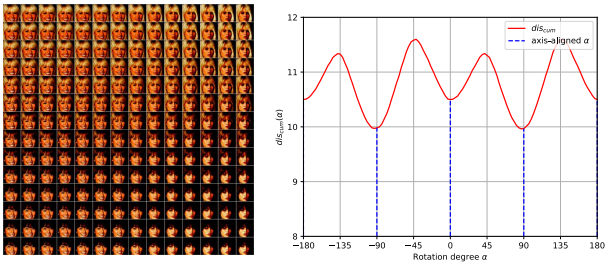
Smile vs Fringe

Figure 34. Smile vs Fringe.



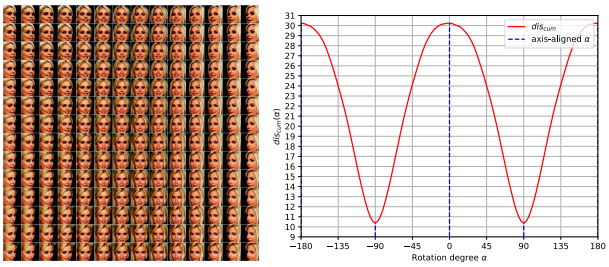
Smile vs Background

Figure 35. Smile vs Background.



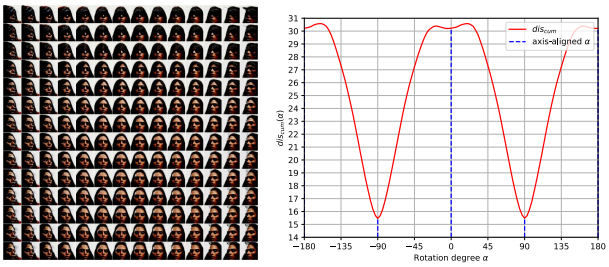
Smile vs Hair-Color

Figure 36. Smile vs Hair-Color.



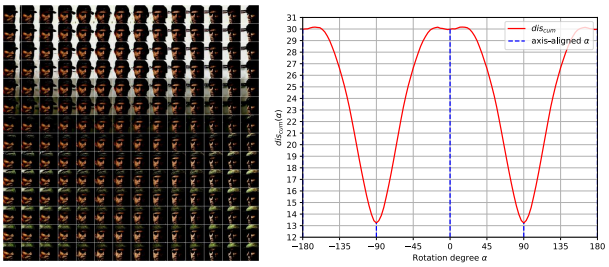
Azimuth vs Elevation

Figure 37. Azimuth vs Elevation.



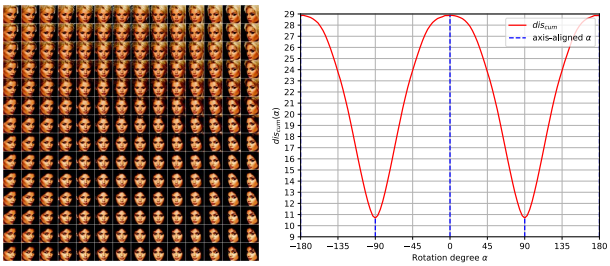
Azimuth vs Fringe

Figure 38. Azimuth vs Fringe.



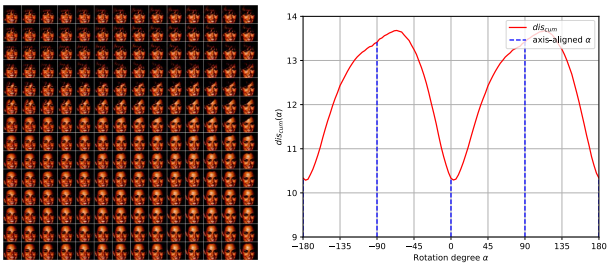
Azimuth vs Background

Figure 39. Azimuth vs Background.



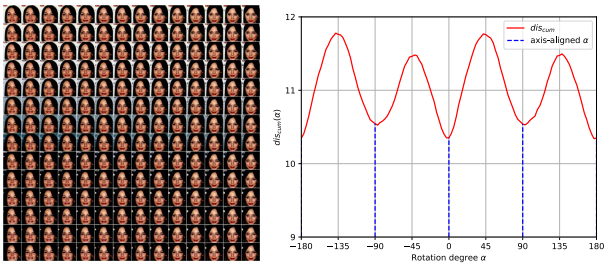
Azimuth vs Hair-Color

Figure 40. Azimuth vs Hair-Color.



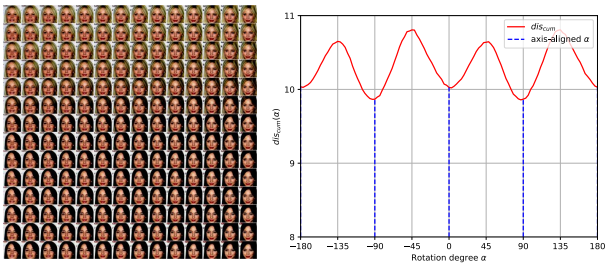
Elevation vs Fringe

Figure 41. Elevation vs Fringe.



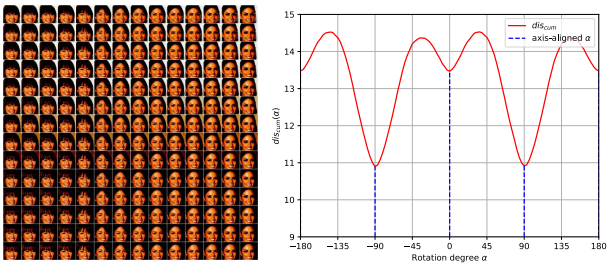
Elevation vs Background

Figure 42. Elevation vs Background.



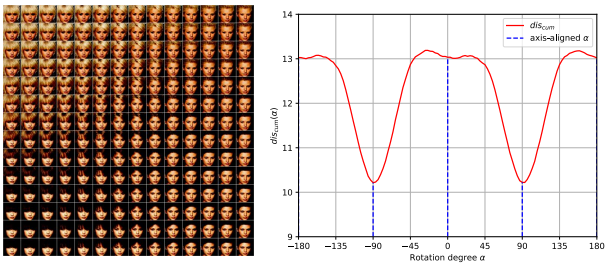
Elevation vs Hair-Color

Figure 43. Elevation vs Hair-Color.



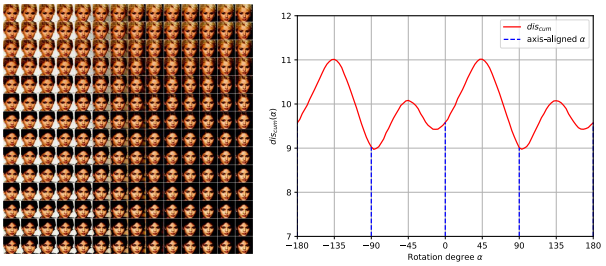
Fringe vs Background

Figure 44. Fringe vs Background.



Fringe vs Hair-Color

Figure 45. Fringe vs Hair-Color.



Background vs Hair-Color

Figure 46. Background vs Hair-Color.

Simplicity, *i.e*. the accumulated perceptual distance scores become local minima when the accumulation directions are aligned with the latent axes ($\alpha = -180, -90, 0, 90, 180$). However, there exist still some exceptions, such as Shape vs Size, and some attributes plotted against Azimuth, *etc*. These are (1) sometimes due to the imperfectly learned representations which are not fully disentangled, and (2) sometimes due to the domination of variations encoded by one dimension over another (*e.g*. Azimuth), leading the Perceptual Simplicity phenomenon not obvious. But in general, the assumption holds for most semantic-pairs.

9. TPL Pros and Cons

Pros: 1) Unlike [12], the TPL computation does not rely on pair-wise comparisons among a herd of models (trained with different hyper-parameters and seeds) to assign a score to a single model. This ensures the TPL is a more efficient method for model selection, and also enables its ability to work as a rough unsupervised metric to evaluate disentanglement quality. 2) Unlike [60], the TPL does not need to train an extra classifier to assign a score to a model, indicating it is a more general and efficient approach. 3) Unlike [25], the TPL leverage the perceptual anisotropy in a disentangled representation, which can select more interpretable ones than the PPL proposed in [25] which only assign a score to a model by only evaluating the perceptual smoothness in the latent space.

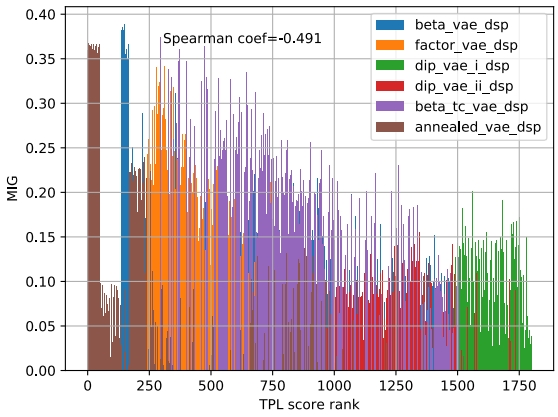


Figure 47. TPL (act>0) vs MIG. Ranked by TPL scores.

Cons: 1) The TPL scores can be biased by the generative ability of the generator, *e.g*. it usually ranks a blurred-image generator higher than a more-detailed-image generator even their disentanglement levels are similar. To alleviate this problem, we recommend using TPL together with some generative quality metrics (*e.g*. FID, IS) to filter out the *cheating* models that achieve disentanglement at the severe cost of generative quality. 2) The TPL is based on the assumption that disentangled representations contain perceptually simple variations along their latent axes. This assumption may not hold for every dataset or for every concept. In those case, supervised disentanglement learning methods may be more preferable.

10. More Results of TPL Experiments

Here we show more correlation plots for more metrics and the dimensions using the pretrained checkpoints. The correlation coefficients for each setting are shown in the plots. In Fig. 47, 48, 49, and 50, we show the plots with all dimensions of activation (act>0) taken into account. There are descending trends shown in each figure, but the reason that the correlation scores are low is due to the left most samples (around <250) shown in each plot. These samples are strangely positioned at the top of the whole rank by TPL, but are generally not scored high by supervised metrics. This is because these samples encode only a subset of factors in the dataset (3 out of 5), and some of them are detected by the supervised metrics and are assigned with low scores. In Fig. 51, 52, 53, 54, 55, 56, 57 and 58, we show the plots of samples with active dimensions larger than 3 and 4. In these figures, the TPL ranks these models much better, and the correlation coefficients also agree with the descending trends better in these plots.

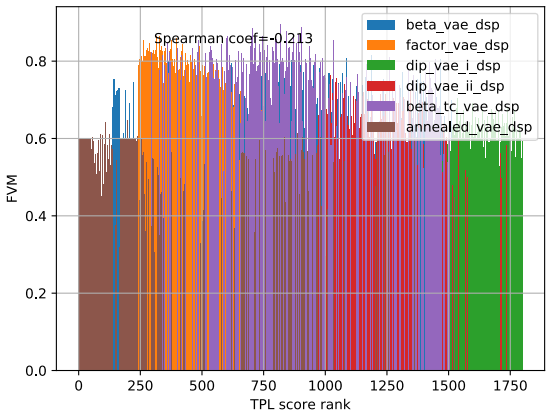


Figure 48. TPL (act>0) vs FVM. Ranked by TPL scores.

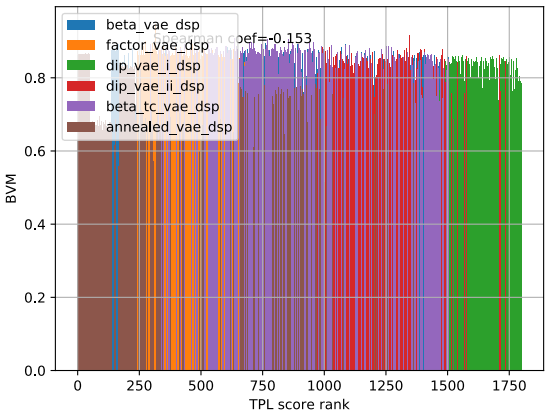


Figure 50. TPL (act>0) vs BVM. Ranked by TPL scores.

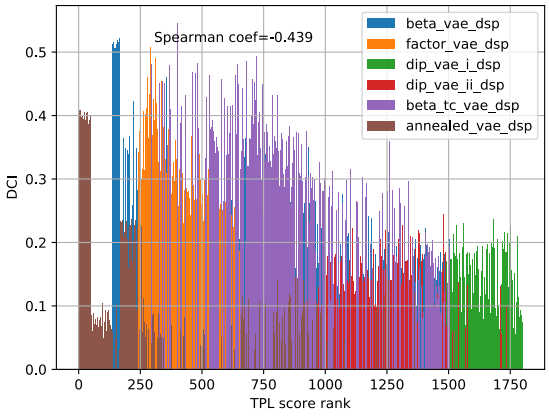


Figure 49. TPL (act>0) vs DCI. Ranked by TPL scores.

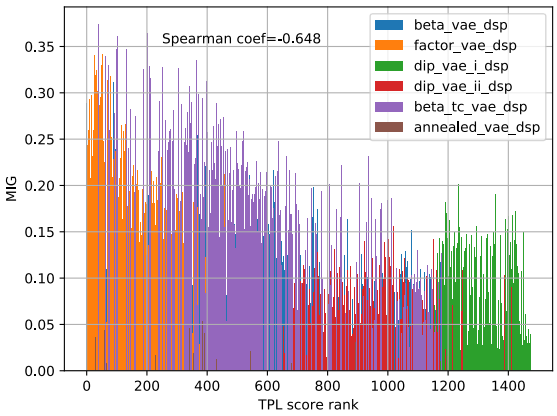


Figure 51. TPL (act>3) vs MIG. Ranked by TPL scores.

11. Ablation Study on the Number of Rectangles J

We show the impact of the number of rectangles used in our SC modules on the CelebA dataset. We vary J from 1 to 9 while keeping all other factors unchanged. The results are shown in Table 6. We can see using too few rectangles harms FID (variation controlled by each code is too simple) while using too many harms TPL (each code controls entangled variations). A balanced J is about 3∼6.

J	TPL	PPL	FID
1	**5.4**	**29.2**	9.0
3	7.0	35.8	7.7
4	7.1	38.4	**5.9**
6	8.1	38.9	6.0
9	8.2	40.0	6.2

Table 6. Ablation on J on CelebA dataset.

12. More Qualitative Results

More traversals are shown in Fig. 59 - Fig. 70. For Clevr-Complex, we can see our model learns the position concepts for both objects, but other concepts like size and shape of objects are still entangled. This indicates the multi-object disentangled representation learning is still a hard and unsolved problem.

13. More Image Editing

More image editing experiments (similar to Fig. 10) are shown in Fig. 71 and Fig. 72. We can see the attributes of background, azimuth, smile, hat, fringe, skin color are successfully disentangled in the learned representation. However, there are still some flaws, such as the transfer of *hat* in

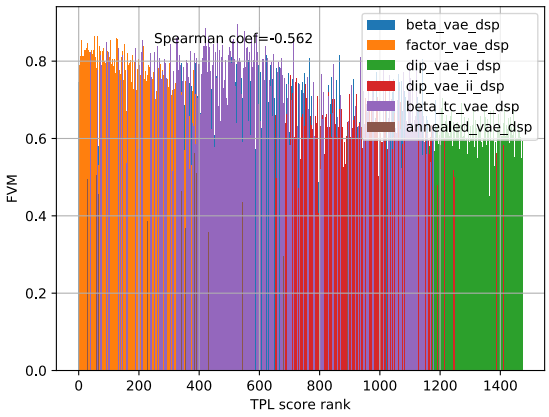


Figure 52. TPL (act>3) vs FVM. Ranked by TPL scores.

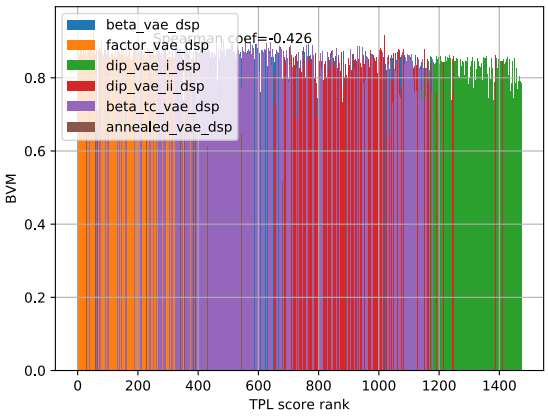


Figure 54. TPL (act>3) vs BVM. Ranked by TPL scores.

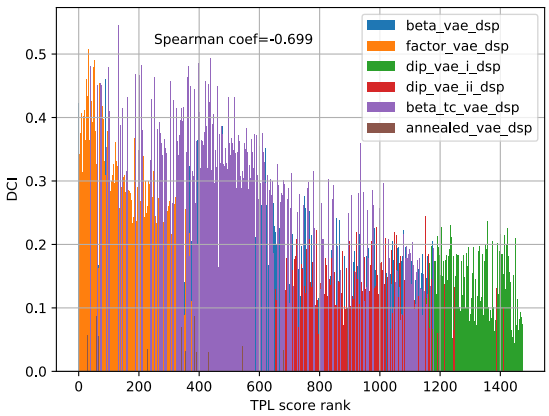


Figure 53. TPL (act>3) vs DCI. Ranked by TPL scores.

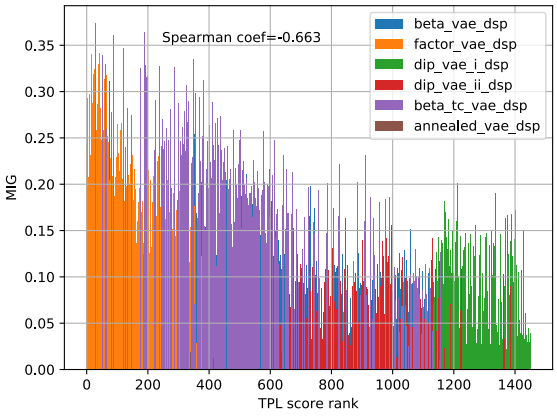


Figure 55. TPL (act>4) vs MIG. Ranked by TPL scores.

the last row of Fig. 72 changes the style and color of hats in the resulting images. This problem may be caused by the lack of data of various hats during training, but may also be alleviated by better models.

14. Implementations

The network architectures are shown in multiple Tables: on CelebA: 7, 8, 9; on Shoes+Edges: 10, 11, 12; on Clevr-Simple and Clevr-Complex: 13, 14, 15; on FFHQ: 16, 17, 18; on DSprites: 19, 20, 21; on 3DShapes: 22, 23, 24; All models are trained with the Adam optimizer. For both generator and discriminator optimizers, $\beta_1 = 0$, $\beta_2 = 0.99$, initial learning rate is 0.002, except for 3DShapes we set 0.005. Note that the Q network is trained together with the generator. The λ for Shoes+Edges, Clevr-Complex, CelebA, FFHQ is 0.01, for Clevr-Simple is 0.05, for DSprites and 3DShapes is 0.001. The p_{var} is 0.2 for DSprites and 3DShapes, and is 1 for other datasets. For CelebA dataset, models are trained for 4,000k images (around 19 epochs). For FFHQ dataset, models are trained until FID starts to saturate (around 28 epochs). For DSprites dataset, models are trained for 15,000k images (around 20 epochs). For 3DShapes dataset, models are trained for 8,000k images (around 18 epochs). For DSprites and 3DShapes, instead of randomly sampling the perturbed dimension k in the PC loss every iteration, we loop the k sequentially for each dimension every 1k images to achieve a stabler convergence.

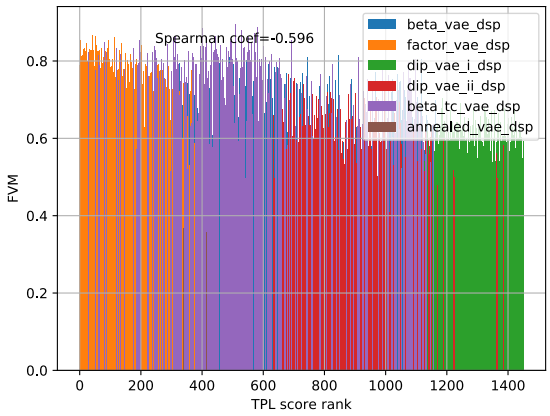


Figure 56. TPL (act>4) vs FVM. Ranked by TPL scores.

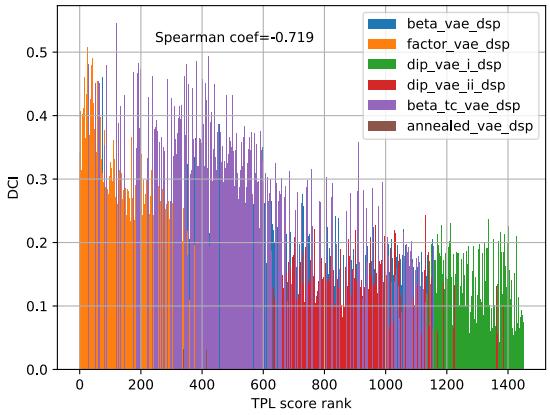


Figure 57. TPL (act>4) vs DCI. Ranked by TPL scores.

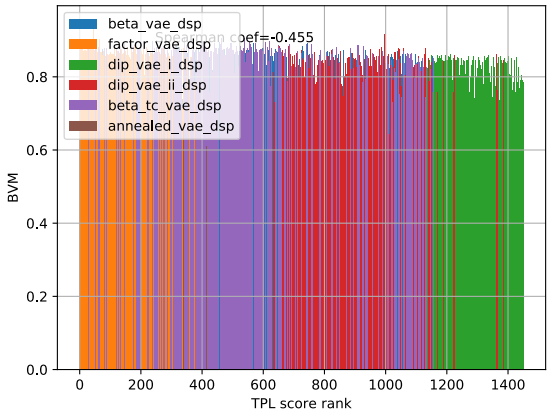


Figure 58. TPL (act>4) vs BVM. Ranked by TPL scores.

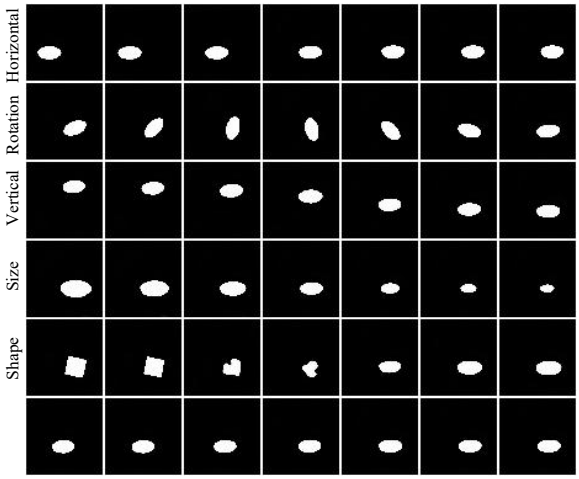


Figure 59. DSprites 1.

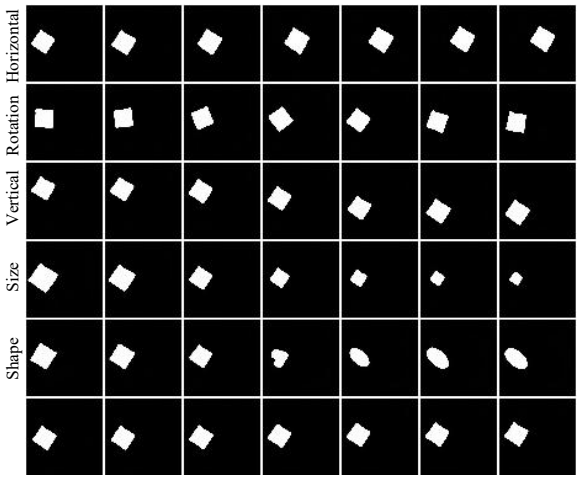


Figure 60. DSprites 2.

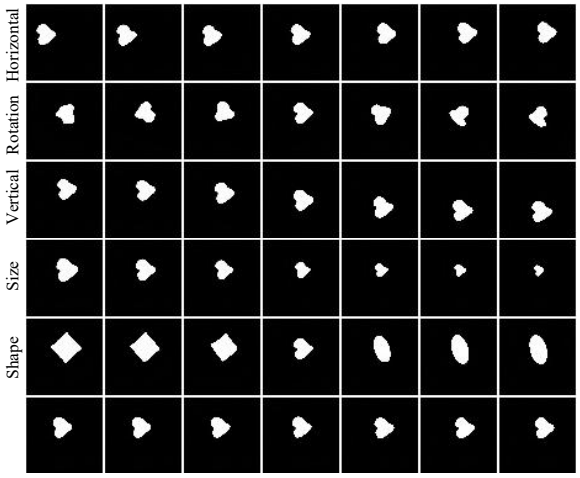


Figure 61. DSprites 3.

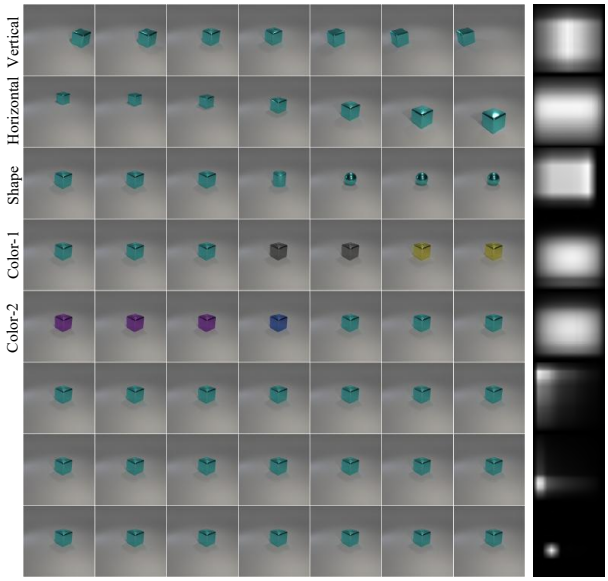


Figure 62. Clevr-Simple 1.

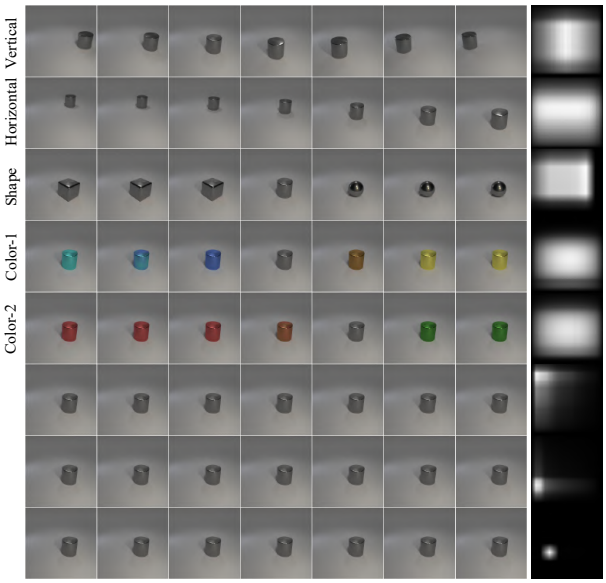


Figure 63. Clevr-Simple 2.

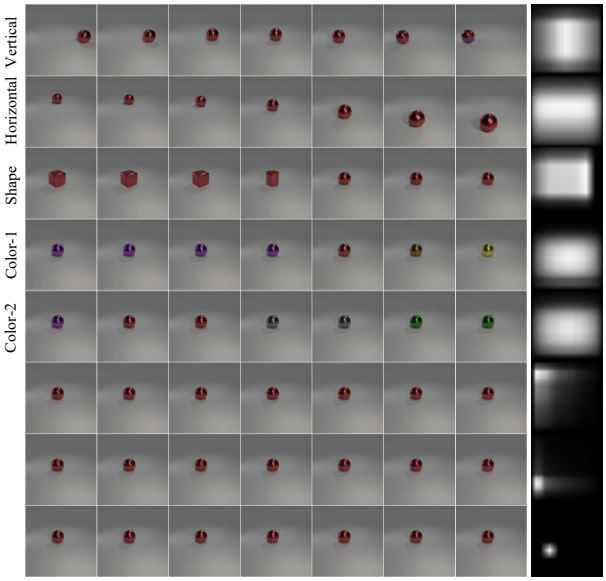


Figure 64. Clevr-Simple 3.

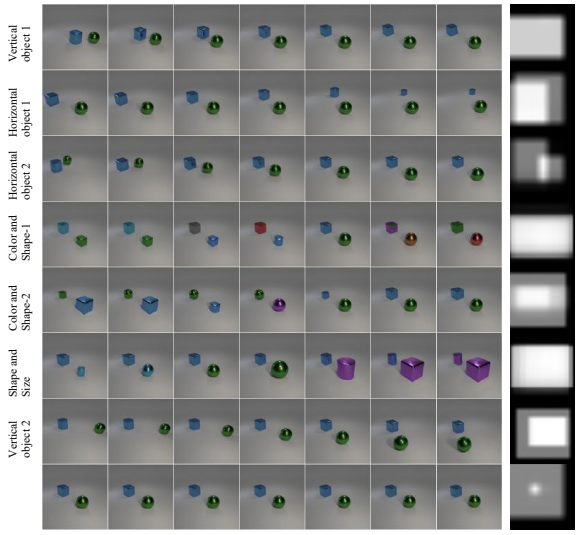


Figure 65. Clevr-Complex 1.

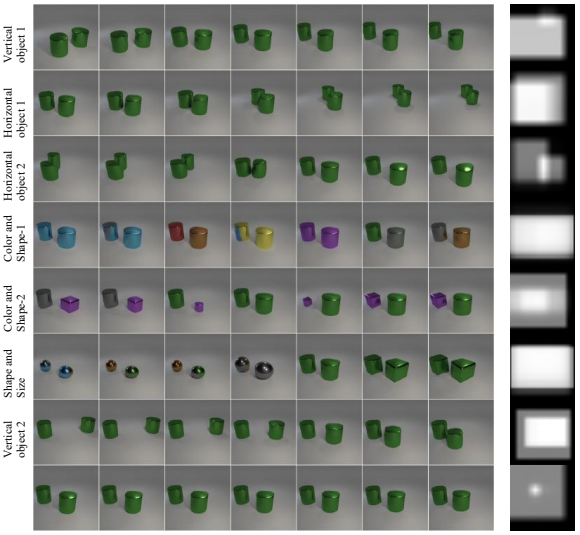


Figure 66. Clevr-Complex 2.

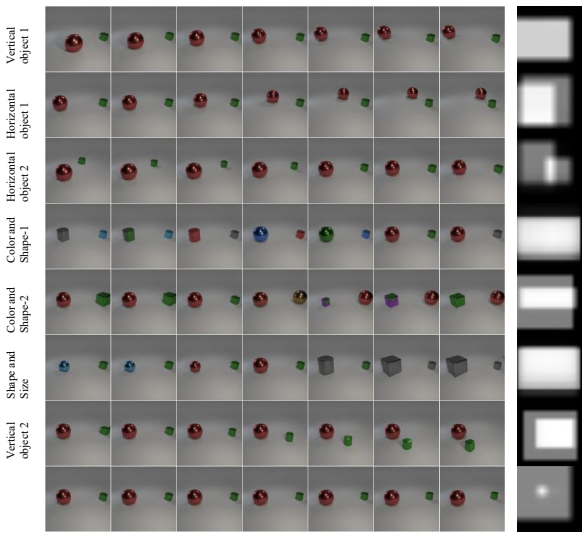


Figure 67. Clevr-Complex 3.

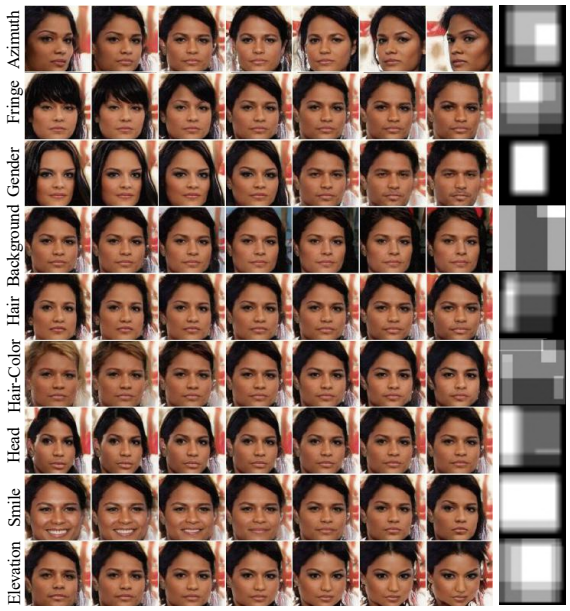


Figure 68. CelebA 1.

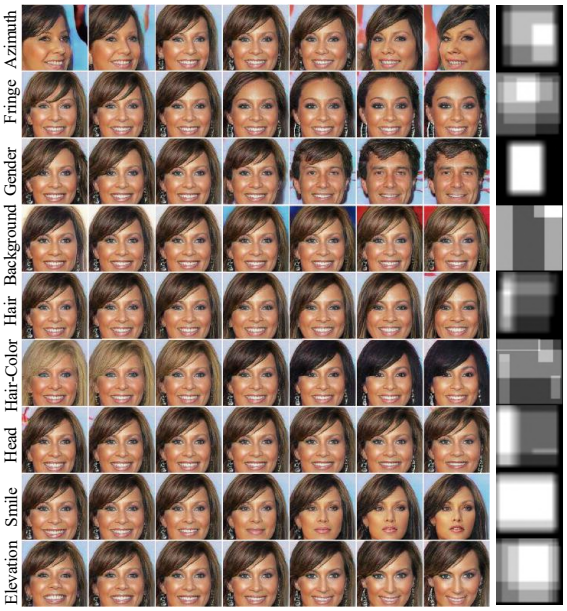


Figure 69. CelebA 2.

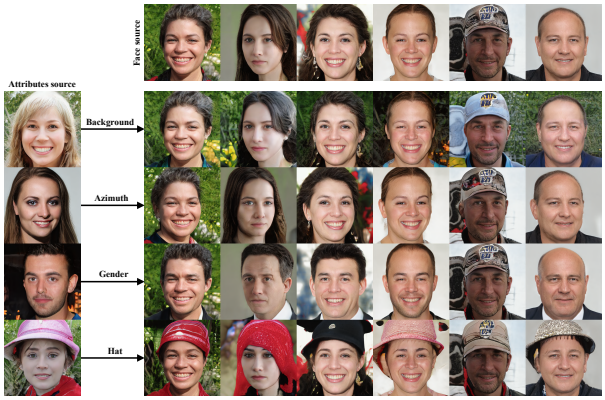


Figure 71. Image editing by transferring attributes.

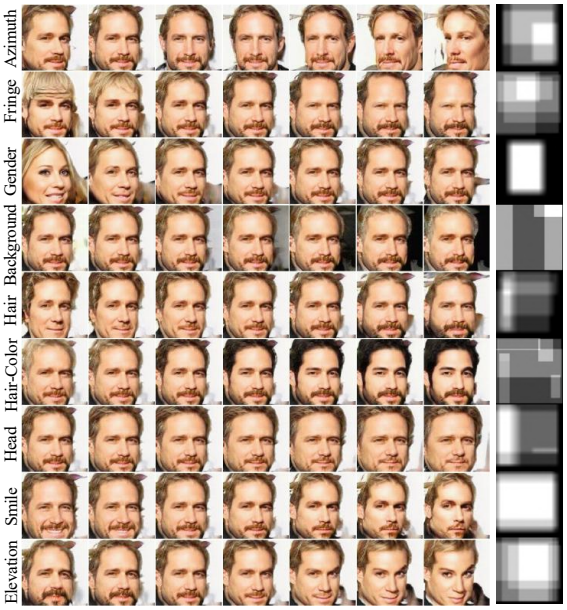


Figure 70. CelebA 3.

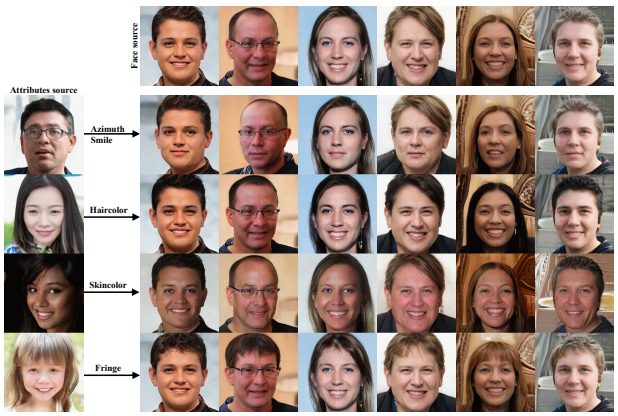


Figure 72. Image editing by transferring attributes.

Layer	Out Shape
Const	4x4x512
ResConv-up-1	8x8x512
SC-4-5	8x8x512
ResConv-id-1	8x8x512
Noise-2	8x8x512
ResConv-up-1	16x16x512
SC-6-5	16x16x512
ResConv-id-1	16x16x512
Noise-2	16x16x512
ResConv-up-1	32x32x512
SC-6-5	32x32x512
ResConv-id-1	32x32x512
Noise-2	32x32x256
ResConv-up-1	64x64x256
SC-6-5	64x64x256
ResConv-id-1	64x64x256
Noise-2	64x64x256
ResConv-up-1	128x128x256
SC-4-5	128x128x256
ResConv-id-1	128x128x128
Noise-2	128x128x128
ResConv-id-2	128x128x128
ResConv-id-1	128x128x3

Table 7. Generator on CelebA.

Layer	Out Shape
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x128
ResConv-down-2	16x16x256
ResConv-down-2	8x8x512
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	1

Table 8. Discriminator on CelebA.

Layer	Out Shape
ResConv-down-2	256x256x64
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x128
ResConv-down-2	16x16x256
ResConv-down-2	8x8x512
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	25

Table 9. Recognizer Q on CelebA.

Layer	Out Shape
Const	4x4x512
ResConv-up-1	8x8x512
SC-4-3	8x8x512
ResConv-id-1	8x8x512
Noise-2	8x8x512
ResConv-up-1	16x16x512
SC-6-3	16x16x512
ResConv-id-1	16x16x256
Noise-2	16x16x256
ResConv-up-1	32x32x256
SC-6-3	32x32x256
ResConv-id-1	32x32x256
Noise-2	32x32x128
ResConv-up-1	64x64x128
SC-6-3	64x64x128
ResConv-id-1	64x64x128
Noise-2	64x64x128
ResConv-up-1	128x128x128
SC-4-3	128x128x128
ResConv-id-1	128x128x64
Noise-2	128x128x64
ResConv-id-2	128x128x64
ResConv-id-1	128x128x3

Table 10. Generator on Shoes+Edges.

Layer	Out Shape
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x64
ResConv-down-2	16x16x128
ResConv-down-2	8x8x256
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	1

Table 11. Discriminator on Shoes+Edges.

Layer	Out Shape
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x64
ResConv-down-2	16x16x128
ResConv-down-2	8x8x256
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	15

Table 12. Recognizer Q on Shoes+Edges.

Layer	Out Shape
Const	4x4x512
ResConv-up-1	8x8x512
SC-1-2	8x8x512
Noise-1	8x8x512
ResConv-up-1	16x16x512
SC-1-2	16x16x512
Noise-1	16x16x256
ResConv-up-1	32x32x256
SC-1-2	32x32x256
Noise-1	32x32x256
ResConv-up-1	64x64x128
SC-1-2	64x64x128
Noise-1	64x64x128
ResConv-up-1	128x128x128
SC-1-2	128x128x128
Noise-1	128x128x64
ResConv-up-1	256x256x64
Noise-1	256x256x64
ResConv-id-1	256x256x3

Table 13. Generator on Clevr-Simple and -Complex.

Layer	Out Shape
ResConv-down-2	256x256x64
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x64
ResConv-down-2	16x16x128
ResConv-down-2	8x8x256
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	1

Table 14. Discriminator on Clevr-Simple and -Complex.

Layer	Out Shape
ResConv-down-2	256x256x64
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x64
ResConv-down-2	16x16x128
ResConv-down-2	8x8x256
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	10

Table 15. Recognizer Q on Clevr-Simple and -Complex.

Layer	Out Shape
Const	4x4x512
ResConv-up-1	8x8x512
SC-4-5	8x8x512
ResConv-id-1	8x8x512
Noise-2	8x8x512
ResConv-up-1	16x16x512
SC-4-5	16x16x512
ResConv-id-1	16x16x512
Noise-2	16x16x512
ResConv-up-1	32x32x512
SC-4-5	32x32x512
ResConv-id-1	32x32x512
Noise-2	32x32x256
ResConv-up-1	64x64x256
SC-4-5	64x64x256
ResConv-id-1	64x64x256
Noise-2	64x64x256
ResConv-up-1	128x128x256
SC-4-5	128x128x256
ResConv-id-1	128x128x128
Noise-2	128x128x128
ResConv-up-1	256x256x128
SC-4-5	256x256x128
ResConv-id-1	256x256x128
Noise-2	256x256x64
ResConv-id-1	256x256x64
ResConv-up-1	512x512x64
ResConv-id-2	512x512x64
ResConv-id-1	512x512x3

Table 16. Generator on FFHQ.

Layer	Out Shape
ResConv-down-2	256x256x64
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x128
ResConv-down-2	16x16x256
ResConv-down-2	8x8x512
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	1

Table 17. Discriminator on FFHQ.

Layer	Out Shape
ResConv-down-2	256x256x64
ResConv-down-2	128x128x64
ResConv-down-2	64x64x64
ResConv-down-2	32x32x128
ResConv-down-2	16x16x256
ResConv-down-2	8x8x512
ResConv-down-2	4x4x512
Conv-id-1	4x4x512
Dense-1	30

Table 18. Recognizer Q on FFHQ.

Layer	Out Shape
Const	4x4x512
AdaIN-1	4x4x128
Conv-id-1	4x4x128
AdaIN-1	4x4x128
Conv-id-1	4x4x64
AdaIN-1	4x4x64
Conv-up-1	8x8x64
AdaIN-1	8x8x64
Conv-id-1	8x8x32
AdaIN-1	8x8x32
Conv-id-1	8x8x32
AdaIN-1	8x8x32
Conv-up-1	16x16x16
AdaIN-1	16x16x16
Conv-id-1	16x16x16
AdaIN-1	16x16x16
Conv-id-1	16x16x16
AdaIN-1	16x16x16
Conv-up-1	32x32x16
Conv-id-1	32x32x16
Conv-up-1	64x64x16
Conv-id-1	64x64x1

Table 19. Generator on DSprites.

Layer	Out Shape
ResConv-down-2	64x64x16
ResConv-down-2	32x32x16
ResConv-down-2	16x16x32
ResConv-down-2	8x8x64
ResConv-down-2	4x4x128
Conv-id-1	4x4x128
Dense-1	1

Table 20. Discriminator on DSprites.

Layer	Out Shape
ResConv-down-2	64x64x16
ResConv-down-2	32x32x16
ResConv-down-2	16x16x16
ResConv-down-2	8x8x16
ResConv-down-2	4x4x32
Conv-id-1	4x4x32
Dense-1	9

Table 21. Recognizer Q on DSprites.

Layer	Out Shape
Const	4x4x512
Conv-up-1	8x8x256
SC-3-2	8x8x256
ResConv-id-1	8x8x256
SC-4-3	8x8x256
ResConv-id-1	8x8x128
Conv-up-1	16x16x128
SC-4-3	16x16x128
ResConv-id-1	16x16x64
SC-4-3	16x16x64
ResConv-id-1	16x16x64
Conv-up-1	32x32x64
ResConv-id-1	32x32x32
Conv-up-1	64x64x32
ResConv-id-1	64x64x32
Conv-id-1	64x64x3

Table 22. Generator on 3DShapes.

Layer	Out Shape
ResConv-down-2	64x64x32
ResConv-down-2	32x32x32
ResConv-down-2	16x16x32
ResConv-down-2	8x8x64
ResConv-down-2	4x4x128
Conv-id-1	4x4x128
Dense-1	1

Table 23. Discriminator on 3DShapes.

Layer	Out Shape
ResConv-down-2	64x64x32
ResConv-down-2	32x32x32
ResConv-down-2	16x16x32
ResConv-down-2	8x8x64
ResConv-down-2	4x4x128
Conv-id-1	4x4x128
Dense-1	12

Table 24. Recognizer Q on 3DShapes.