

Generative Adversarial Networks (GANs)

From Ian Goodfellow et al.

A short tutorial by :-

Binglin, Shashank & Bhargav

Outline

- **Part 1: Introduction to GANs**
- **Part 2: Some challenges with GANs**
- **Part 3: Applications of GANs**

Part 1

- Motivation for Generative Models
- From Adversarial Training to GANs
- GAN's Architecture
- GAN's objective
- DCGANs

GANs

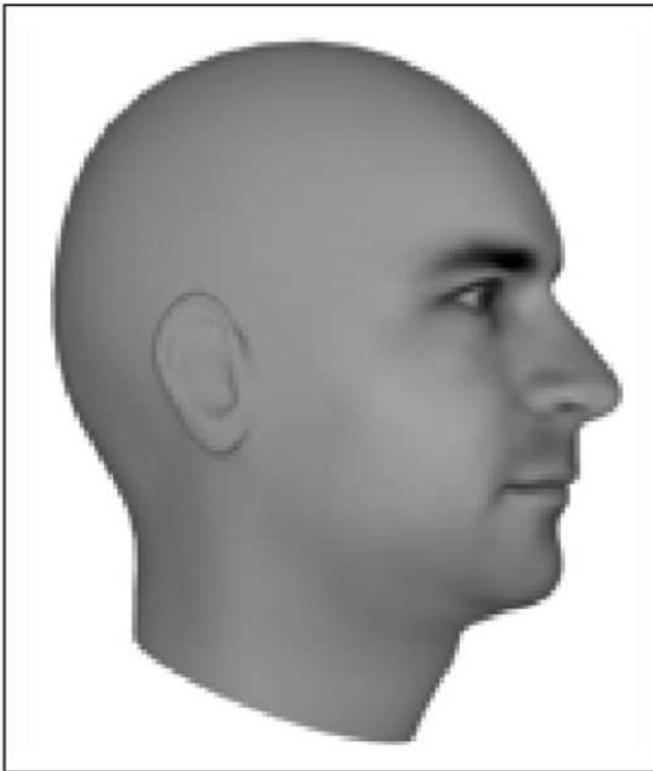
- **Generative**
 - Learn a generative model
- **Adversarial**
 - Trained in an adversarial setting
- **Networks**
 - Use Deep Neural Networks

Why Generative Models?

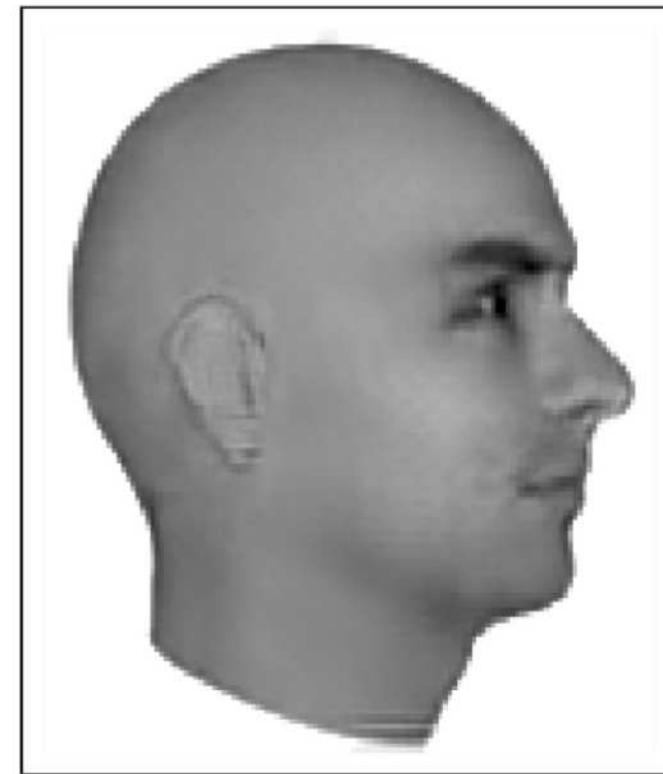
- **We've only seen discriminative models so far**
 - Given an image X , predict a label Y
 - Estimates $P(Y|X)$
- **Discriminative models have several key limitations**
 - Can't model $P(X)$, i.e. the probability of seeing a certain image
 - Thus, can't sample from $P(X)$, i.e. **can't generate new images**
- **Generative models (in general) cope with all of above**
 - Can model $P(X)$
 - Can generate new images

Magic of GANs...

Ground Truth



Adversarial



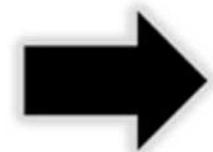
Magic of GANs...

Which one is Computer generated?

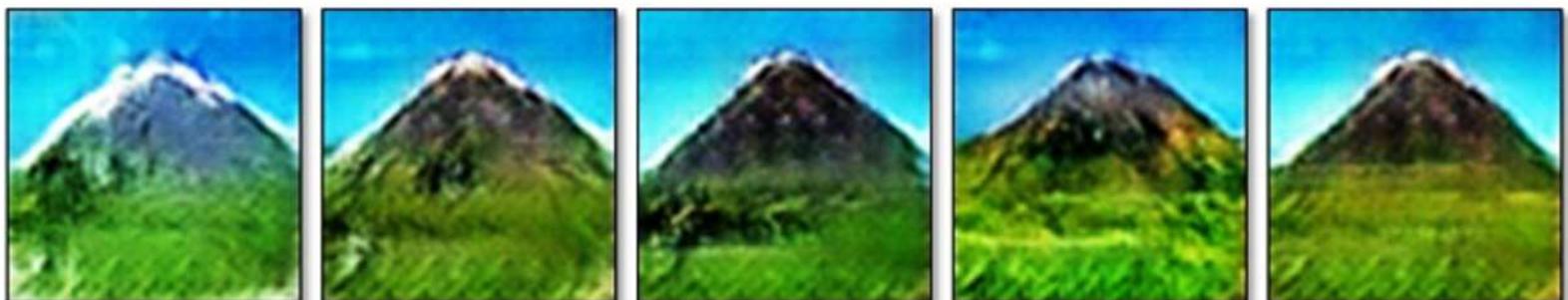


Magic of GANs...

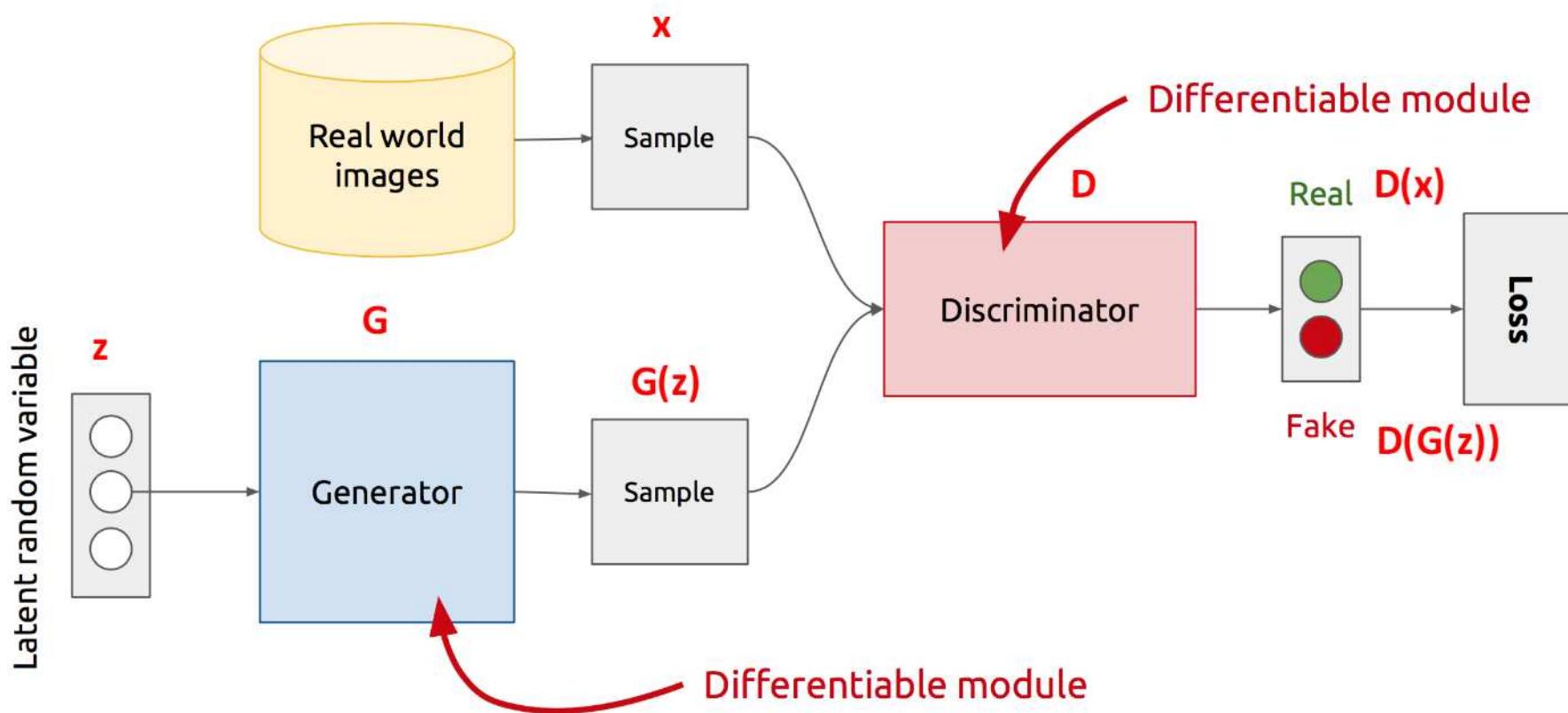
User edits



Generated images

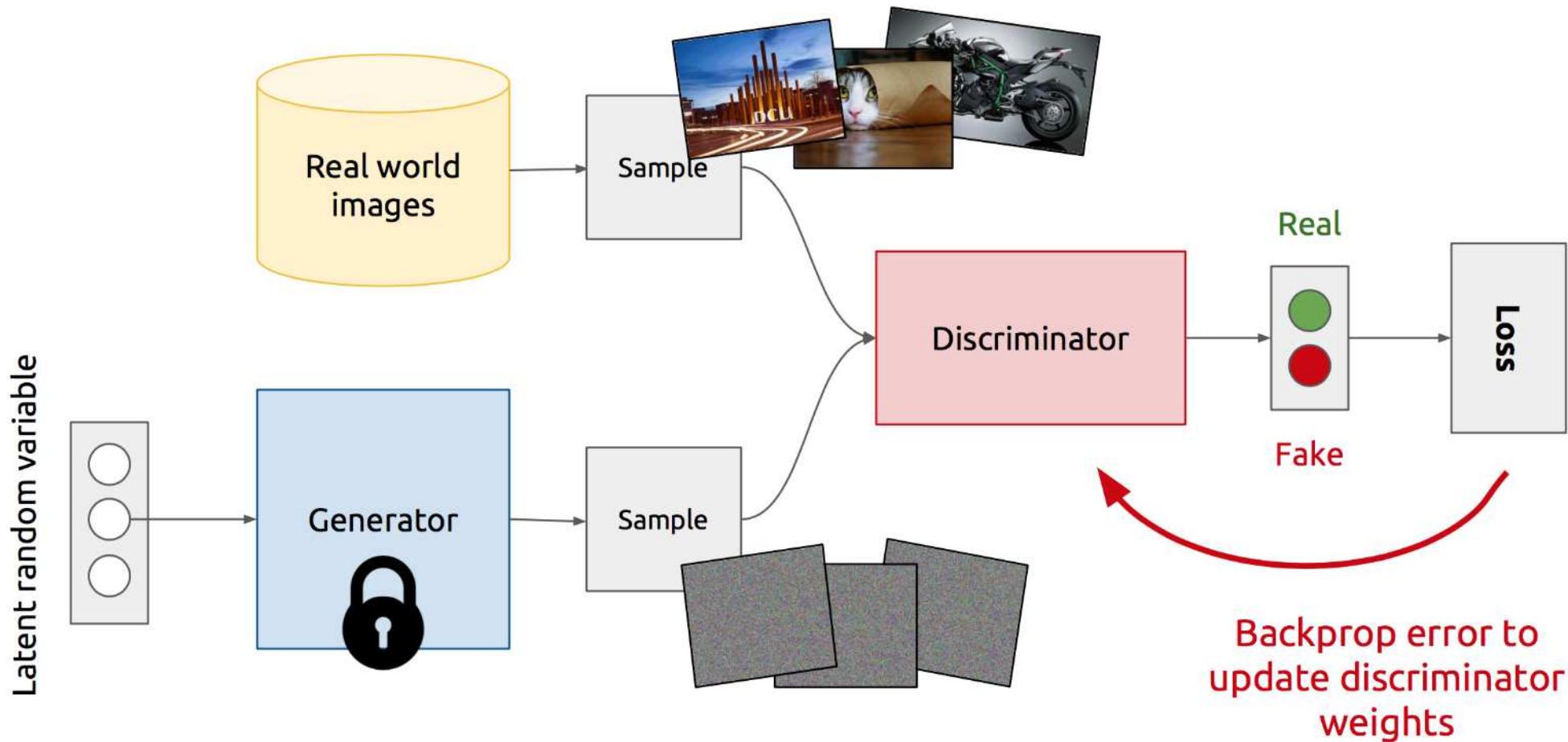


GAN's Architecture

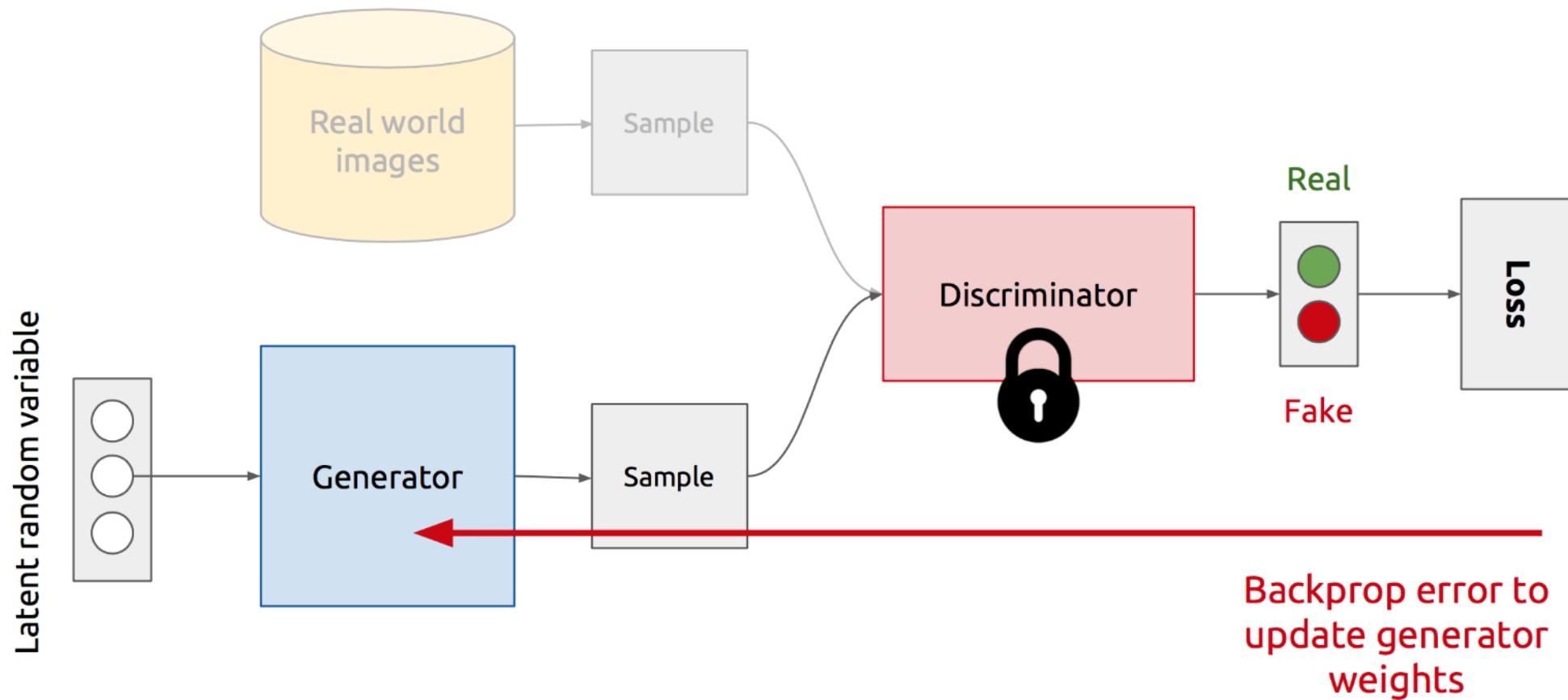


- \mathbf{Z} is some random noise (Gaussian/Uniform).
- \mathbf{Z} can be thought as the latent representation of the image.

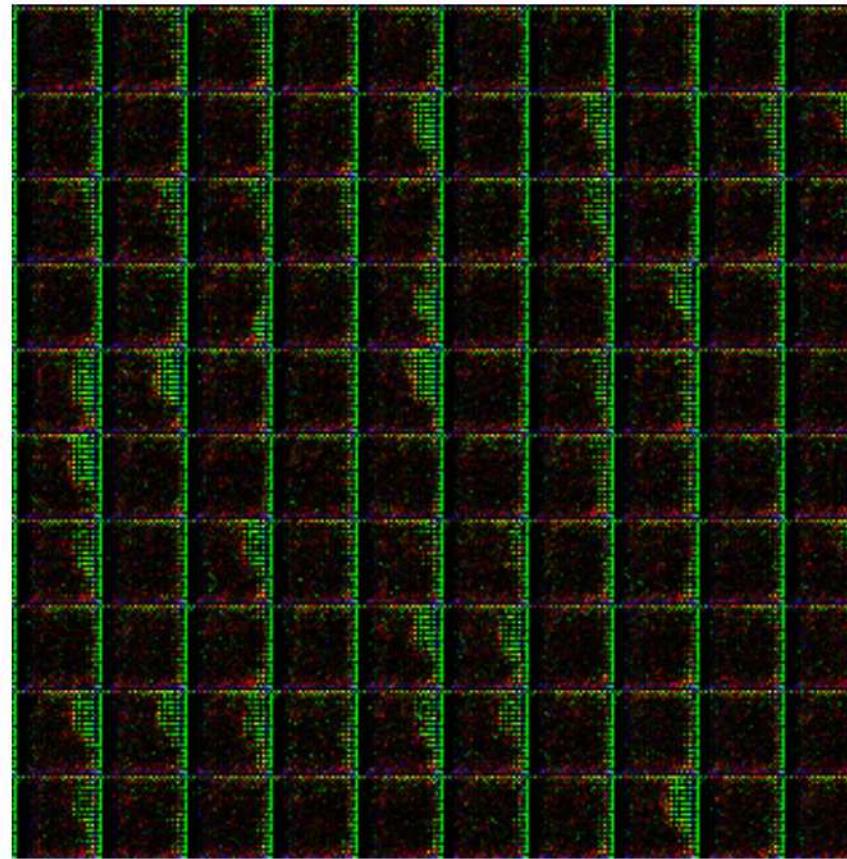
Training Discriminator



Training Generator



Generator in action



GAN's formulation

$$\min_G \max_D V(D, G)$$

- It is formulated as a **minimax game**, where:
 - The Discriminator is trying to maximize its reward $V(D, G)$
 - The Generator is trying to minimize Discriminator's reward (or maximize its loss)

$$V(D, G) = \boxed{\mathbb{E}_{x \sim p(x)} [\log D(x)]} + \boxed{\mathbb{E}_{z \sim q(z)} [\log(1 - D(G(z)))]}$$

- The Nash equilibrium of this particular game is achieved at:
 - $P_{data}(x) = P_{gen}(x) \quad \forall x$
 - $D(x) = \frac{1}{2} \quad \forall x$

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

Discriminator
updates

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(\mathbf{x}^{(i)}) + \log (1 - D(G(\mathbf{z}^{(i)}))) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(\mathbf{z}^{(i)}))).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Generator
updates

Vanishing gradient strikes back again...

$$V(D, G) = \mathbb{E}_{x \sim p(x)} [\log D(x)] + \boxed{\mathbb{E}_{z \sim q(z)} [\log(1 - D(G(z)))]}$$

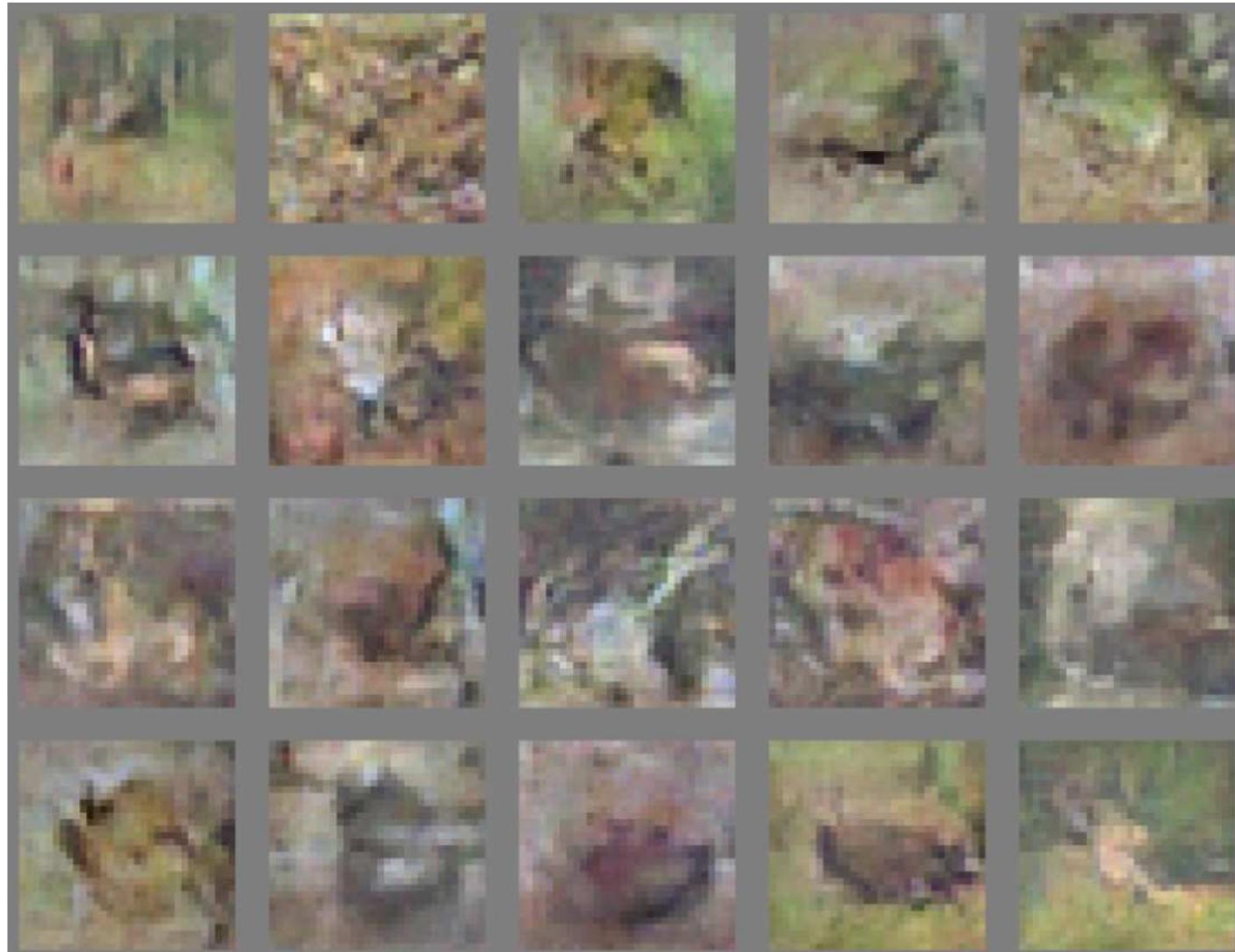
$$\nabla_{\theta_G} V(D, G) = \nabla_{\theta_G} \mathbb{E}_{z \sim q(z)} [\log(1 - D(G(z)))]$$

- $\nabla_a \log(1 - \sigma(a)) = \frac{-\nabla_a \sigma(a)}{1 - \sigma(a)} = \frac{-\sigma(a)(1 - \sigma(a))}{1 - \sigma(a)} = -\sigma(a) = -D(G(z))$
- Gradient goes to 0 if D is confident, i.e. $D(G(z)) \rightarrow 0$
- Minimize $-\mathbb{E}_{z \sim q(z)} [\log D(G(z))]$ for **Generator** instead (keep Discriminator as it is)

Faces



CIFAR



Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.

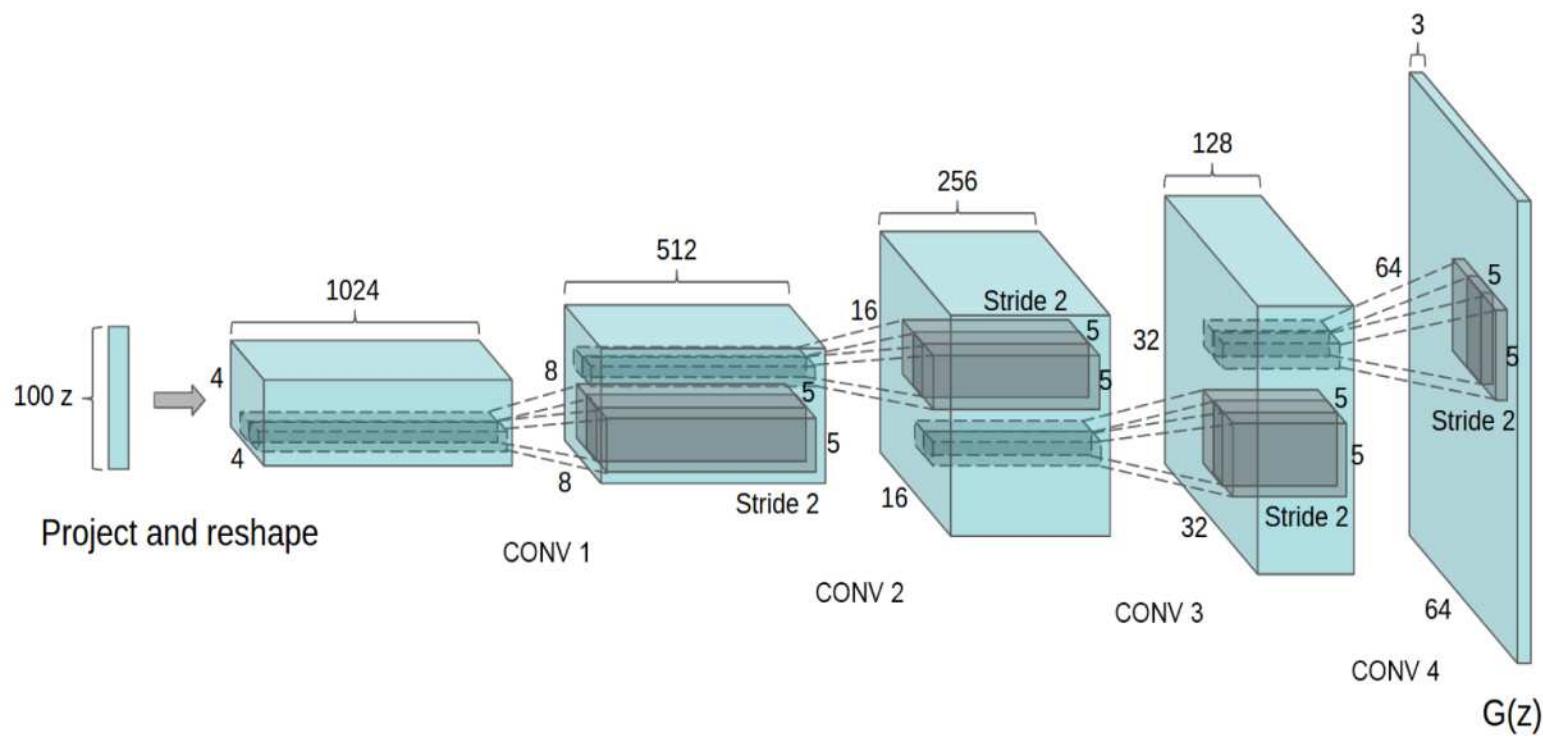
DCGAN: Bedroom images



Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv:1511.06434 (2015).

Deep Convolutional GANs (DCGANs)

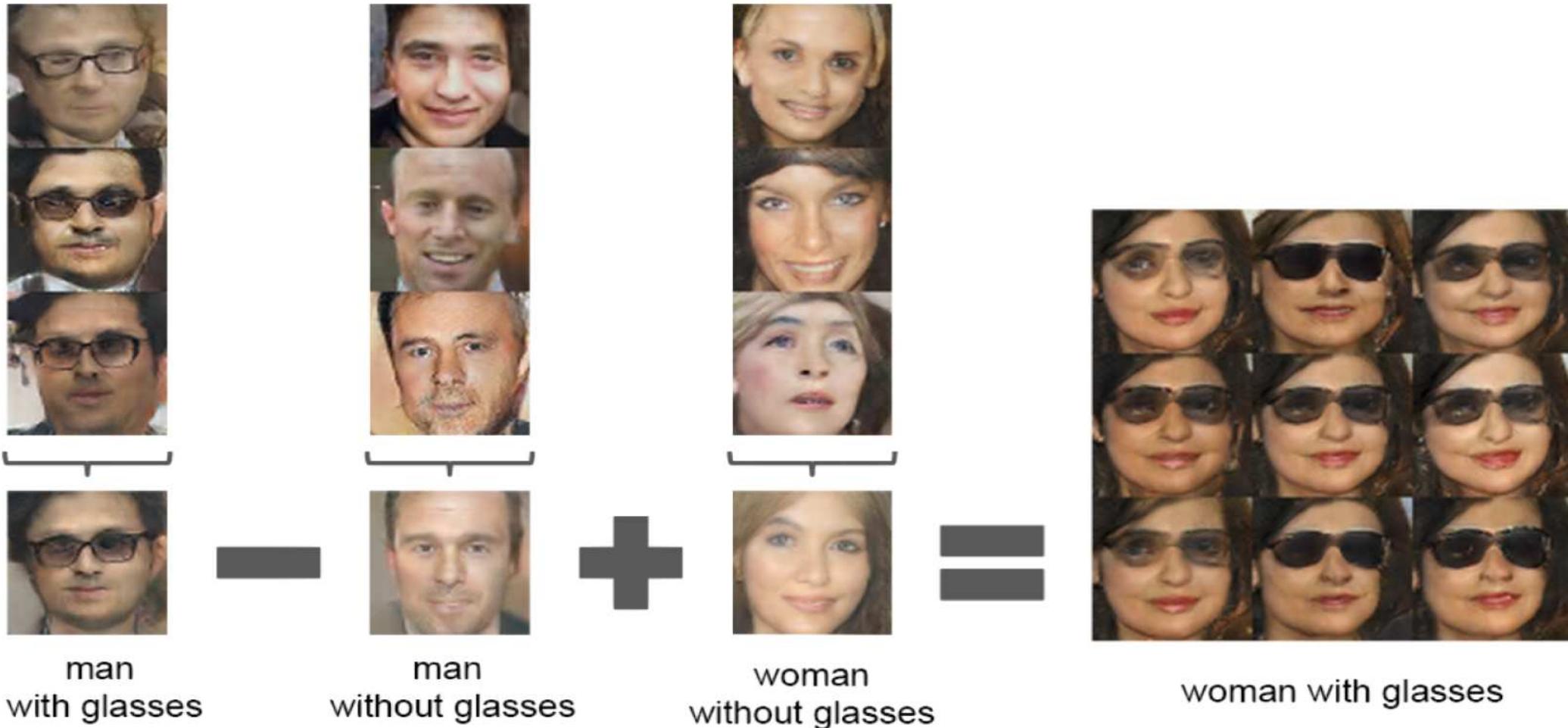
Generator Architecture



Key ideas:

- Replace FC hidden layers with Convolutions
 - **Generator:** Fractional-Strided convolutions
- Use Batch Normalization after each layer
- **Inside Generator**
 - Use ReLU for hidden layers
 - Use Tanh for the output layer

Latent vectors capture interesting patterns...



Part 2

- **Advantages of GANs**
- **Training Challenges**
 - Non-Convergence
 - Mode-Collapse
- **Proposed Solutions**
 - Supervision with Labels
 - Mini-Batch GANs
- **Modification of GAN's losses**
 - Discriminator (**EB-GAN**)
 - Generator (**InfoGAN**)

Advantages of GANs

- **Plenty of existing work on Deep Generative Models**

- Boltzmann Machine
- Deep Belief Nets
- Variational AutoEncoders (VAE)

- **Why GANs?**

- Sampling (or generation) is straightforward.
- Training doesn't involve Maximum Likelihood estimation.
- Robust to Overfitting since Generator never sees the training data.
- Empirically, GANs are good at capturing the modes of the distribution.

Problems with GANs

- **Probability Distribution is Implicit**
 - Not straightforward to compute $P(X)$.
 - Thus **Vanilla GANs** are only good for Sampling/Generation.
- **Training is Hard**
 - Non-Convergence
 - Mode-Collapse

Training Problems

- Non-Convergence
- Mode-Collapse

- Deep Learning models (in general) involve a single player

- The player tries to maximize its reward (minimize its loss).
- Use SGD (with Backpropagation) to find the optimal parameters.
- SGD has convergence guarantees (under certain conditions).
- **Problem:** With non-convexity, we might converge to local optima.

$$\min_G L(G)$$

- GANs instead involve two (or more) players

- Discriminator is trying to maximize its reward.
- Generator is trying to minimize Discriminator's reward.

$$\min_G \max_D V(D, G)$$

- SGD was not designed to find the Nash equilibrium of a game.
- **Problem:** We might not converge to the Nash equilibrium at all.

Non-Convergence

$$\min_x \max_y V(x, y)$$

Let $V(x, y) = xy$

- State 1:

x > 0	y > 0	V > 0
-------	-------	-------

Increase y	Decrease x
------------	------------
- State 2:

x < 0	y > 0	V < 0
-------	-------	-------

Decrease y	Decrease x
------------	------------
- State 3:

x < 0	y < 0	V > 0
-------	-------	-------

Decrease y	Increase x
------------	------------
- State 4 :

x > 0	y < 0	V < 0
-------	-------	-------

Increase y	Increase x
------------	------------
- State 5:

x > 0	y > 0	V > 0
-------	-------	-------

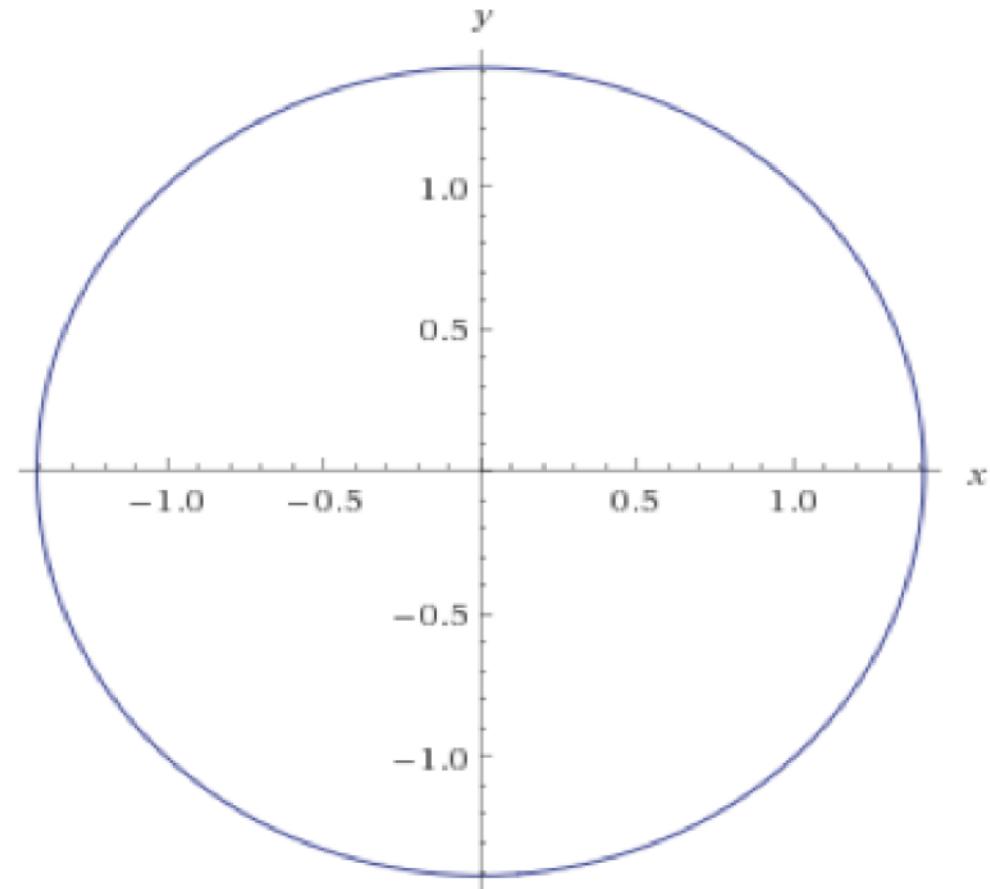
 == State 1

Increase y	Decrease x
------------	------------

Non-Convergence

$$\min_x \max_y xy$$

- $\frac{\partial}{\partial x} = -y \quad \dots \quad \frac{\partial}{\partial y} = x$
- $\frac{\partial^2}{\partial y^2} = \frac{\partial}{\partial x} = -y$
- Differential equation's solution has sinusoidal terms
- Even with a small learning rate, it will not converge

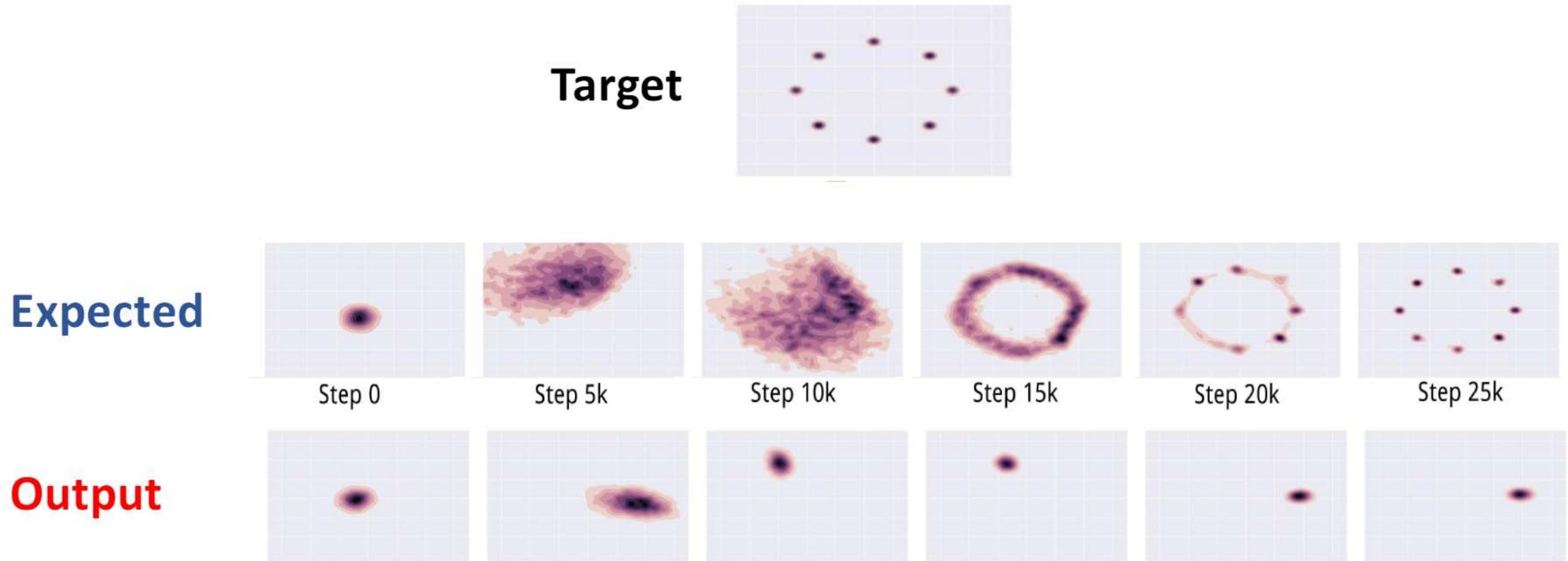


Problems with GANs

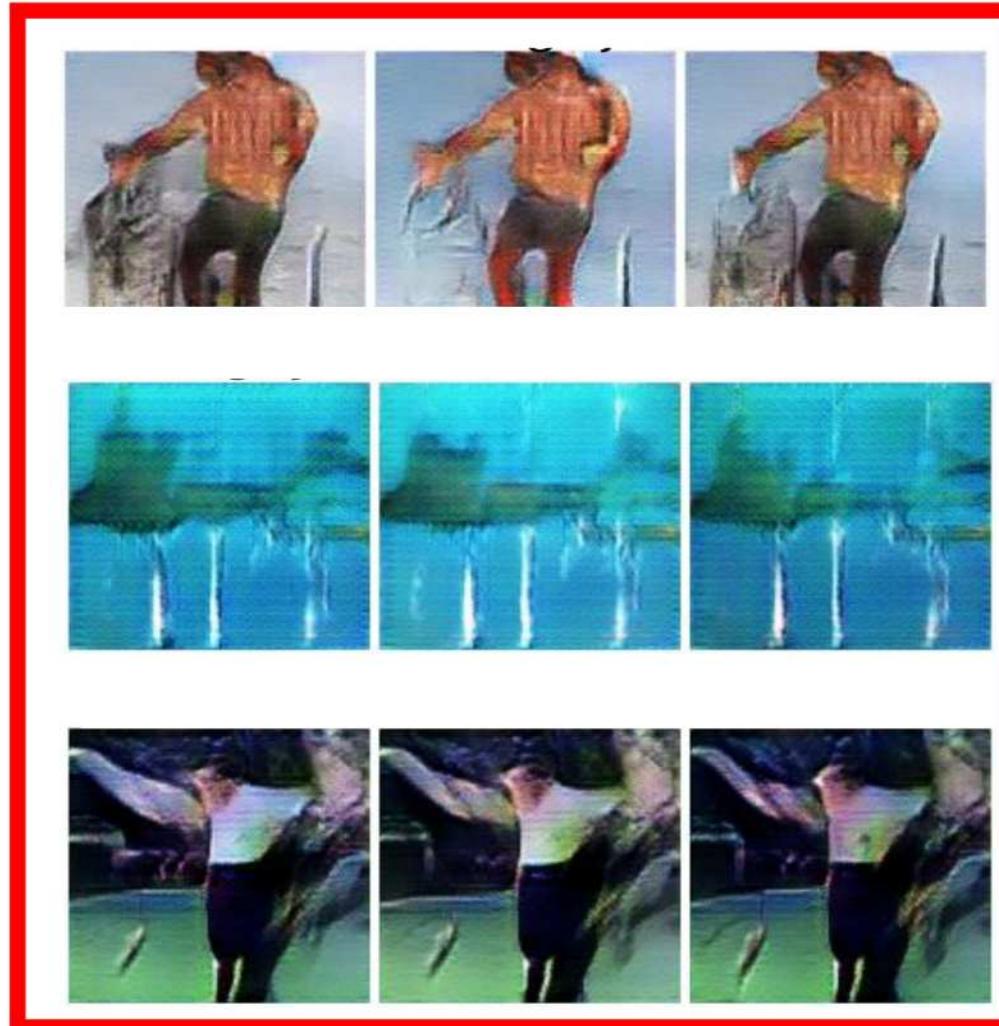
- Non-Convergence
- **Mode-Collapse**

Mode-Collapse

- Generator fails to output diverse samples



Some real examples



Some Solutions

- Mini-Batch GANs
- Supervision with labels
- Some recent attempts :-
 - Unrolled GANs
 - W-GANs

Basic (Heuristic) Solutions

- Mini-Batch GANs
- Supervision with labels

How to reward sample diversity?

- **At Mode Collapse,**
 - Generator produces good samples, but a very few of them.
 - Thus, Discriminator can't tag them as fake.
- **To address this problem,**
 - Let the Discriminator know about this edge-case.
- **More formally,**
 - Let the Discriminator look at the entire batch instead of single examples
 - If there is lack of diversity, it will mark the examples as fake
- **Thus,**
 - Generator will be forced to produce diverse samples.

Mini-Batch GANs

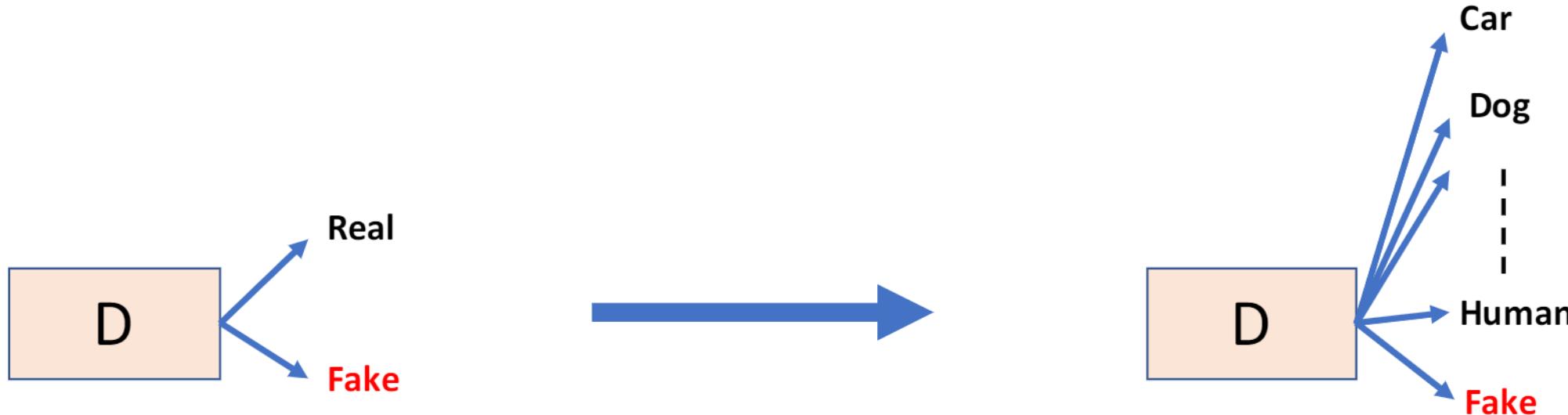
- Extract features that capture diversity in the mini-batch
 - For e.g. L2 norm of the difference between all pairs from the batch
- Feed those features to the discriminator along with the image
- Feature values will differ b/w diverse and non-diverse batches
 - Thus, Discriminator will rely on those features for classification
- This in turn,
 - Will force the Generator to match those feature values with the real data
 - Will generate diverse batches

Basic (Heuristic) Solutions

- Mini-Batch GANs
- **Supervision with labels**

Supervision with Labels

- Label information of the real data might help

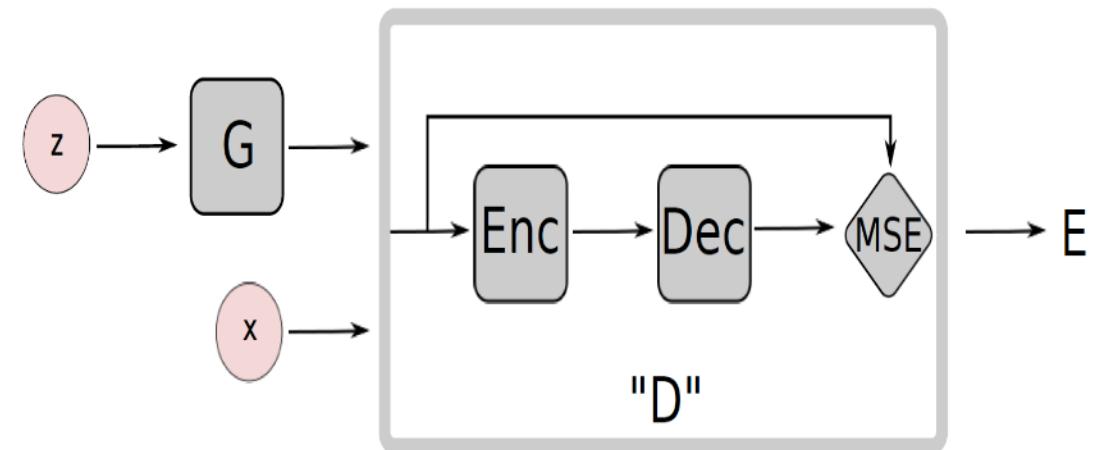


- Empirically generates much better samples

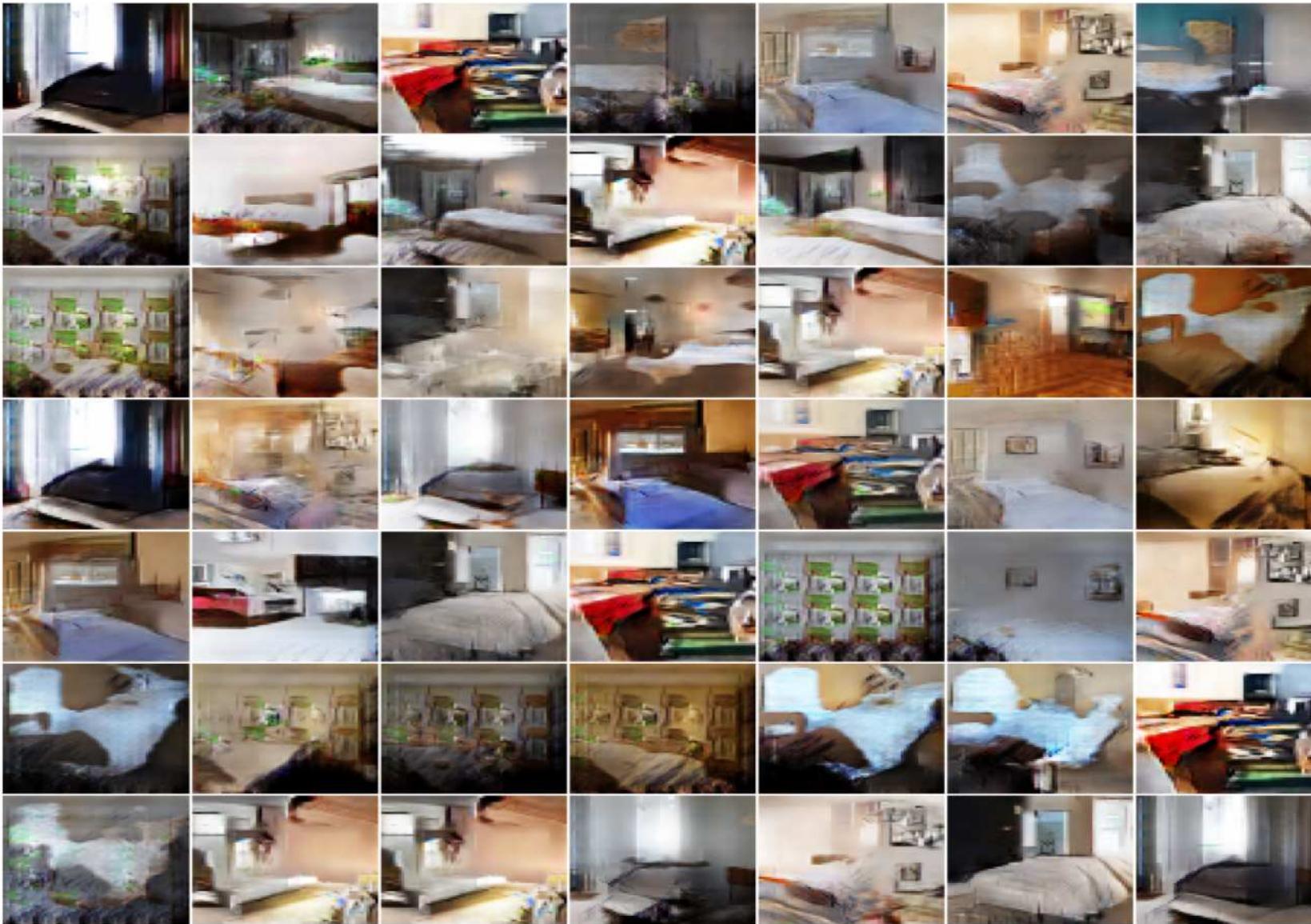
Energy-Based GANs

- Modified game plans
 - **Generator** will try to generate samples with low values
 - **Discriminator** will try to assign high scores to fake values
- Use AutoEncoder inside the Discriminator
- Use Mean-Squared Reconstruction error as $D(x)$
 - High Reconstruction Error for Fake samples
 - Low Reconstruction Error for Real samples

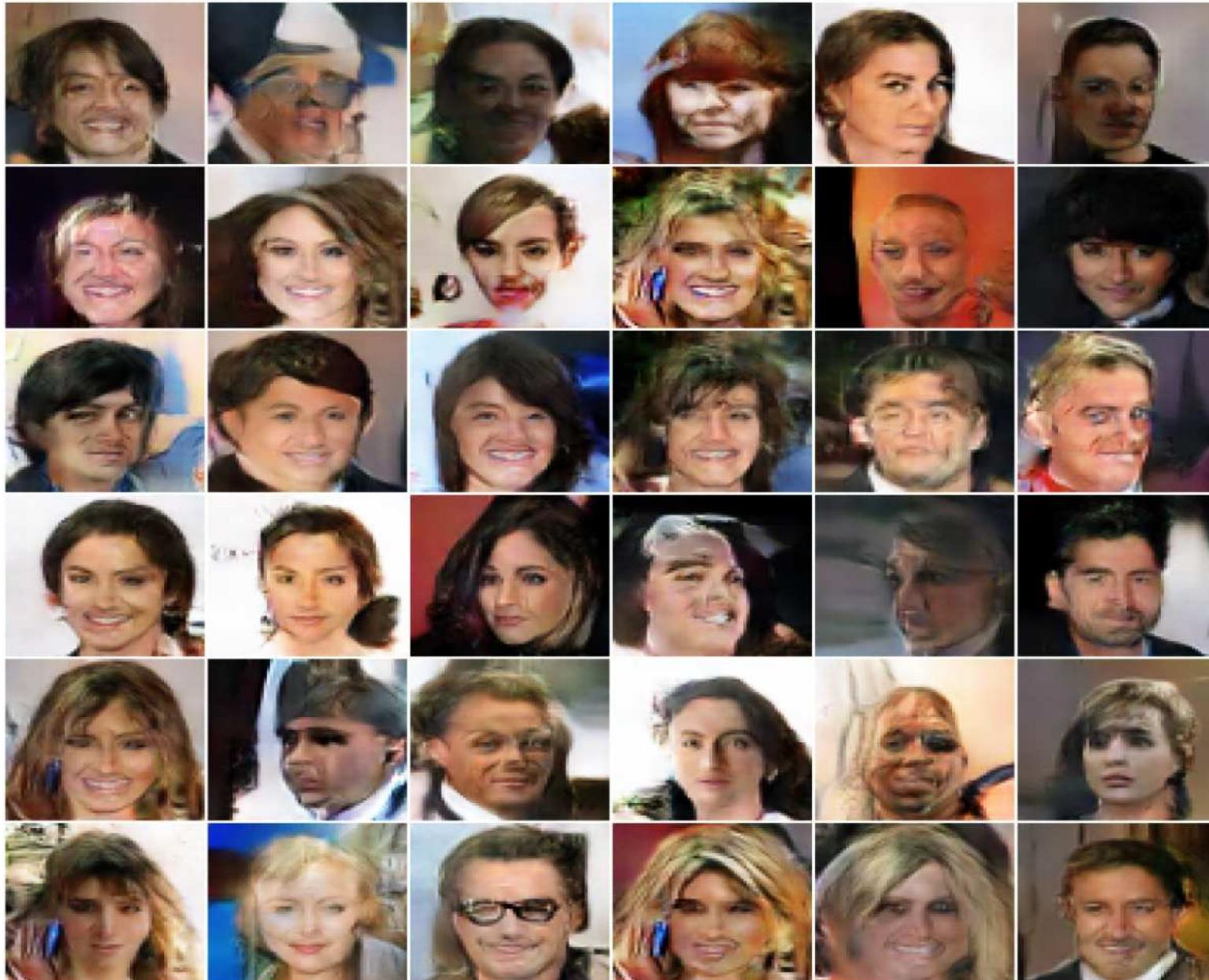
$$D(x) = ||Dec(Enc(x)) - x||_{MSE}$$



More Bedrooms...



Celebs...



The Cool Stuff...

3D Faces



(a) Azimuth (pose)



(b) Elevation



(c) Lighting



(d) Wide or Narrow

Cool Stuff (contd.)

3D Chairs



How to reward Disentanglement?

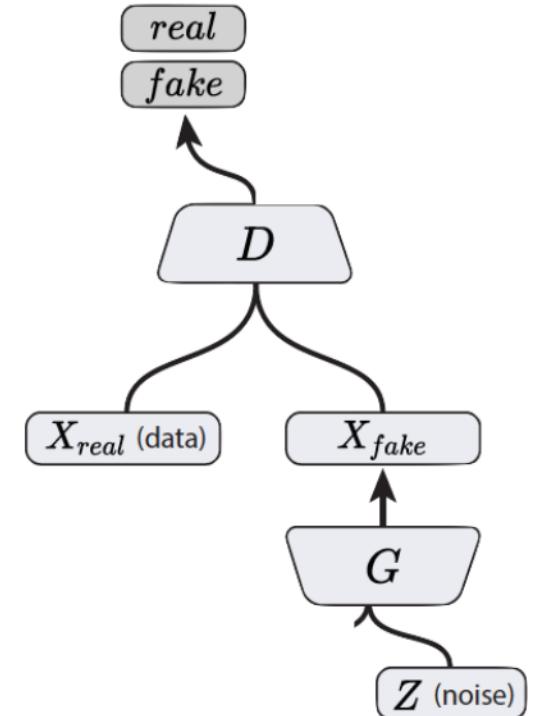
- Disentanglement means individual dimensions independently capturing key attributes of the image

- Let's partition the noise vector into 2 parts :-

- **z** vector will capture slight variations in the image
 - **c** vector will capture the main attributes of the image
 - For e.g. **Digit**, **Angle** and **Thickness** of images in MNIST

- If **c** vector captures the key variations in the image,

Will c and x_{fake} be highly correlated or weakly correlated?



Recap: Mutual Information

- Mutual Information captures the mutual dependence between two variables
- Mutual information between two variables X, Y is defined as:

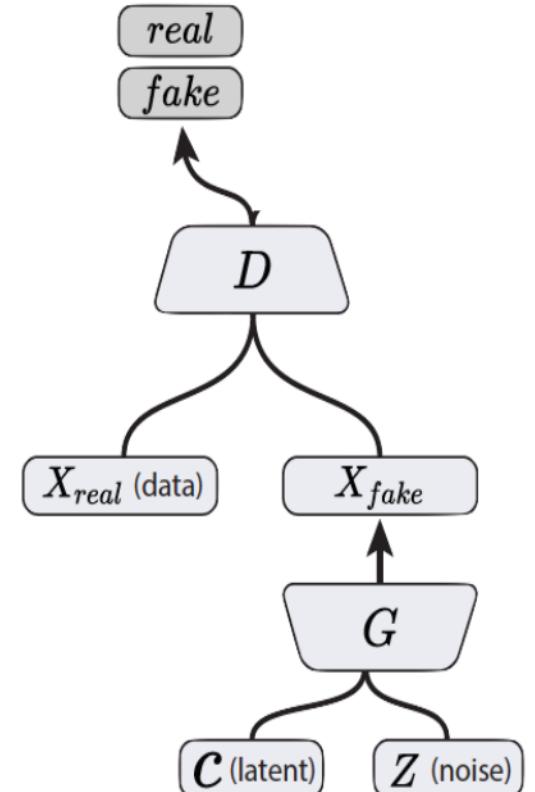
$$I(X; Y) = \sum_{x,y} p(x, y) \log\left(\frac{p(x, y)}{p(x)p(y)}\right)$$

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

InfoGAN

- We want to maximize the mutual information I between \mathbf{c} and $\mathbf{x} = \mathbf{G}(\mathbf{z}, \mathbf{c})$
- Incorporate in the value function of the minimax game.

$$\min_G \max_D V_I(D, G) = V(D, G) - \lambda I(c; G(z, c))$$



Part 3

- **Conditional GANs**
- **Applications**
 - Image-to-Image Translation
 - Text-to-Image Synthesis
 - Face Aging
- **Advanced GAN Extensions**
 - Coupled GAN
 - LAPGAN – Laplacian Pyramid of Adversarial Networks
 - Adversarially Learned Inference
- **Summary**

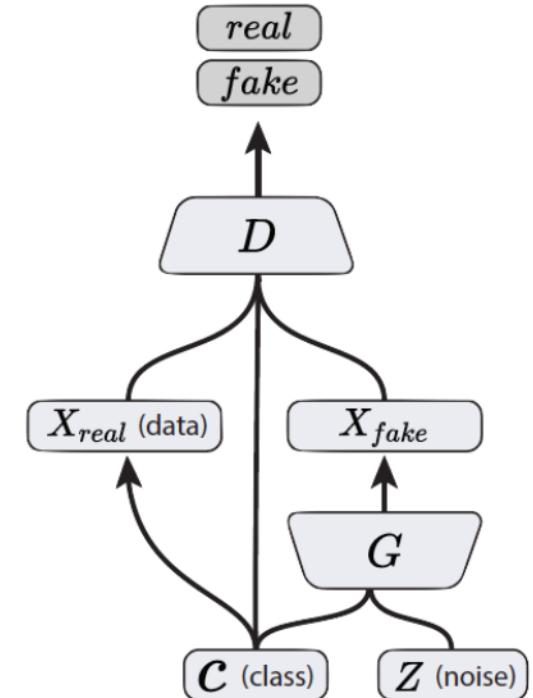
Conditional GANs

MNIST digits generated conditioned on their class label.

Figure 2 in the original paper.

Conditional GANs

- Simple modification to the original GAN framework that conditions the model on *additional information* for better multi-modal learning.
- Lends to many practical applications of GANs when we have explicit *supervision* available.



Conditional GAN
(Mirza & Osindero, 2014)

Image Credit: Figure 2 in Odena, A., Olah, C. and Shlens, J., 2016. Conditional image synthesis with auxiliary classifier GANs. *arXiv preprint arXiv:1610.09585*.

Part 3

- **Conditional GANs**
- **Applications**
 - Image-to-Image Translation
 - Text-to-Image Synthesis
 - Face Aging
- **Advanced GAN Extensions**
 - Coupled GAN
 - LAPGAN – Laplacian Pyramid of Adversarial Networks
 - Adversarially Learned Inference
- **Summary**

Image-to-Image Translation

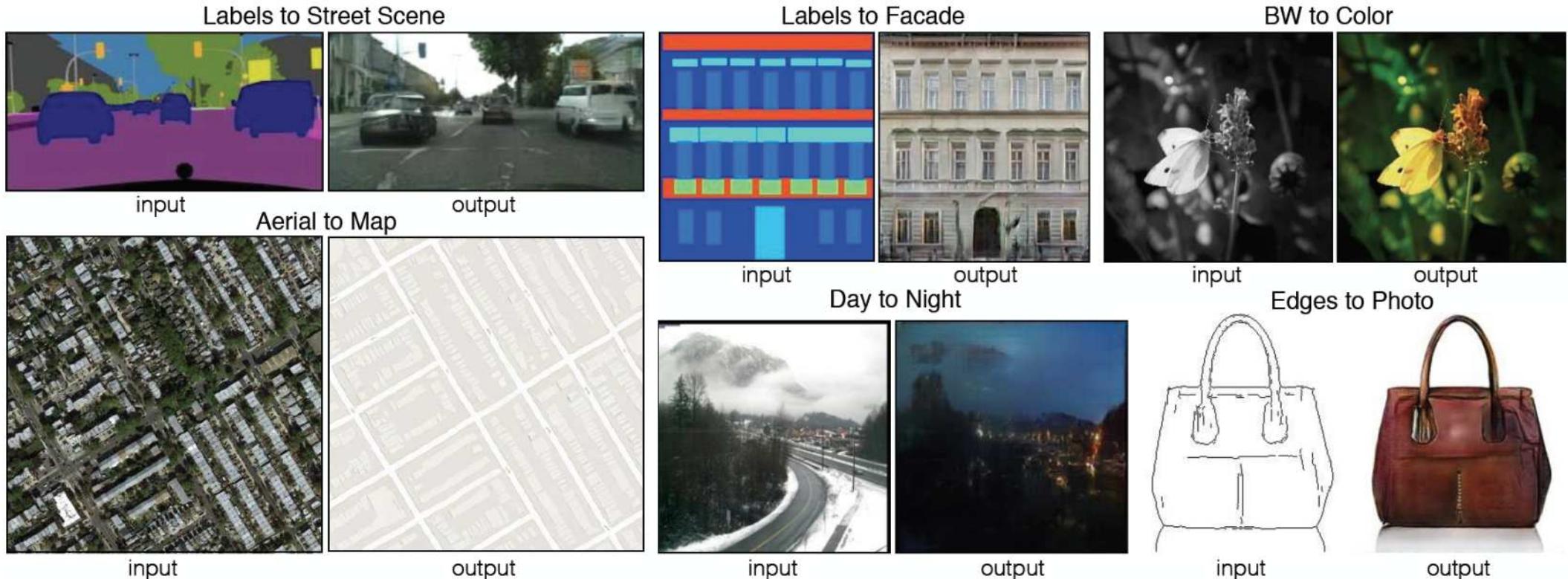


Figure 1 in the original paper.

[Link to an interactive demo of this paper](#)

Image-to-Image Translation

- Architecture: *DCGAN-based architecture*
- Training is conditioned on the images from the source domain.
- Conditional GANs provide an effective way to handle many complex domains without worrying about designing *structured loss* functions explicitly.

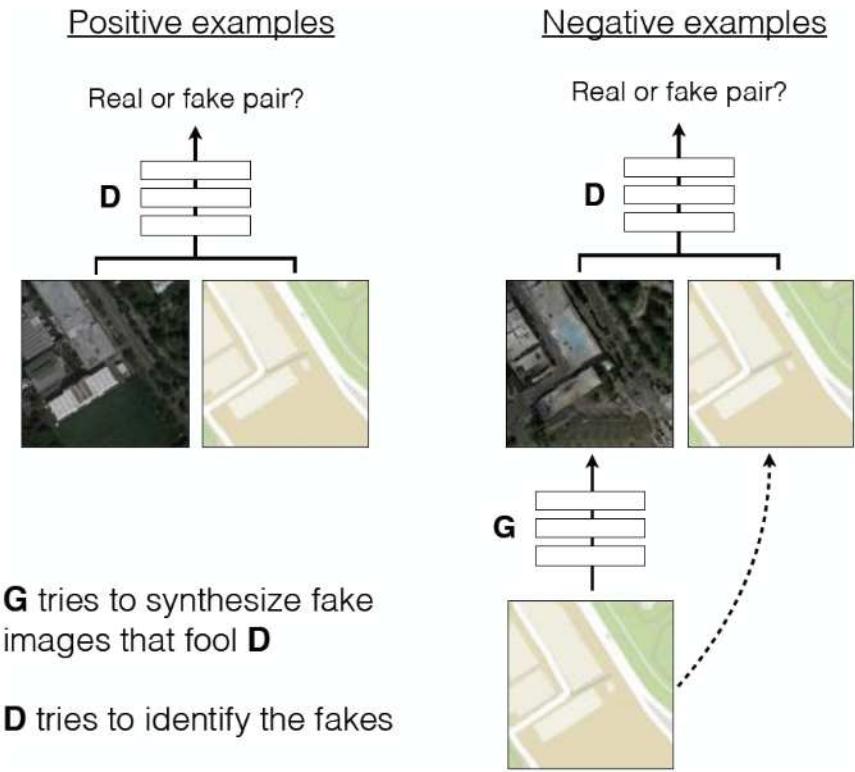


Figure 2 in the original paper.

Text-to-Image Synthesis

Motivation

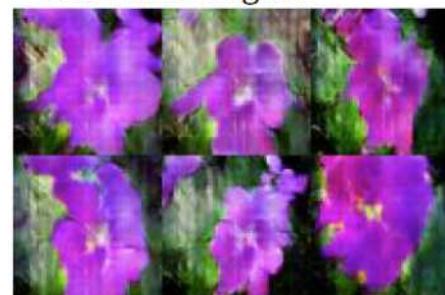
Given a text description, generate images closely associated.

Uses a conditional GAN with the generator and discriminator being condition on “dense” text embedding.

this small bird has a pink breast and crown, and black primaries and secondaries.



the flower has petals that are bright pinkish purple with white stigma



this magnificent fellow is almost all black with a red crest, and white cheek patch.



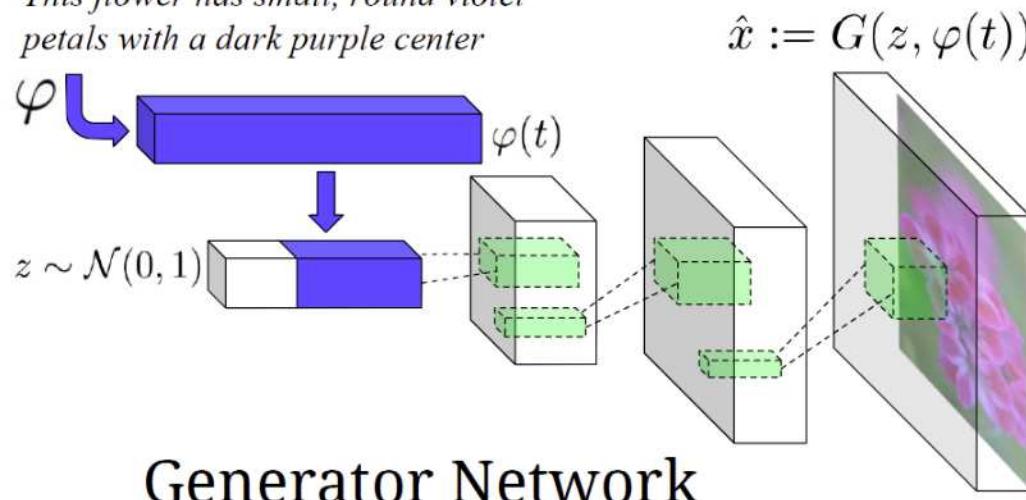
this white and yellow flower have thin white petals and a round yellow stamen



Figure 1 in the original paper.

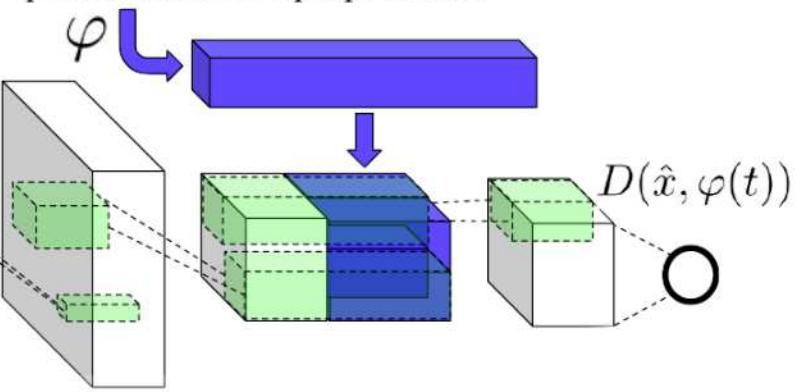
Text-to-Image Synthesis

This flower has small, round violet petals with a dark purple center



Generator Network

This flower has small, round violet petals with a dark purple center



Discriminator Network

Figure 2 in the original paper.

Positive Example:

Real Image, Right Text

Negative Examples:

Real Image, Wrong Text

Fake Image, Right Text

Face Aging with Conditional GANs

- Differentiating Feature: Uses an *Identity Preservation Optimization* using an auxiliary network to get a better approximation of the latent code (z^*) for an input image.
- Latent code is then conditioned on a discrete (one-hot) embedding of age categories.

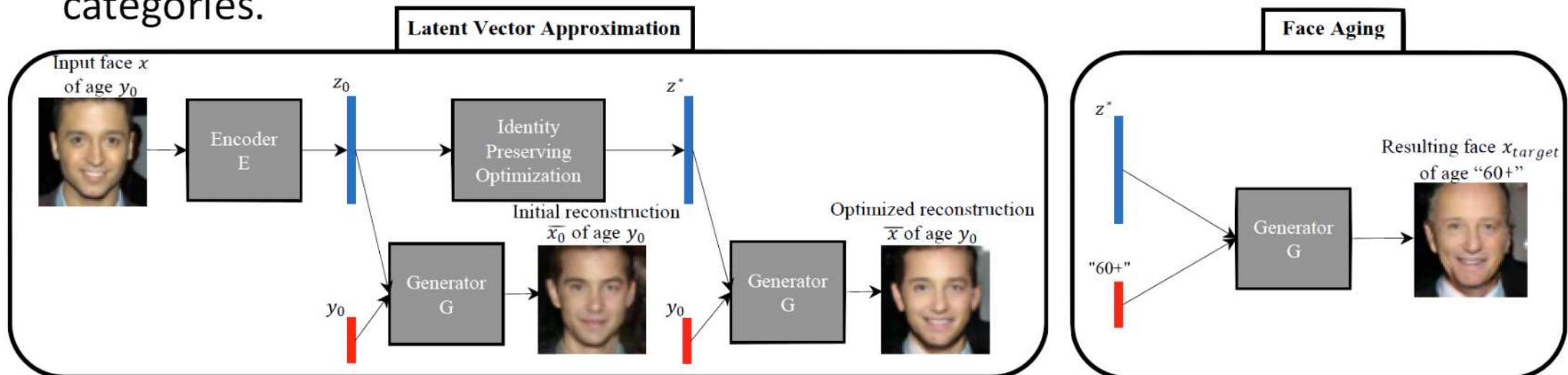


Figure 1 in the original paper.

Face Aging with Conditional GANs

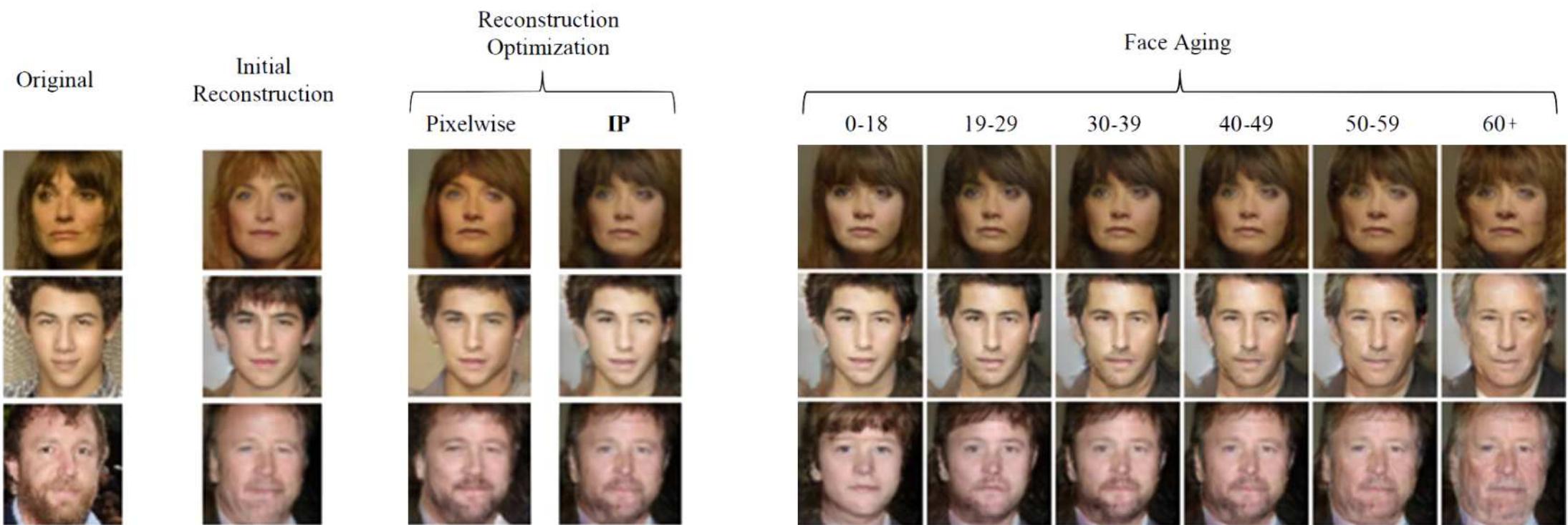


Figure 3 in the original paper.

Conditional GANs

Conditional Model Collapse

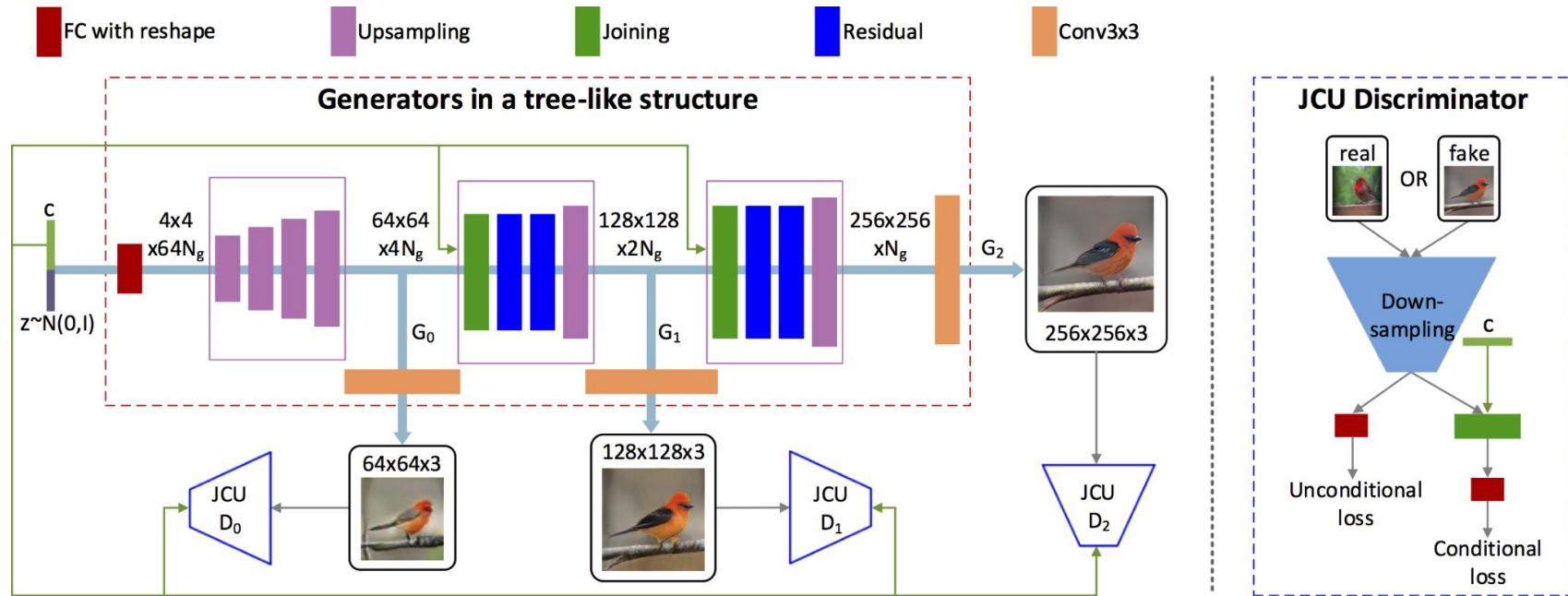
- Scenario observed when the Conditional GAN starts *ignoring* either the code (c) or the noise variables (z).
- This limits the diversity of images generated.



Credit?

STACKGAN++

- Text to Image generation
- High resolution, multi-scale image synthesis
- Smoothly increasing resolution with several stacked generators



STACKGAN++

Text
description

This bird is red and brown in color, with a stubby beak



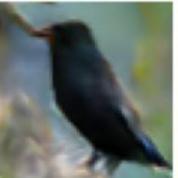
The bird is short and stubby with yellow on its body



A bird with a medium orange bill white body gray wings and webbed feet



This small black bird has a short, slightly curved bill and long legs



A small bird with varying shades of brown with white under the eyes



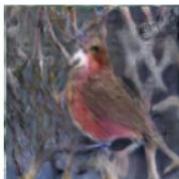
A small yellow bird with a black crown and a short black pointed beak



This small bird has a white breast, light grey head, and black wings and tail



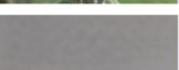
64x64
GAN-INT-CLS



128x128
GAWWN



256x256
StackGAN-v1



256x256
StackGAN-v2

