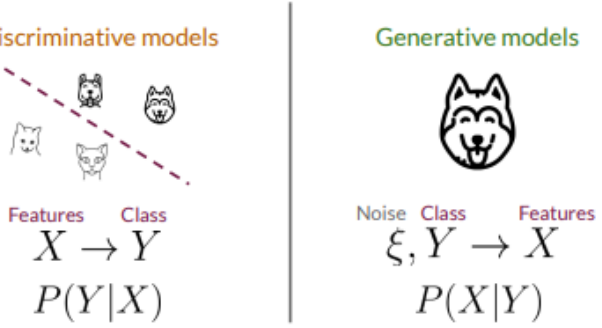
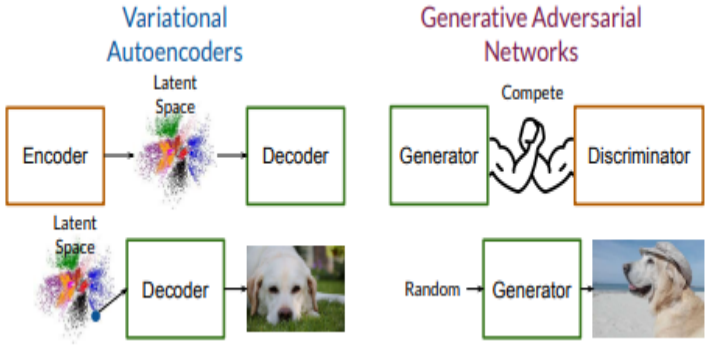
Generative Models

Generative Models vs. Discriminative Models



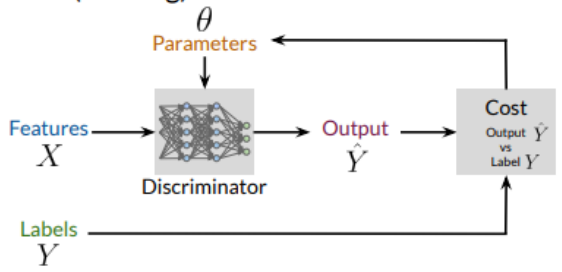
Generative Models



- Generative models learn to produce examples
- Discriminative models distinguish between classes

Discriminator

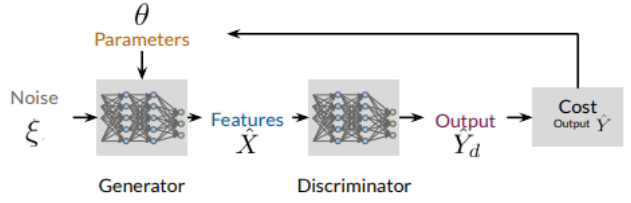
Classifiers (training)

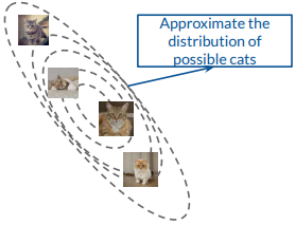


- The Discriminator is a classifier
- It learns the probability of class Y(real or fake) given features X
- The Probabilities are the feedback for the generator

Generator

Generator: Learning





- The generator produces fake data.
- It learns the probability of features X.
- The generator takes as input noise (random features)

BCE Cost Function

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log h(x^{(i)},\theta) + (1-y^{(i)})\log(1-h(x^{(i)},\theta))\right]$$

$y^{(i)}$	$h(x^{(i)},\theta)$	$y^{(i)}\log h(x^{(i)},\theta)$
0	any	0
1	0.99	~0
1	~0	-inf

Relevant when the label is 1

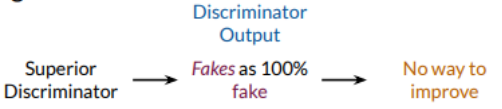
$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log h(x^{(i)},\theta) + (1-y^{(i)})\log(1-h(x^{(i)},\theta))\right]$$

$y^{(i)}$	$h(x^{(i)},\theta)$	$(1-y^{(i)})\log(1-h(x^{(i)},\theta))$
1	any	0
0	0.01	~0
0	~1	-inf

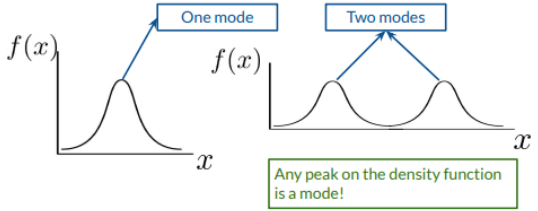
Relevant when the label is 0

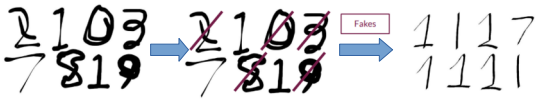
- Close to zero when the label and the prediction are similar
- Approaches infinity when the label and the prediction are different

Training GANs



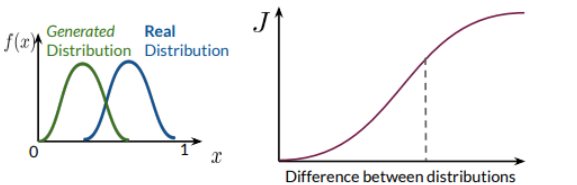
Mode Collapse

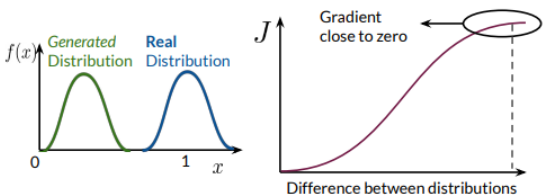




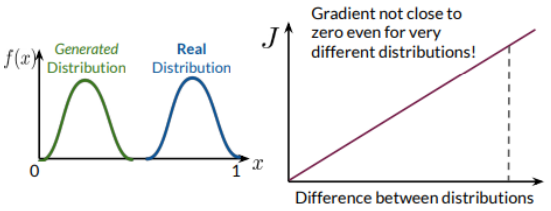
- Modes are peaks in the distribution of features
- Mode collapse happens when the generator gets stuck in one mode

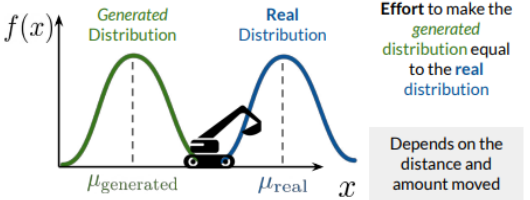
Problem with BCE Loss



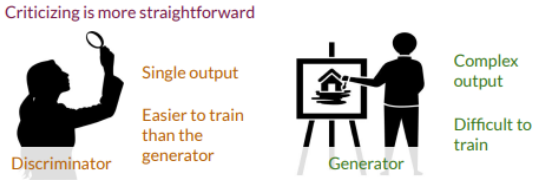


Earth Mover's Distance



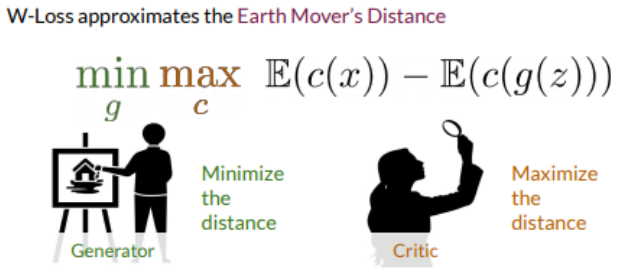


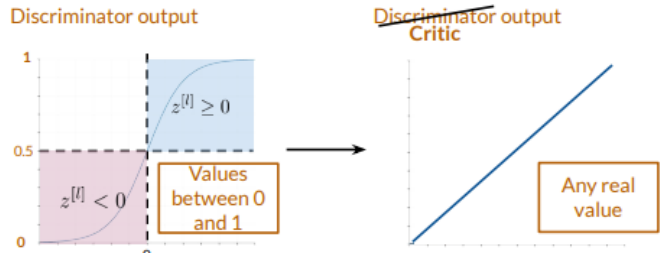
- Earth mover's distance (EMD) is a function of amount and distance
- Doesn't have flat regions when the distributions are very
- Approximating EMD solves the problems associated with BCE



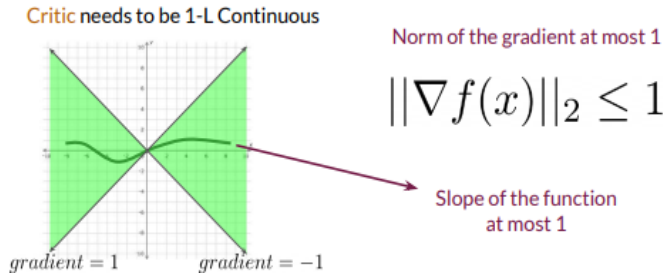
- GANs try to make the real and generated distributions look similar
- When the discriminator improves too much, the function approximated by BCE Loss will contain flat regions
- Flat regions on the cost function = vanishing gradients

Wasserstein Loss





- W-Loss looks very similar to BCE Loss
- W-Loss prevents mode collapse and vanishing gradient problems

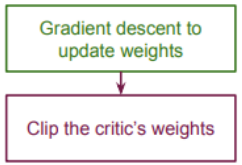


- Critic's neural network needs to be 1-L Continuous when using W-Loss
- This condition ensures that W-Loss is validly approximating Earth Mover's Distance

1-Lipschitz Continuity Enforcement

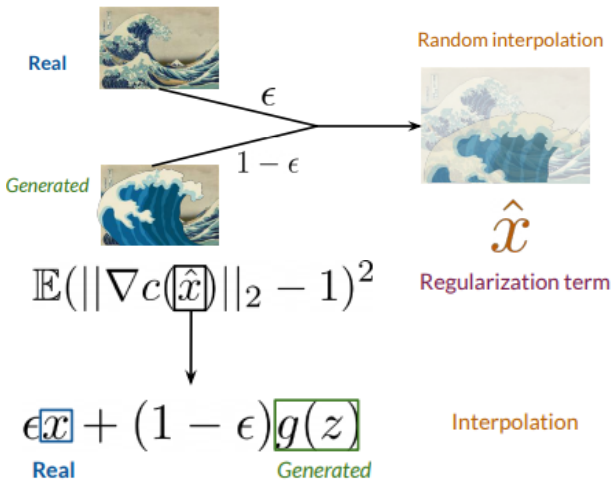
1-L Enforcement: Weight Clipping

Weight clipping forces the weights of the critic to a fixed interval



Limits the learning ability of the critic

1-L Enforcement: Gradient Penalty



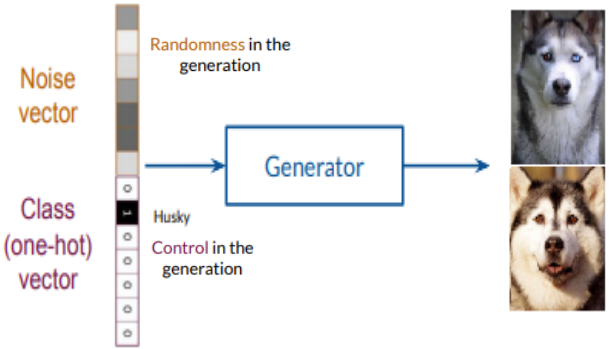
Putting It All Together

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z))) + \lambda \mathbb{E}(||\nabla c(\hat{x})||_2 - 1)^2$$

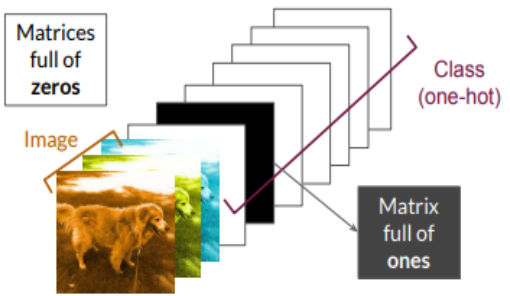
ConditionalGeneration

Conditional	Unconditional
Examples from **the classes you want**	Examples from *random classes*
Training dataset needs to be **labeled**	Training dataset *doesn't need to be labeled*

Generator Input

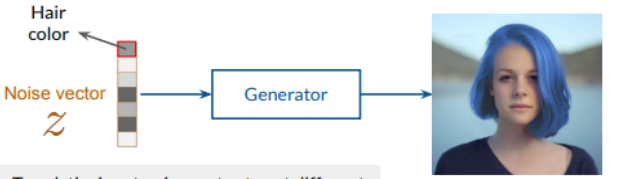


Discriminator Input



- The class is passed to the generator as one-hot vectors
- The class is passed to the discriminator as one-hot matrices
- The size of the vector and the number of matrices represent the number of classes

ControllableGeneration

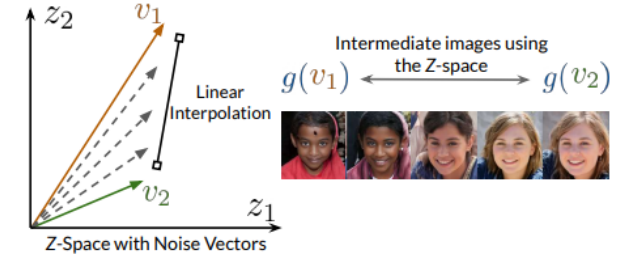


Tweak the input noise vector to get different features on the output

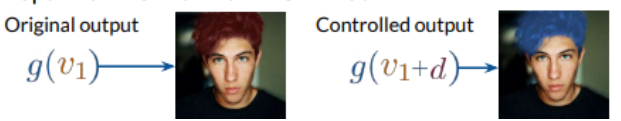
Controllable	Conditional
Examples with the **features that you want**	Examples from *the classes you want*
Training dataset **doesn't need to be labeled**	Training dataset *needs to be labeled*
Manipulate the z vector input	*Append a class vector* to the input

Z-Space

Interpolation Using the Z-Space

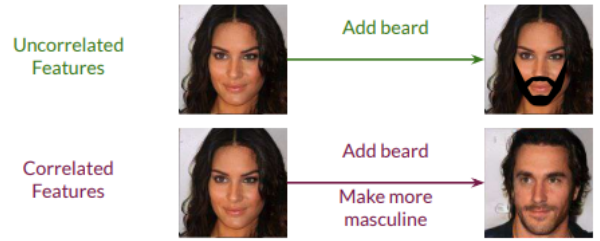


Z-Space and Controllable Generation



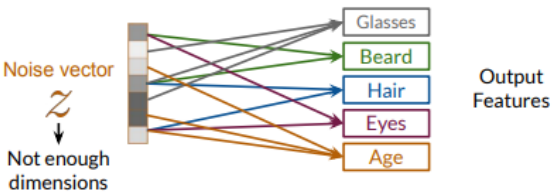
Challenges

Feature Correlation



- When trying to control one feature, others that are correlated change

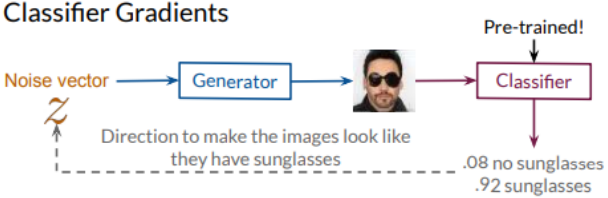
Z-Space Entanglement



It is not possible to control single output features

- Entanglement happens when z does not have enough dimensions

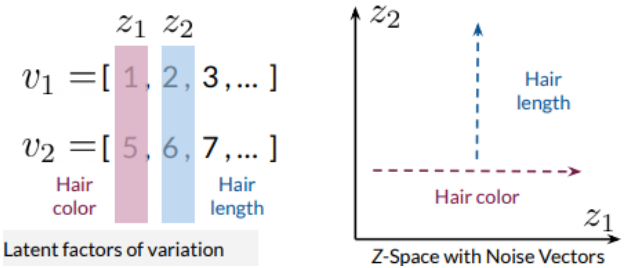
Classifier Gradients

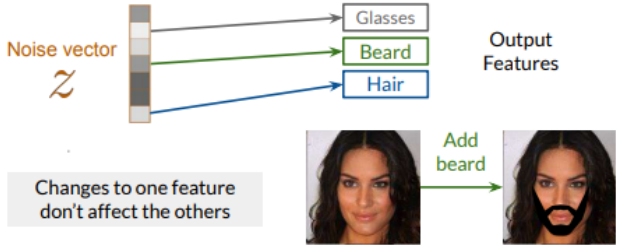


Modify **just** the noise vector until the feature emerges

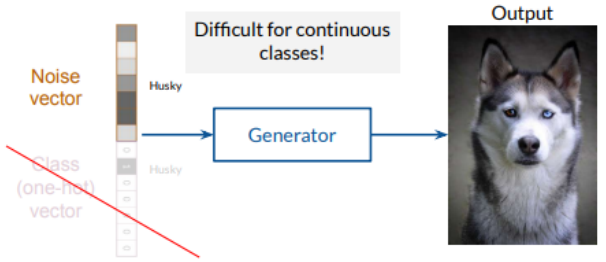
- Classifiers can be used to find directions in the Z-space
- To find directions, the updates are done just to the noise vector

Disentanglement

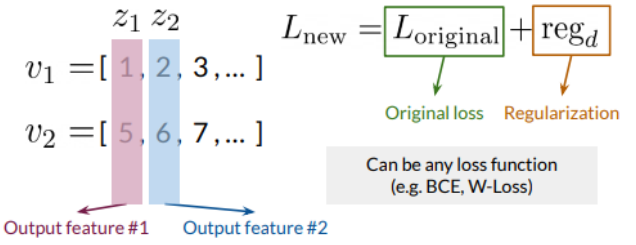




Encourage Disentanglement: Supervision



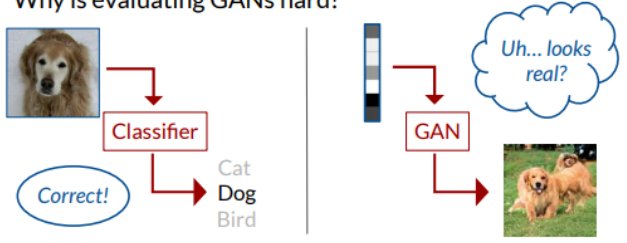
Encourage Disentanglement: Loss Function



- Disentangled Z-spaces let you control individual features by corresponding z values directly to them
- There are supervised and unsupervised methods to achieve disentanglement

Evaluation

Why is evaluating GANs hard?



- No ground-truth = challenging to evaluate
- Fidelity measures image quality and diversity measures variety
- Evaluation metrics try to quantify fidelity & diversity

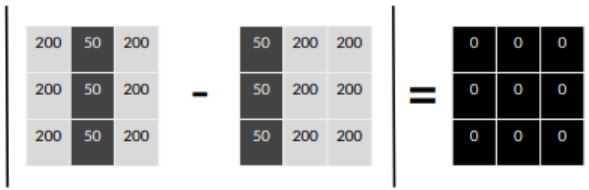
Two Important Properties

Fidelity: quality of images

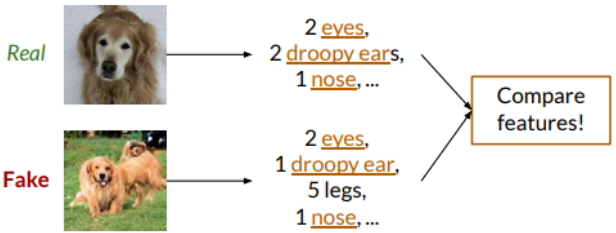
Diversity: variety of images

Comparing Images

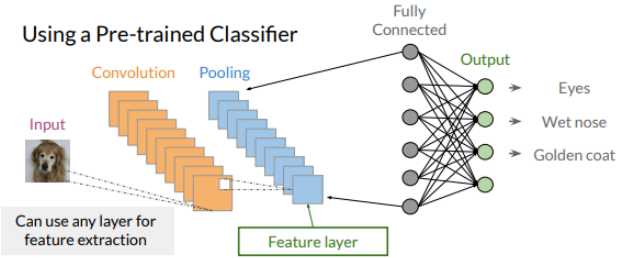
Pixel Distance



Feature Distance



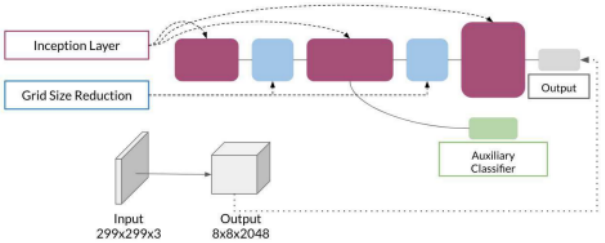
Feature Extraction

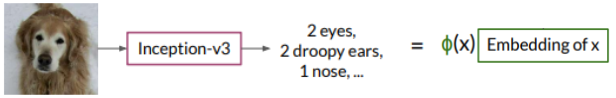


- Classifiers can be used as feature extractors by cutting the network at earlier layers
- The last pooling layer is most commonly used for feature extraction
- Best to use classifiers that have been trained on large datasets ImageNet

Inception-v3 Embeddings

Inception-v3 Architecture

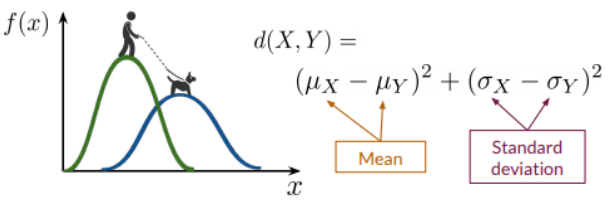




Comparing Embeddings

Fréchet Inception Distance (FID)

Fréchet Distance Between Normal Distributions



Univariate Normal Fréchet Distance =

$$(\mu_X - \mu_Y)^2 + (\sigma_X^2 + \sigma_Y^2 - 2\sigma_X\sigma_Y)$$

Lower FID = closer distributions

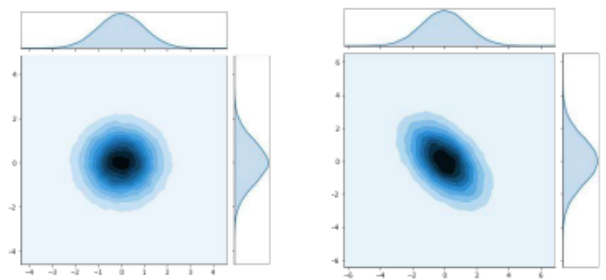
Multivariate Normal Fréchet Distance =

$$\|\mu_X - \mu_Y\|^2 + \mathrm{Tr}\left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X\Sigma_Y}\right)$$

Real and fake embeddings are two multivariate normal distributions

Use *large sample size* to reduce noise

Multivariate Normal Distributions



$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

all dimensions are *independent* — dimensions covary

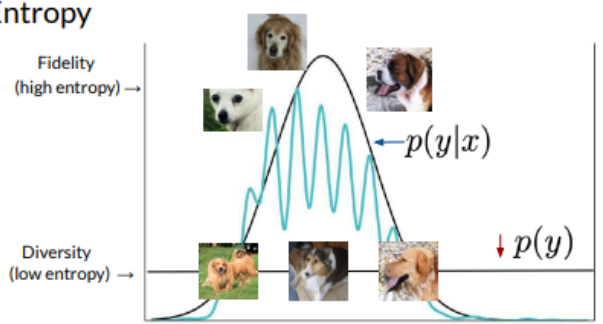
- FID calculates the difference between reals and fakes
- FID uses the Inception model and multivariate normal Fréchet distance
- Sample size needs to be large for FID to work well

Shortcomings of FID

- Uses pre-trained Inception model, which may not capture all features
- Needs a large sample size
- Slow to run
- Limited statistics used: only mean and covariance

Inception Score

Entropy



KL Divergence

$$D_{KL}(p(y|x)\|p(y)) = p(y|x)\log\left(\frac{p(y|x)}{p(y)}\right)$$

Conditional distribution (fidelity)

Marginal distribution (diversity)

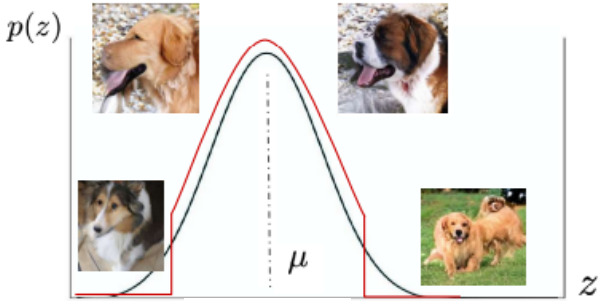
$$\text{IS} = \exp(\mathbb{E}_{x\sim p_g} D_{KL}(p(y \mid x)\|p(y)))$$

KL Divergence

- Inception Score tries to capture fidelity & diversity
- Inception Score has many shortcomings
  - Can be gamed too easily
  - Only looks at fake images, not reals
  - ImageNet doesn't teach a model all features
- Worse than Fréchet Inception Distance

Sampling and Truncation

Sampling Fakes



Truncation Trick

- Fakes are sampled using the training or prior distribution of z
- Truncate more for higher fidelity, lower diversity
- Human evaluation is still necessary for sampling
- HYPE(Human eYe Perceptual Evaluation)

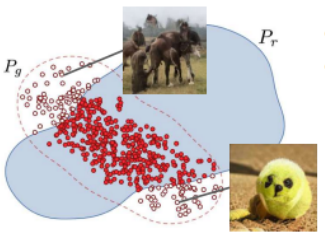
HYPE and Human Evaluation

- Crowdsourced evaluation from Amazon Mechanical Turk
- $\text{HYPE}_{\text{time}}$ measures time-limited perceptual thresholds
- HYPE_{∞} measures error rate on a percentage of images
- Ultimately, evaluation depends on the type of downstream task

Precision and Recall

Precision

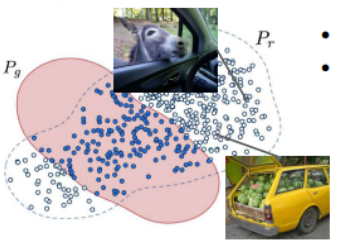
- Use truncation trick to improve precision



- Relates to fidelity
- Looks at overlap between reals and fakes, over how much extra gunk the generator produces (non-overlap red)

Recall

- Models tend to be better at recall



- Relates to diversity
- Looks at overlap between reals and fakes, over all the reals that the generator cannot model (non-overlap blue)

Alternatives to GANs

- VAEs have the opposite pros/cons as GANs
  - Often lower fidelity results
  - Density estimation, inversion, stable training

Autoregressive Models

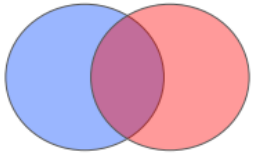
Relies on previous pixels to generate next pixel

Left: source image Right: new portraits generated from high-level latent representation

Flow Models

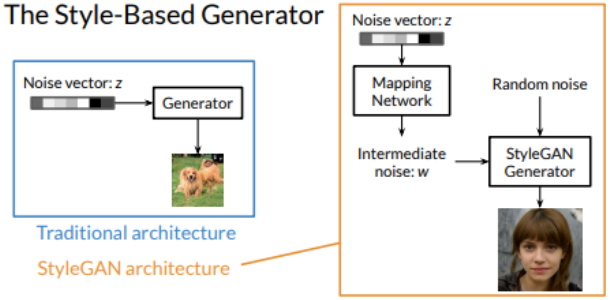
Uses invertible mappings

Hybrid Models

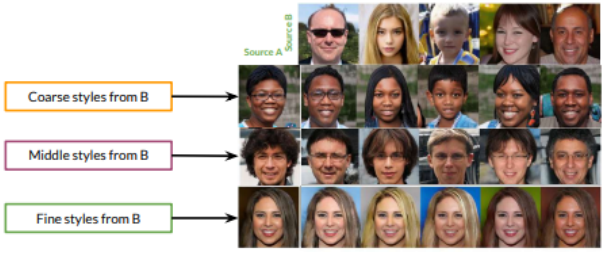


StyleGAN

The Style-Based Generator



Style Mixing



Stochastic Variation

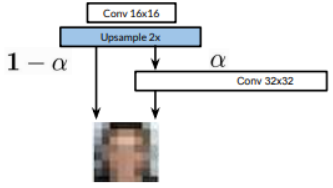
Small details: hair strands, wrinkles, etc.

Different extra noise values create stochastic variation

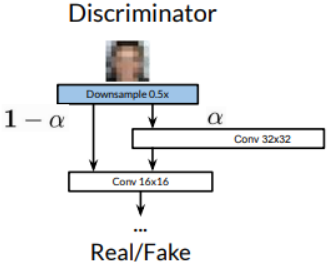
1. Sample noise from Normal distribution
2. Concatenate noise to **x**, before AdaIN

- Style mixing increases diversity that the model sees during training
- Stochastic noise causes small variations to output
- Coarse or fineness depends where in the network style or noise is added
  - Earlier for coarser variation
  - Later for finer variation

Progressive Growing

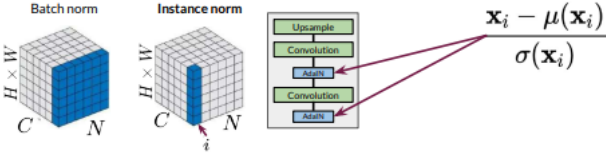


Discriminator

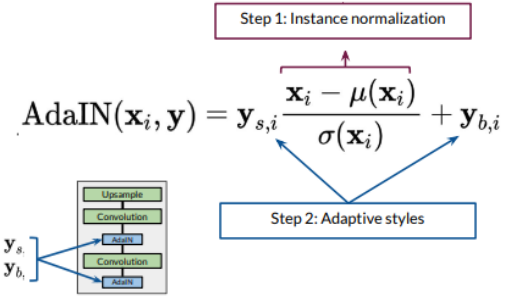


- Progressive growing gradually doubles image resolution
- Helps with faster, more stable training for higher resolutions

AdaIN



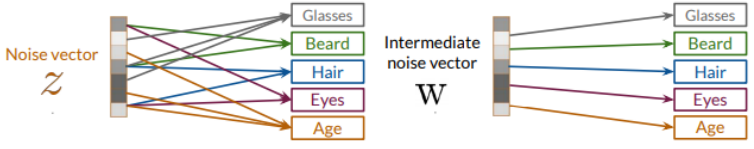
Step 1: Normalize convolution outputs using **Instance Normalization**



Step 2: Apply adaptive styles **using the intermediate noise vector**

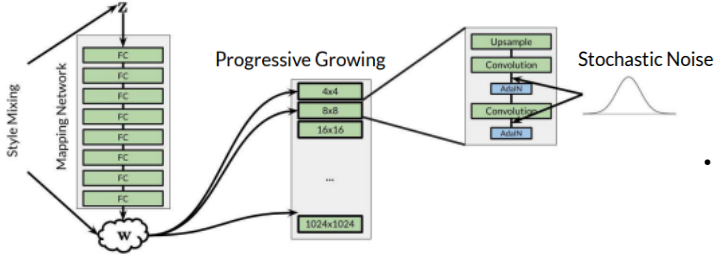
- AdaIN transfers style information onto the generated image from the intermediate noise vector W
- Instance Normalization is used to normalize individual examples before apply style statistics from W

Noise Mapping Network



- Noise mapping allows for a more disentangled noise space
- The intermediate noise vector is used as input to the generator

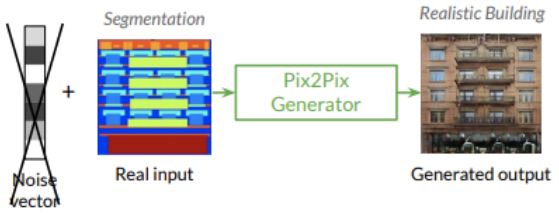
StyleGAN Architecture



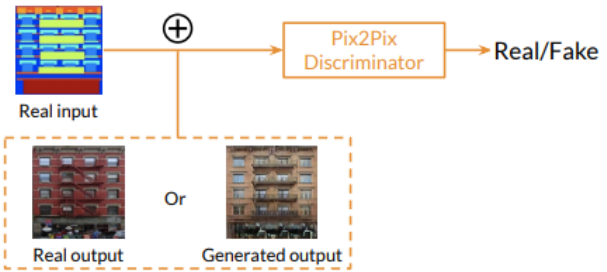
- Main components of StyleGAN:
  - Progressive Growing
  - Noise Mapping Network
  - AdaIN
  - Style Mixing
  - Stochastic Noise
- StyleGAN Goals:
  - Greater fidelity on high-resolution images
  - Increased diversity of outputs
  - More control over image features

Pix2Pix

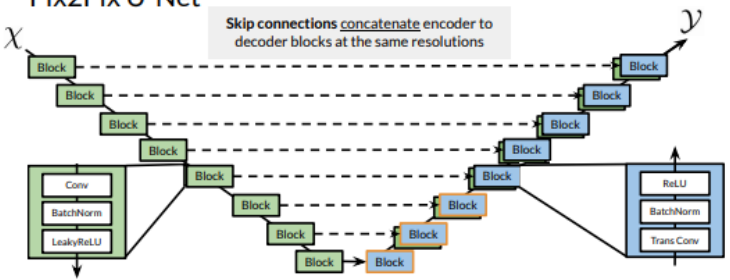
Pix2Pix Generator



Pix2Pix Discriminator



Pix2Pix U-Net

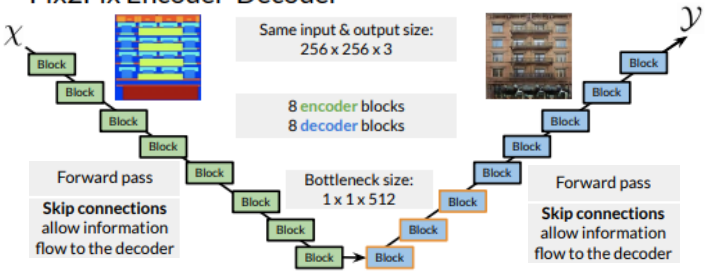


- Inputs and outputs are similar to a conditional GAN
  - Take in the original image, instead of the class vector
  - No explicit noise as input
- Generator and discriminator models are upgraded

Additional Loss Term

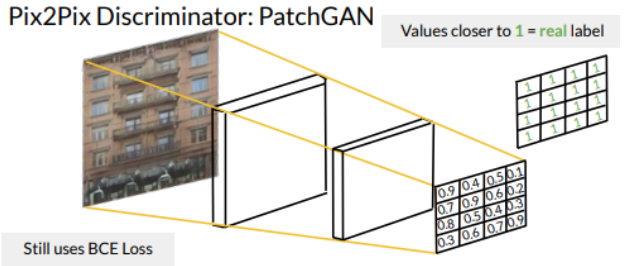
$$\min_g \max_c \text{Adversarial Loss} + \lambda * \text{Pixel loss term}$$

Pix2Pix Encoder-Decoder



- Pix2Pix's generator is a U-Net
- U-Net is an encoder-decoder, with same-size inputs and outputs
- U-Net uses skip connections
  - Skip connections help the decoder learn details from the encoder directly
  - Skip connections the encoder learn from more
  - gradients flowing from decoder

Pix2Pix Discriminator: PatchGAN



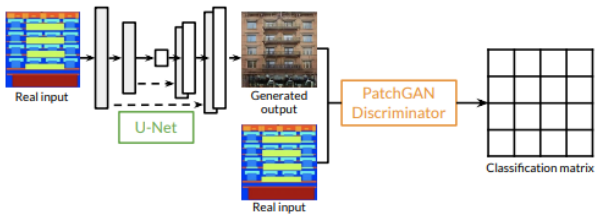
- PatchGAN discriminator outputs a matrix of values, each between 0 and 1
- Label matrices: 0's = fake, 1's = real

Pix2Pix Generator Loss

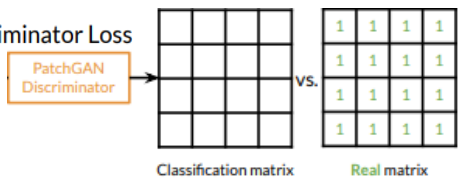
$$\text{BCE Loss} + \lambda \sum_{i=1}^{n} \left| \text{image} - \text{image} \right|$$

- Softly encourages the generator with this additional supervision
  - The target output labels are the supervision
  - Generator essentially "sees" these labels

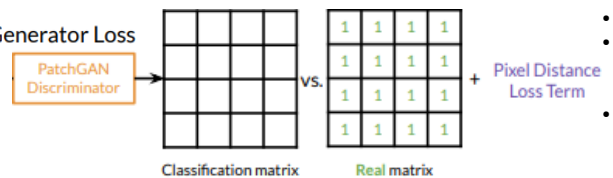
Pix2Pix



Discriminator Loss



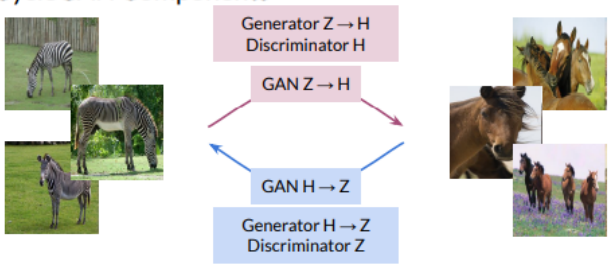
Generator Loss



- U-Net generator: image → image
- PatchGAN discriminator
  - Inputs input image and paired output (either real target or fake
  - Outputs classification matrix
- Generator loss has a regularization term

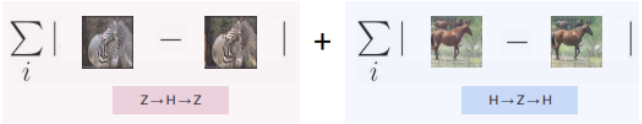
CycleGAN

CycleGAN Components



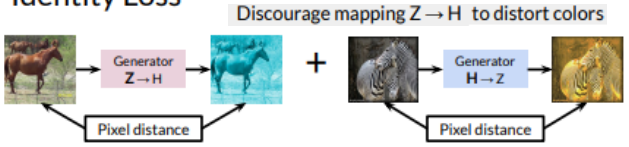
- Unpaired image-to-image translation: (this method is unsupervised)
  - Learns a mapping between two piles of images
  - Examines common elements of the two piles (content) and unique elements of each pile (style)
- CycleGAN uses two GANs for unpaired image-to-image translation
- The discriminators are PatchGAN
- The generators are similar to a U-Net and DCGAN generator with additional skip connections

Cycle Consistency Loss



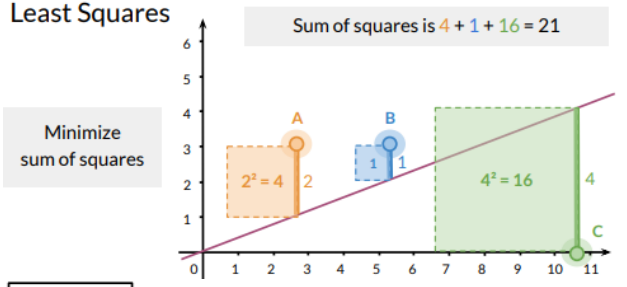
- Cycle consistency helps transfer uncommon style elements between the two GANs, while maintaining common content
- Add an extra loss term to each generator to softly encourage cycle consistency
- Cycle consistency is used in both directions

Identity Loss



- Identity Loss takes a real image in domain B and inputs it into Generator: A → B, expecting an identity mapping
  - the output is the same as the input
- Pixel distance is used
  - Ideally, no difference between input and output!
- Identity Loss is optionally added to help with color preservation

Least Squares



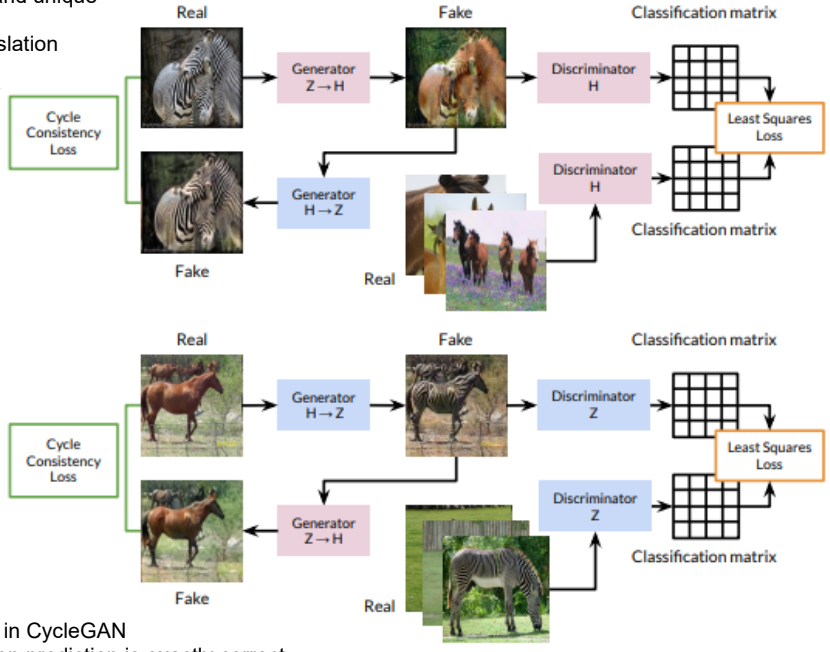
Discriminator Loss

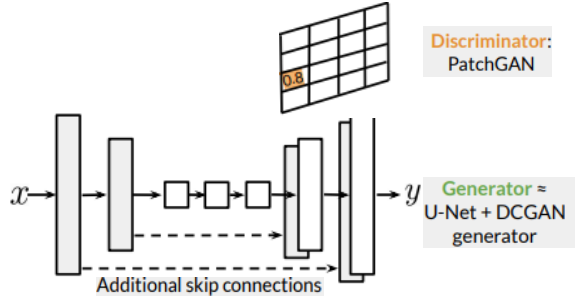
$$\mathbb{E}_{\boldsymbol{x}}\left[(D(\boldsymbol{x})-1)^2\right]+\mathbb{E}_{\boldsymbol{z}}\left[(D(G(\boldsymbol{z}))-0)^2\right]$$

Generator Loss

$$\mathbb{E}_{\boldsymbol{z}}\left[(D(G(\boldsymbol{z}))-1)^2\right]$$

- Least squares fits a line from several points
- Least Squares Loss is used as the Adversarial Loss function in CycleGAN
- More stable than BCELoss, since the gradient is only flat when prediction is exactly correct





Generator Loss

= Least Squares Adversarial Loss H → Z + Cycle Consistency Loss H → Z → H + Identity Loss H → Z

+ Least Squares Adversarial Loss Z → H + Cycle Consistency Loss Z → H → Z + Identity Loss Z → H

- Components
  - Two GANs
  - Cycle Consistency Loss
  - Least Squares Adversarial Loss
  - Identity Loss