StyleGAN

20181896 조유빈

StyleGAN

- **Abstract**

  - 본논문의 새로운 architecture는 자동으로 학습되고, 비지도의 높은 수준의 속성(e.g. pose and identity)과 생성된 이미지(e.g. 주근깨, 머리카락)의 stochastic variation을 separation하고, 직관적으로 scale-specific control이 가능하게 한다.

  - 본논문의 Generator는 traditional distribution quality metrics 측면에서 SOTA(state-of-the-art)를 향상시키고 더 나은 interpolation properties를 입증하며 latent factor of variation을 더 잘 disentangled하게 한다.

  - 매우 다양하고 고품질의 human faces 데이터셋(FFHQ)을 소개한다.

StyleGAN

- Latent vector z가 Generator의 입력으로 바로 들어가면 GAN은 latent space가 무조건 학습 데이터셋의 확률분포와 비슷한 형태로 만들어 지도록 학습하게 된다. 이렇게 되면 latent space가 entangle하게 만들어 진다.

- Entangle : 서로 얽혀 있는 상태여서 특징 구분이 어려운 상태. 즉, 각 특징들이 서로 얽혀 있어 구분이 어려움

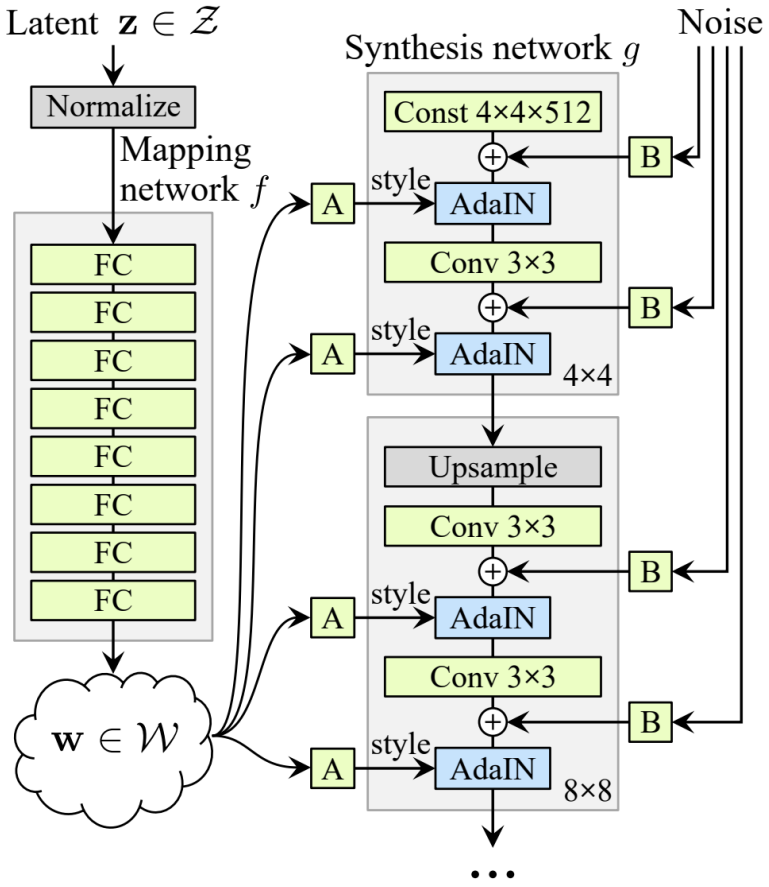
- Disentangle : 각 style들이 잘 구분되어 있는 상태. 어느 방향으로 가는 지에 따라 특징 A가 변할 수도 있고 특징 B가 변할 수도 있게 된다. 선형적으로 변수를 변경했을 때 어떤 결과물의 feature인지를 예측할 수 있는 상태이다.

(a) Distribution of features in training set

(b) Mapping from $\mathcal{Z}$ to features

(c) Mapping from $\mathcal{W}$ to features

StyleGAN



- 학습 데이터셋이 어떤 분포를 가지고 있을지 모르니 z를 바로 Generator에 넣지 않고 학습 데이터셋과 비슷한 확률 분포를 갖도록 non-linear mapping network를 사용해 w를 생성하여 w를 각 scale에 넣어서 학습시킨다.

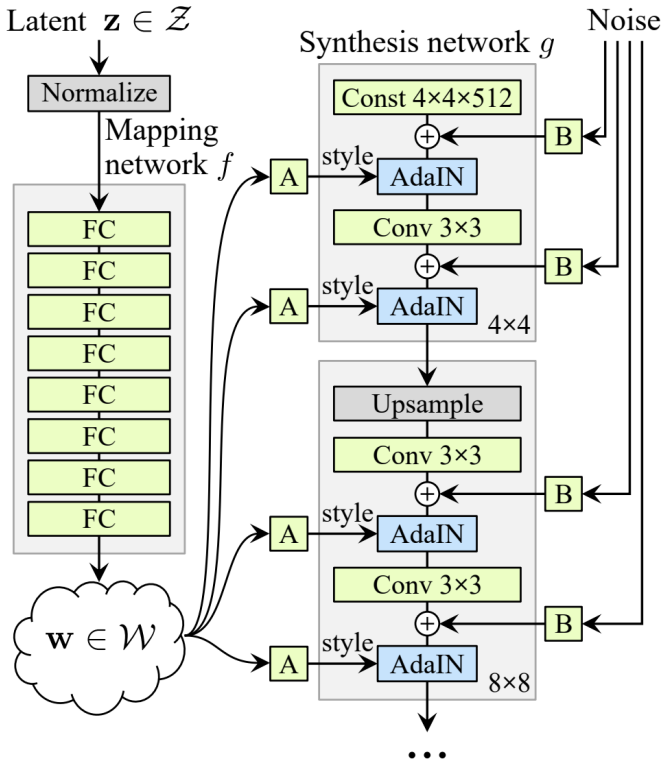
- W는 정확하지는 않지만 학습 데이터셋의 확률분포와 비슷한 모양으로 mapping된 상태이므로 특징이 disentangle하게 된다.

(a) Distribution of features in training set

(b) Mapping from $\mathcal{Z}$ to features

(c) Mapping from $\mathcal{W}$ to features

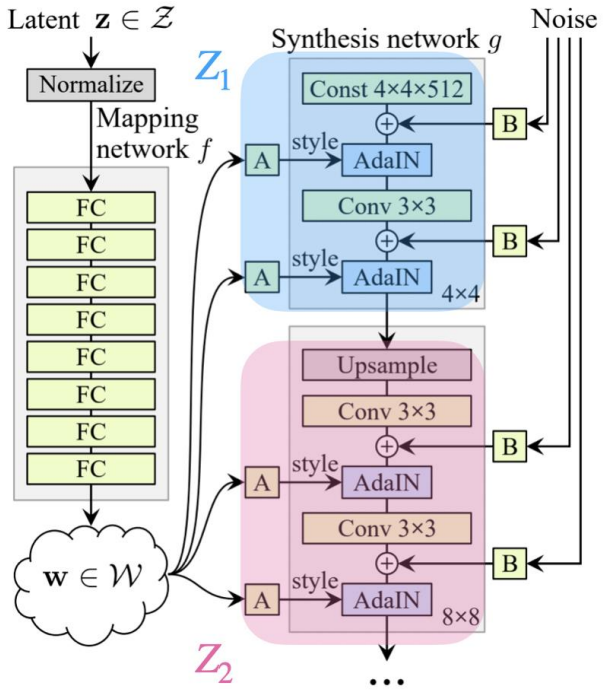
StyleGAN

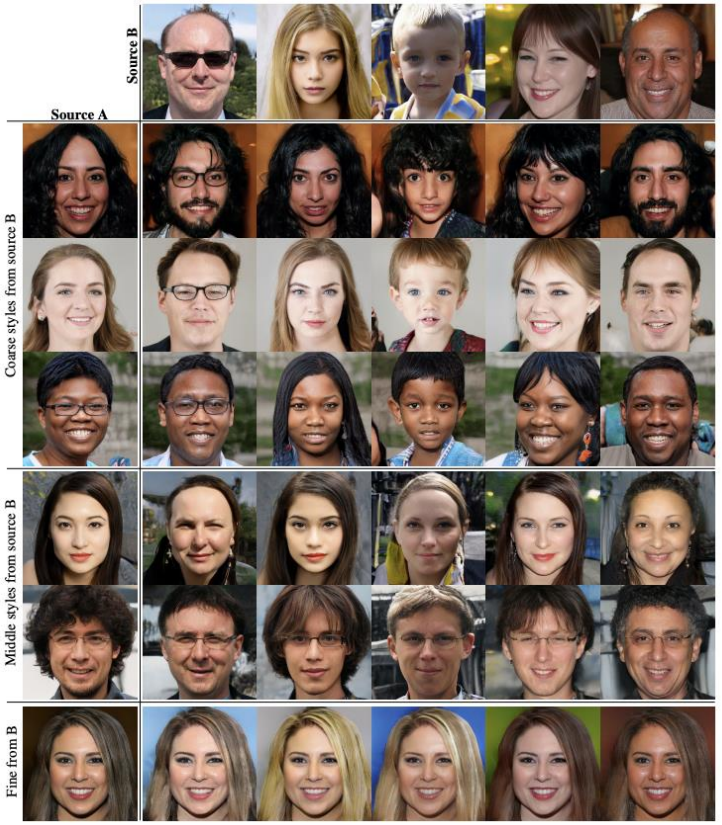


$$\mathrm{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i}, \qquad (1)$$

- "A"는 learned affine transform, "B"는 learned per-channel scaling factors를 의미한다.
- Neural Network에서 각 layer를 통과하며 scale, variance의 변화가 생기는 일이 발생하며 이는 학습이 불안정해지게 한다. 이를 해결하기 위해 normalization기법을 사용한다.
- 본논문에서는 adaptive instance normalization(AdaIN)를 각 Conv 뒤에 사용한다.
- W가 AdaIN을 통해 style을 입힐 때 shape가 안 맞으므로 "A"를 거쳐서 shape를 맞춰준다.
- y = (y_s, y_b)는 각 feature map x_i에 style을 입힌다.
- AdaIN 한번에 w 하나씩만 들어가므로 하나의 style이 각각의 scale에만 영향을 끼칠 수 있도록 분리해주는 효과를 갖는다. 따라서 style을 분리하는 방법으로 AdaIN이 효과적이다.

StyleGAN





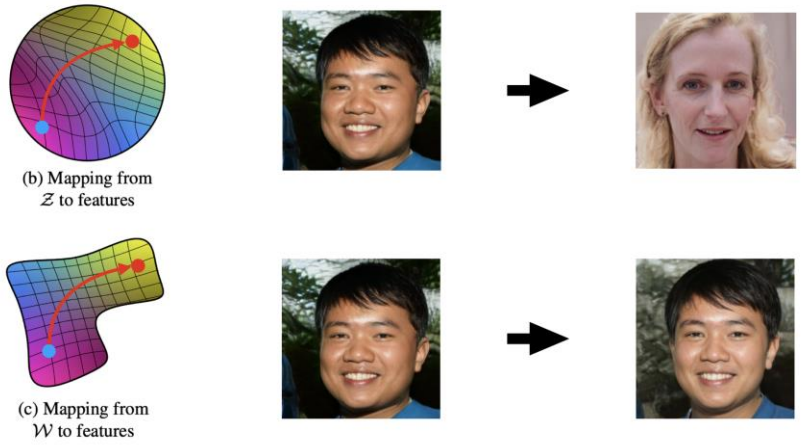
- 동일한 latent vector Z를 통해 만들어진 W만 계속 학습하다보면 correlation이 발생하게 될 수 있다.
- 따라서 latent space에서 뽑은 $Z_1, Z_2, \dots Z_n$을 통해 $W_1, W_2, \dots W_n$을 만든다.
- 왼쪽 그림과 같이 50:50 비율로 나눠서 각 layer에 적용해도 되고 다른 비율로 상황에 따라 적용해도 된다.
- 이러한 style mixing방법을 사용하면 다양한 style이 섞여서 synthesis network 학습이 된다.

StyleGAN

$$l_{\mathcal{Z}} = \mathbb{E}\left[\frac{1}{\epsilon^2} d\big(G(\text{slerp}(\mathbf{z}_1, \mathbf{z}_2; t)), G(\text{slerp}(\mathbf{z}_1, \mathbf{z}_2; t+\epsilon))\big)\right], \quad (2)$$

$$l_{\mathcal{W}} = \mathbb{E}\left[\frac{1}{\epsilon^2} d\big(g(\text{lerp}(f(\mathbf{z}_1), f(\mathbf{z}_2); t)), g(\text{lerp}(f(\mathbf{z}_1), f(\mathbf{z}_2); t+\epsilon))\big)\right], \quad (3)$$

slerp : spherical interpolation
lerp : linear interpolation (W는 not normalized이므로)



- Latent space가 disentangle하다는 것을 정량화 하기 위해 본 눈문에서는 두가지 measure를 제안한다.
- **Perceptual path length**
  - Latent space에서 interpolation했을 때 얼마나 큰 변화가 있는지 측정한다.
  - Interpolation을 했을 때 이미지에서 non-linear한 변화가 일어난다면 latent space가 entangle한 것이다.
  - Pretrained VGG16의 중간 레이어 feature map은 마지막 레이어보다 이미지에 대한 더 많은 정보들을 가지고 있어 embedding된 중간 레이어의 feature map을 사용하여 perceptual path length를 구하게 된다.
  - g(lerp($f(z_1), f(z_2); t$))와 g(lerp($f(z_1), f(z_2); t+\epsilon$))을 VGG16에 embedding시켜 pairwise image distance를 구한다. 이 값이 커지게 되면 latent space가 entangle한 것이고 작을 수록 disentangle한 것이다.

StyleGAN

- **Linear separability**
  - Latent space가 충분히 disentangle하다면 각각의 factors of variation에 해당하는 방향 벡터를 찾을 수 있어야 한다.
  - 따라서 linear hyperplane(linear SVM)을 통해 latent space points가 두 개의 개별 집합으로 얼마나 잘 분리할 수 있는지를 측정한다.
  - 40개의 특성을 갖는 데이터 셋으로 auxiliary classification network를 이진분류 학습을 시킨 후 Generator에서 생성된 20만장의 이미지를 pre-trained classifier에 넣어 분류한다. 이 중에서 신뢰도가 높은 10만장의 이미지를 SVM으로 이진분류를 하고, 예측된 class를 따져서 conditional entropy $H(Y|X)$를 구한다. 여기서 X는 SVM으로 예측한 class, Y는 pre trained classifier로 예측한 class이다.
  - 이를 통해 true class를 결정하기 위해 additional information이 얼마만큼 필요한지 알 수 있다.
  - Low value는 해당 factor of variation에 대한 일관성 있는 latent space 방향을 나타낸다.
  - 최종 separability score는 $\exp(\sum_i H(Y_i|X_i))$를 계산하여 구한다. i는 40.

StyleGAN

(a) Generated image (b) Stochastic variation (c) Standard deviation

Figure 5. Effect of noise inputs at different layers of our generator. (a) Noise is applied to all layers. (b) No noise. (c) Noise in fine layers only (64^2 – 1024^2). (d) Noise in coarse layers only (4^2 – 32^2). We can see that the artificial omission of noise leads to featureless "painterly" look. Coarse noise causes large-scale curling of hair and appearance of larger background features, while the fine noise brings out the finer curls of hair, finer background detail, and skin pores.

- **Stochastic variation**
  - 기존의 Generator는 z만 입력되기 때문에 미세한 특정 부분만을 변경할 수 없다는 특징이 있다.
  - 본 논문에서는 style을 담당하는 w을 AdaIN을 통해 입력해줄 뿐만 아니라 별도의 Noise가 입력된다.
  - 이는 랜덤하게 Noise를 입력시켜주면서 위의 그림과 같이 현재 style에서 화가가 리터칭 하듯이 인공지능이 리터칭 하는 것처럼 만들어 준다.

StyleGAN

- **Truncation trick in W**

  - Training data에서 low density 영역은 제대로 표현되지 않는 영역이기에 Generator가 이들을 학습 할 수 없고 유사한 이미지를 생성할 수 없다. (좋지 않은 이미지를 생성)

  - 이를 방지하기 위해 중간벡터 w를 잘라내어 평균 중간 벡터에 가깝게 유지하도록 한다.

  - Center of mass of W인 $\bar{w} = \mathbb{E}_{z \sim P(z)}[f(z)]$을 구하고 새 이미지를 생성 할 때 mapping network출력 w를 직접 사용하는 대신 $w' = \bar{w} + \psi(w - \bar{w})$를 사용한다. 여기서 ψ는 이미지가 평균 이미지에서 얼마나 멀리 떨어져 있을 것인지를 의미한다.

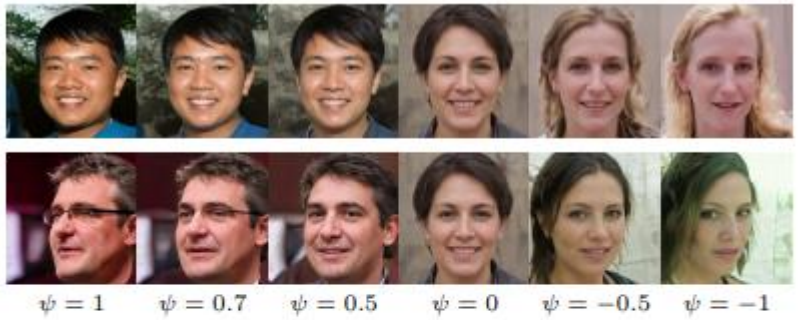


Figure 8. The effect of truncation trick as a function of style scale ψ. When we fade $\psi \to 0$, all faces converge to the "mean" face of FFHQ. This face is similar for all trained networks, and the interpolation towards it never seems to cause artifacts. By applying negative scaling to styles, we get the corresponding opposite or "anti-face". It is interesting that various high-level attributes often flip between the opposites, including viewpoint, glasses, age, coloring, hair length, and often gender.

StyleGAN

Figure 1. Instance normalization causes water droplet -like artifacts in StyleGAN images. These are not always obvious in the generated images, but if we look at the activations inside the generator network, the problem is always there, in all feature maps starting from the 64x64 resolution. It is a systemic problem that plagues all StyleGAN images.

- **한계점**
  - 생성된 이미지들에서 물방울 형태의 blob들이 관측된다.
  - 이는 StyleGAN2에 언급되어 이를 해결하는 방안이 StyleGAN2에서 소개된다.