Synthetic Image Generation via Generative Adversarial Nets

Jianping Fan
Department of Computer Science
UNC-Charlotte

Course Website:
http://webpages.uncc.edu/jfan/itcs5152.html

Generative Adversarial Nets, NIPS 2014

GANs (Generative Adversarial Nets) simultaneously train **two models**: a **generative model** G that captures the data distribution, and a **discriminative model** D that estimates the probability that a sample came from the training data rather than G.

The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to 1/2 everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples.

Generative Adversarial Nets, NIPS 2014

GANs

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

Generative Adversarial Nets, NIPS 2014

Why Generative Models?

- **We've only seen discriminative models so far**
  - Given an image **X**, predict a label **Y**
  - Estimates **P(Y|X)**

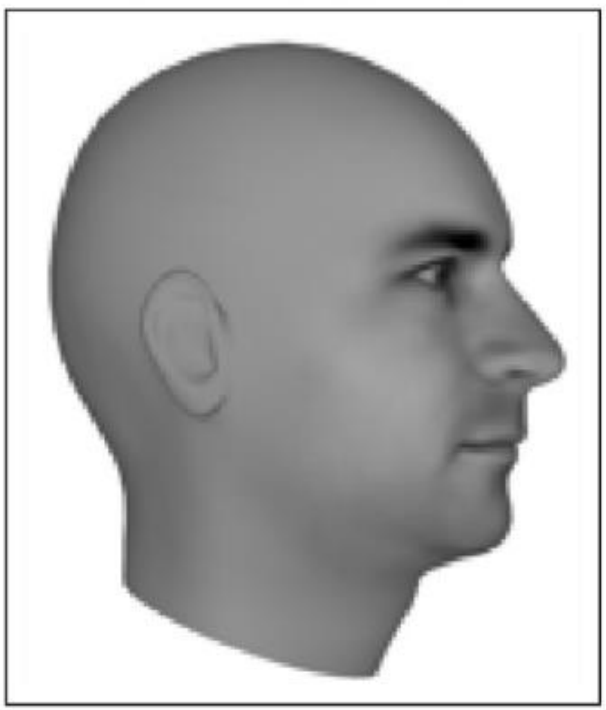
- **Discriminative models have several key limitations**
  - Can't model **P(X)**, i.e. the probability of seeing a certain image
  - Thus, can't sample from **P(X)**, i.e. **can't generate new images**

- **Generative models (in general) cope with all of above**
  - Can model **P(X)**
  - Can generate new images

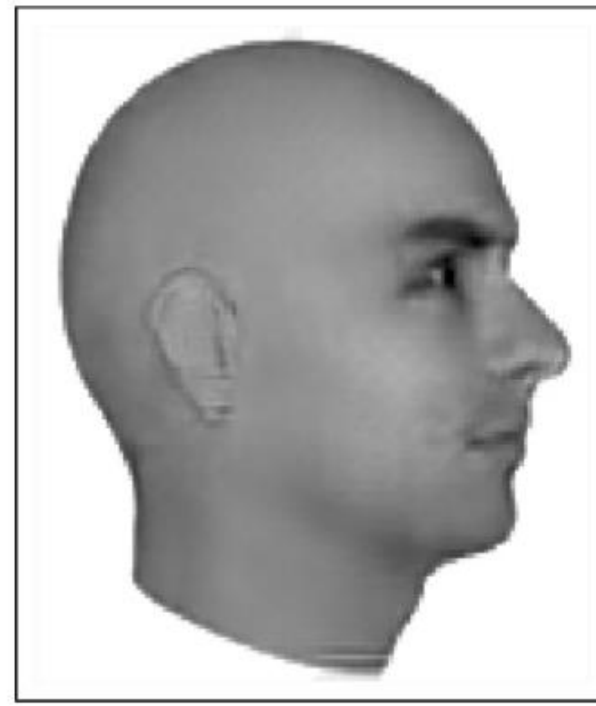
Generative Adversarial Nets, NIPS 2014

Magic of GANs…

Ground Truth



Adversarial



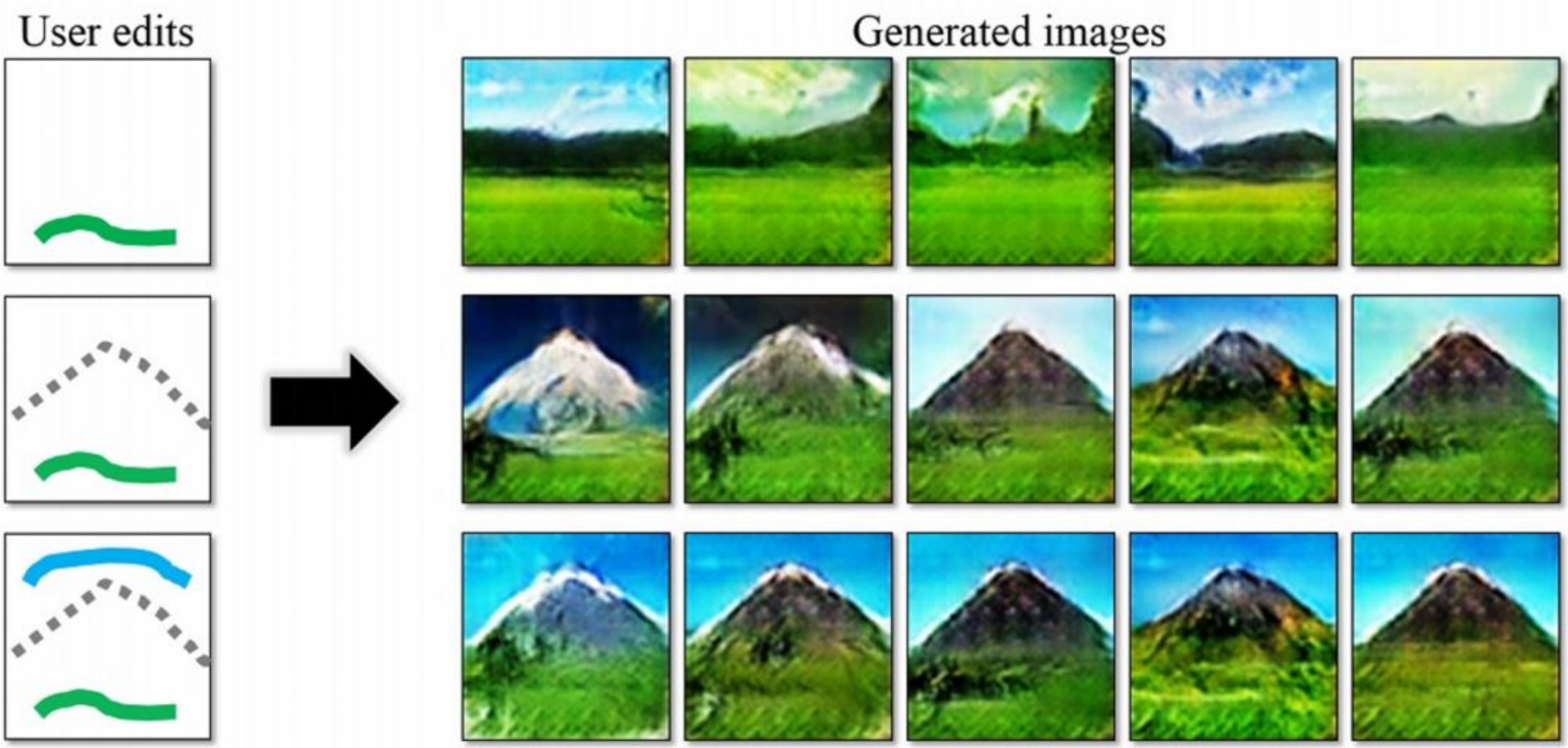
Generative Adversarial Nets, NIPS 2014

Magic of GANs…

Which one is Computer generated?

Generative Adversarial Nets, NIPS 2014

Magic of GANs…



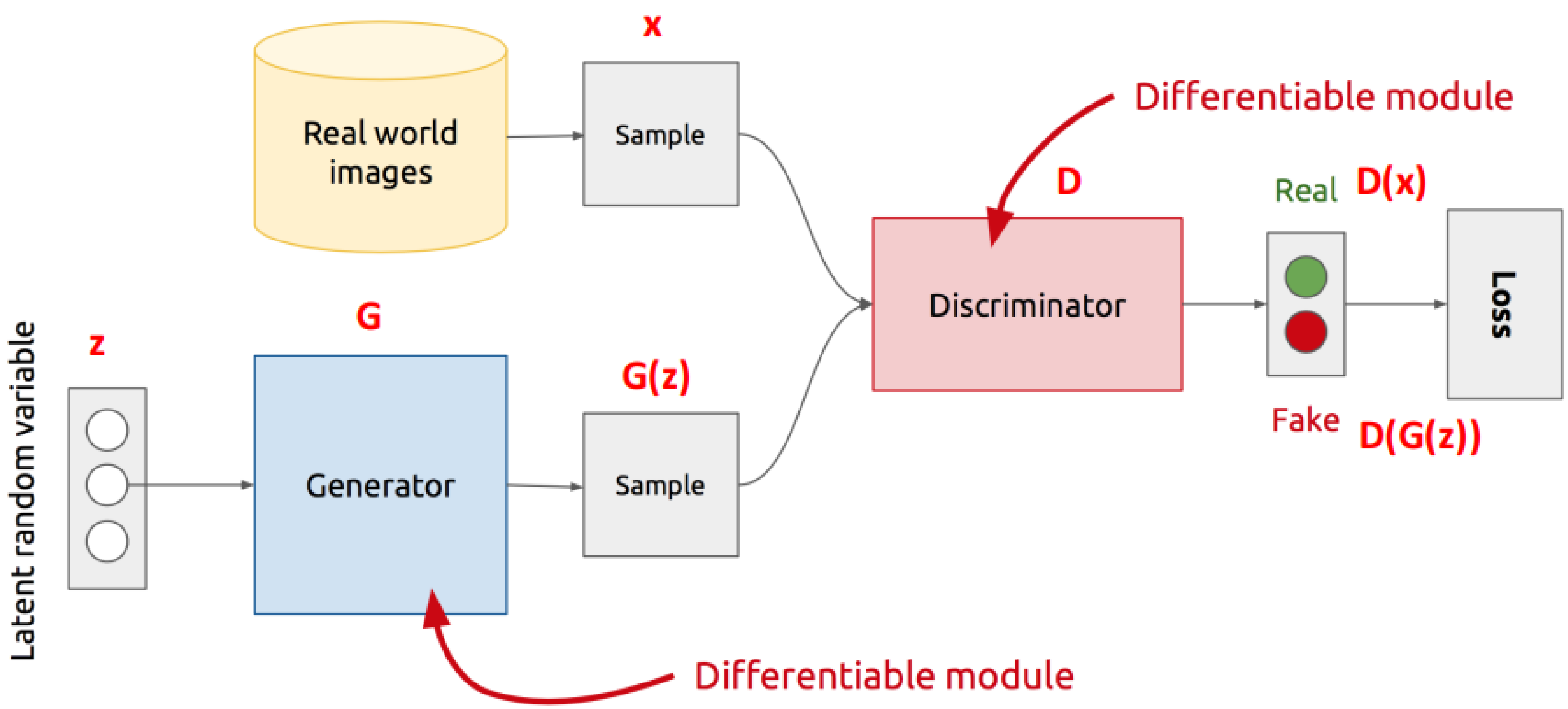
http://people.eecs.berkeley.edu/~junyanz/projects/gvm/

Adversarial Training

- **In the last lecture, we saw:**
  - We can generate adversarial samples to fool a discriminative model
  - We can use those adversarial samples to make models robust
  - We then require more effort to generate adversarial samples
  - Repeat this and we get better discriminative model

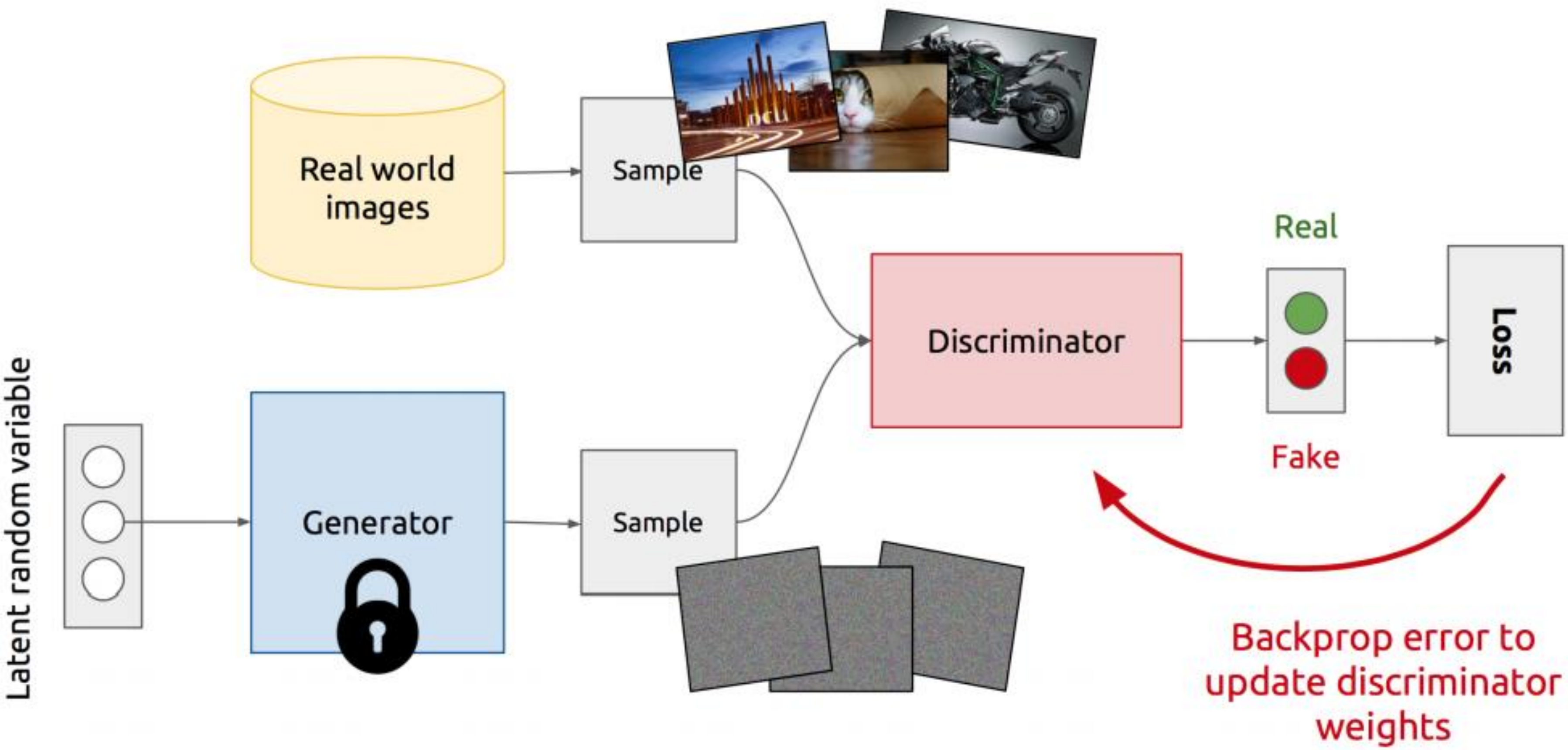
- **GANs extend that idea to generative models:**
  - Generator: generate fake samples, tries to fool the Discriminator
  - Discriminator: tries to distinguish between real and fake samples
  - Train them against each other
  - Repeat this and we get better Generator and Discriminator

GAN's Architecture

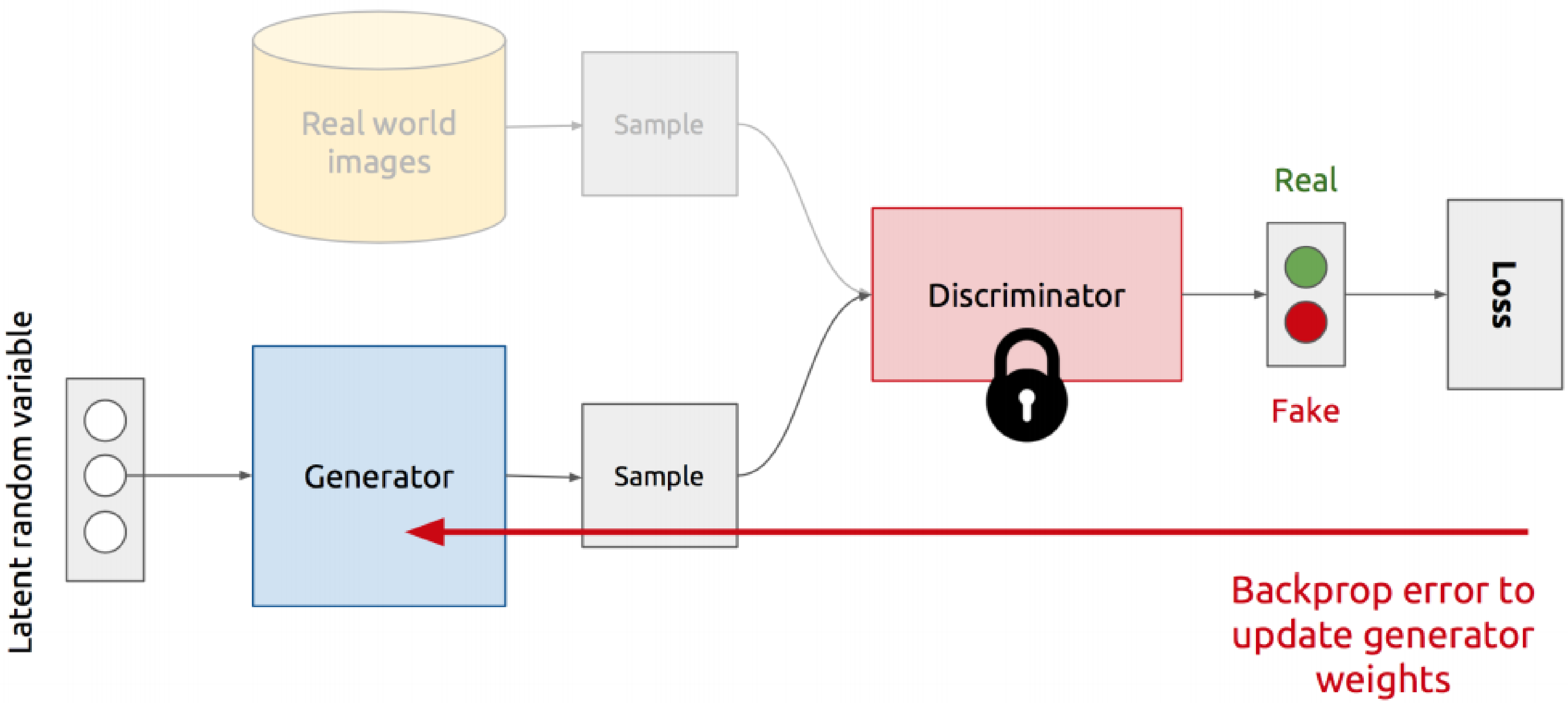


- **Z** is some random noise (Gaussian/Uniform).
- **Z** can be thought as the latent representation of the image.

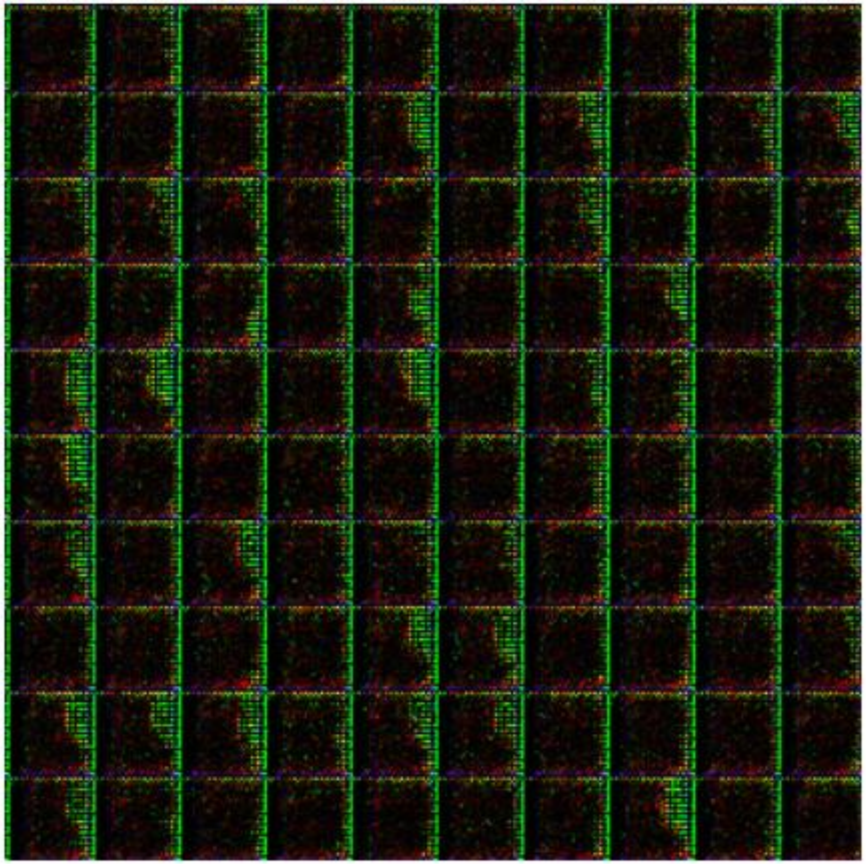
Training Discriminator



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 $\boldsymbol{V(D, G)}$
  - 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:
  - $P_{data}(x) = P_{gen}(x) \;\; \forall x$
  - $\mathrm{D}(x) = \frac{1}{2} \;\; \forall x$

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.

for number of training iterations **do**

Discriminator updates

for k steps **do**

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

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

end for

Generator updates

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

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

end for

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

Vanishing gradient strikes back again…

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

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

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

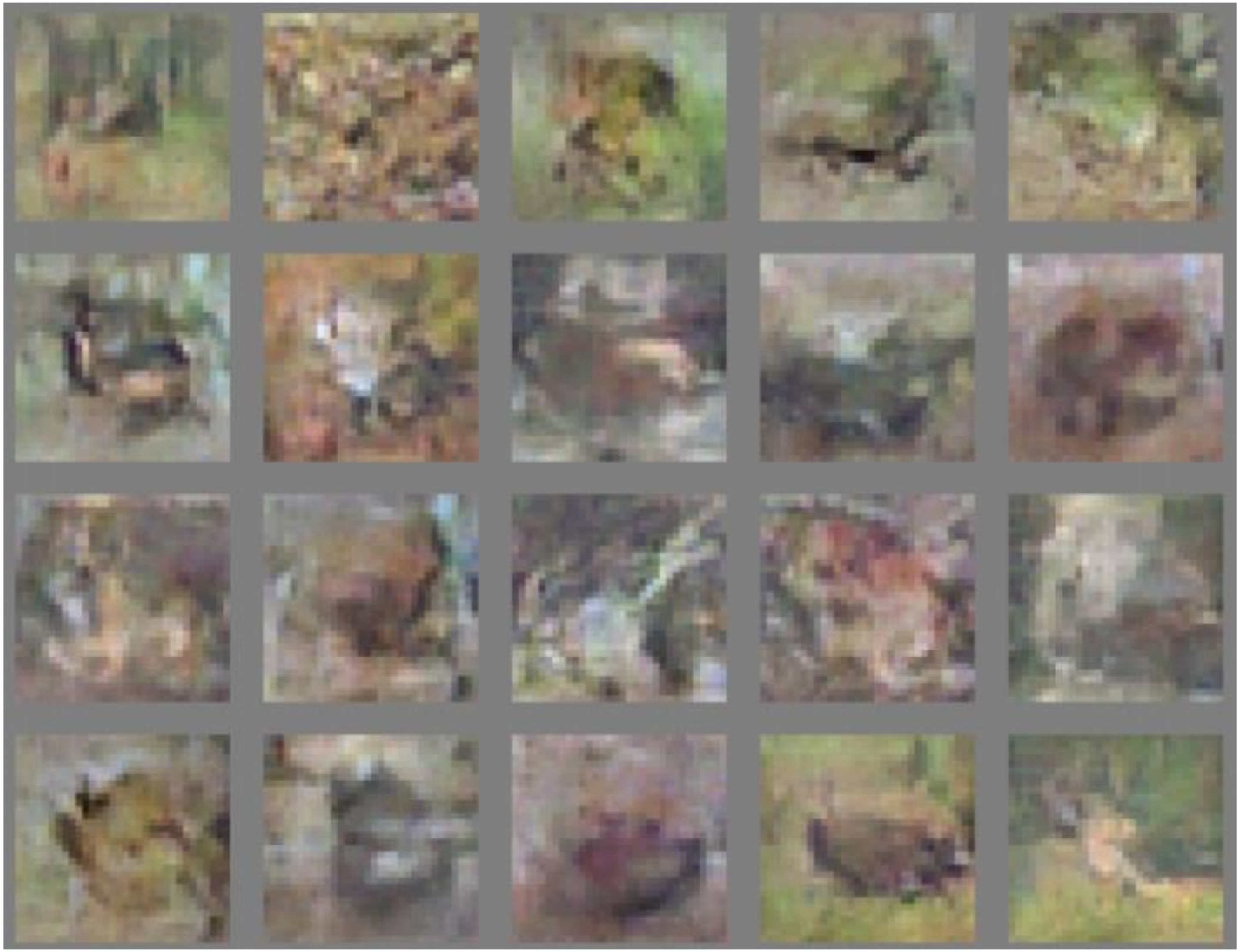
- $\nabla_a \log\left(1 - \sigma(a)\right) = \frac{-\nabla_a \sigma(a)}{1-\sigma(a)} = \frac{-\sigma(a)\left(1-\sigma(a)\right)}{1-\sigma(a)} = -\sigma(a) = -D\left(G(z)\right)$

- Gradient goes to 0 if D is confident, i.e. $D\left(G(z)\right) \to 0$

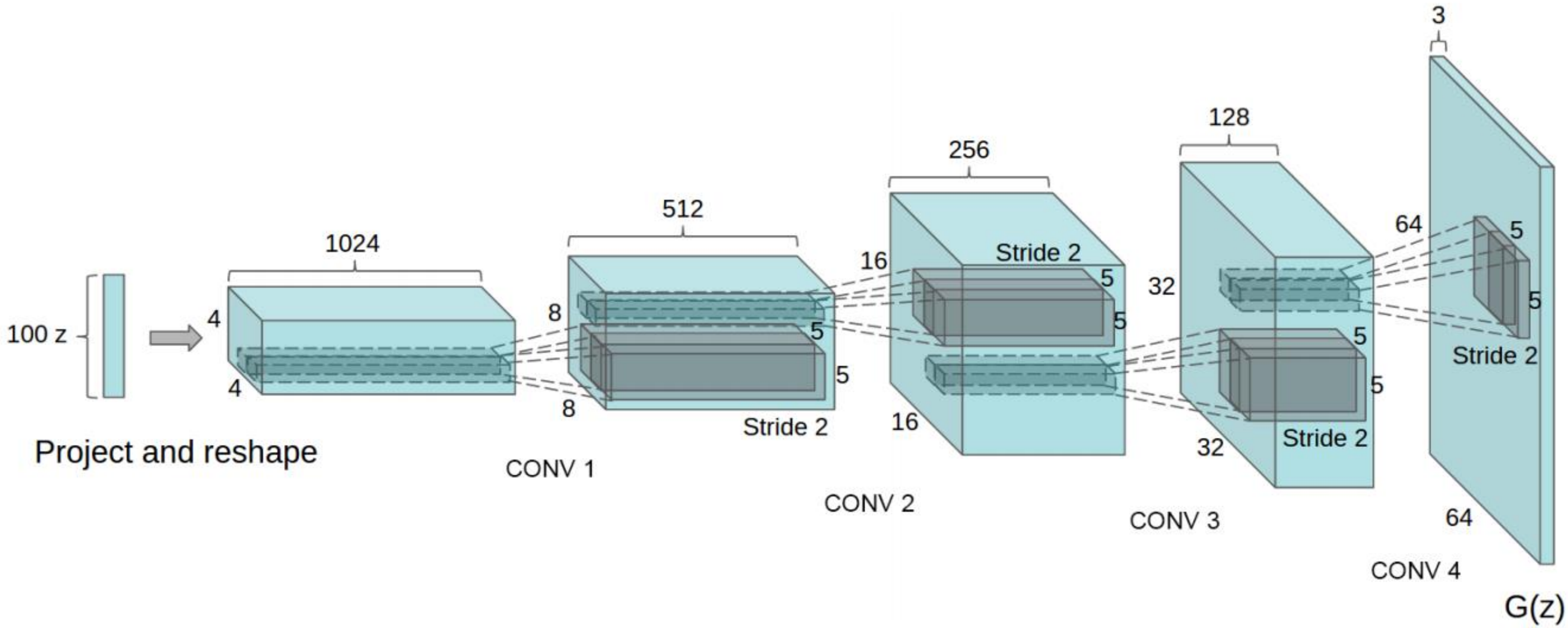
- Minimize $\boxed{-\mathbb{E}_{z\sim q(z)}\left[\log D\left(G(z)\right)\right]}$ for **Generator** instead (keep Discriminator as it is)

Faces

CIFAR



UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS, ICLR 2016

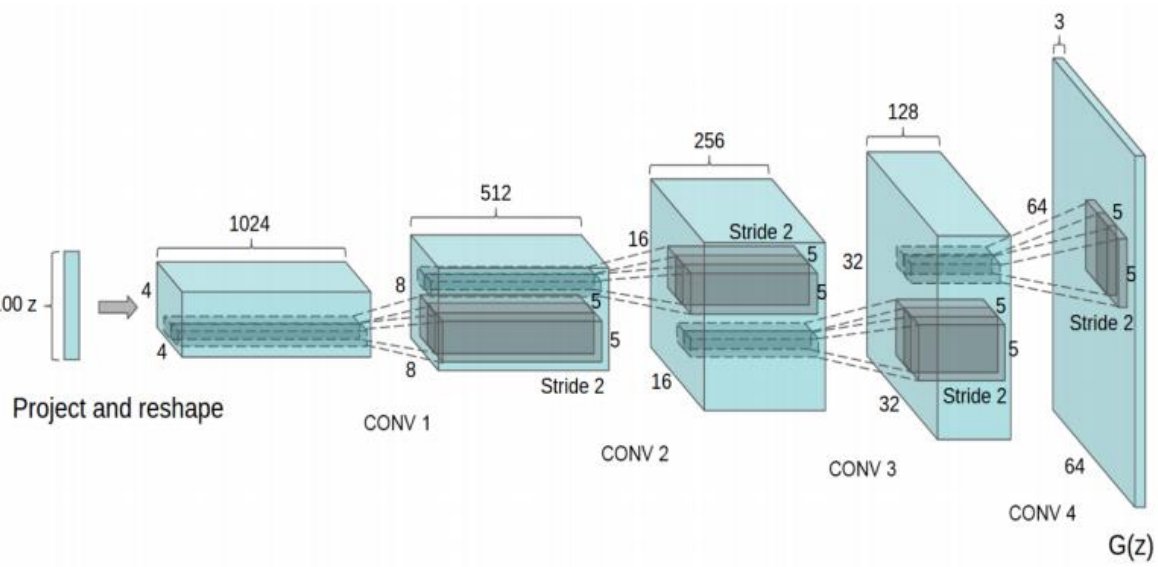


DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64 × 64 pixel image. Notably, no fully connected or pooling layers are used.

UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS, ICLR 2016

Deep Convolutional GANs (DCGANs)

Generator Architecture

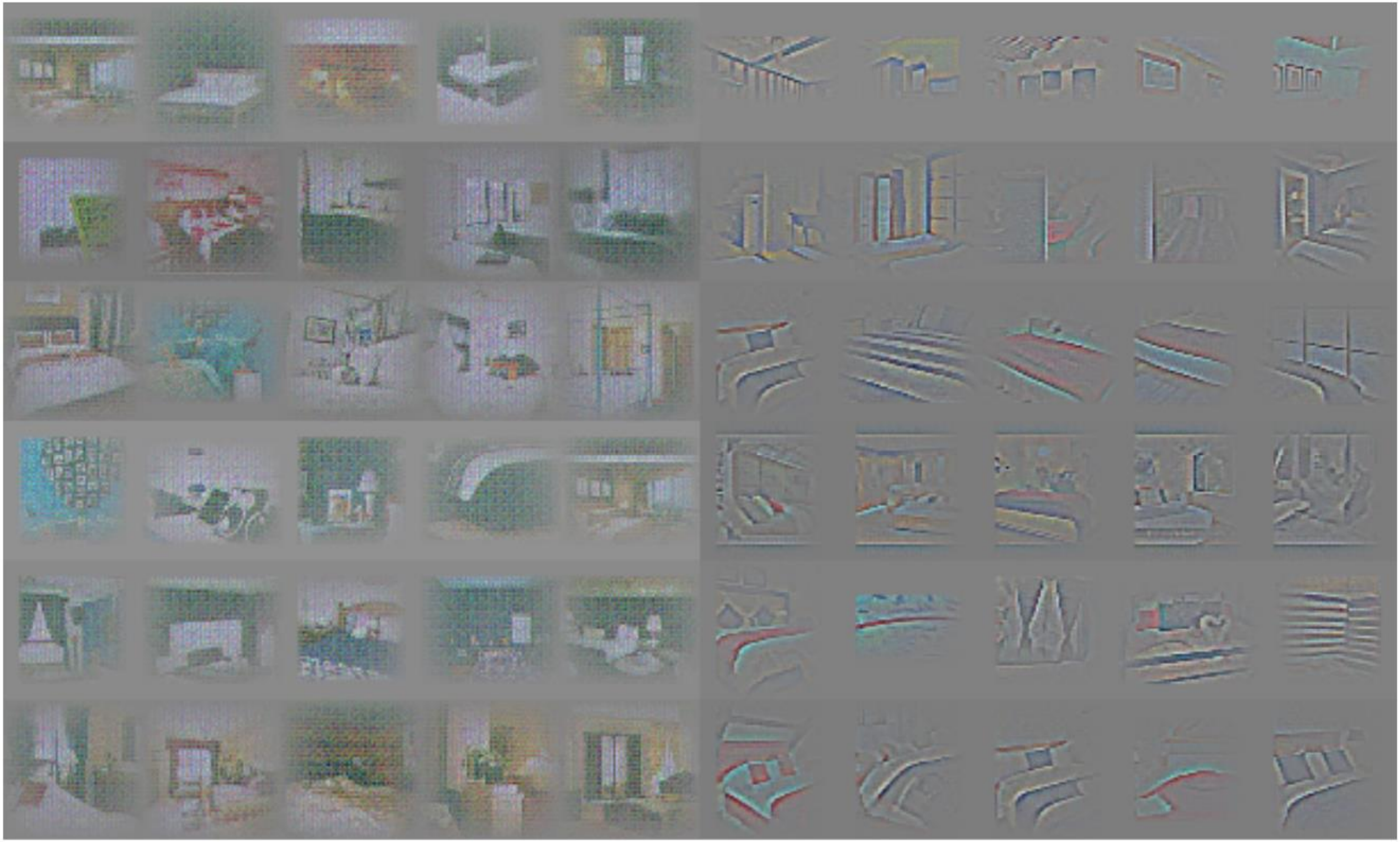


Key ideas:

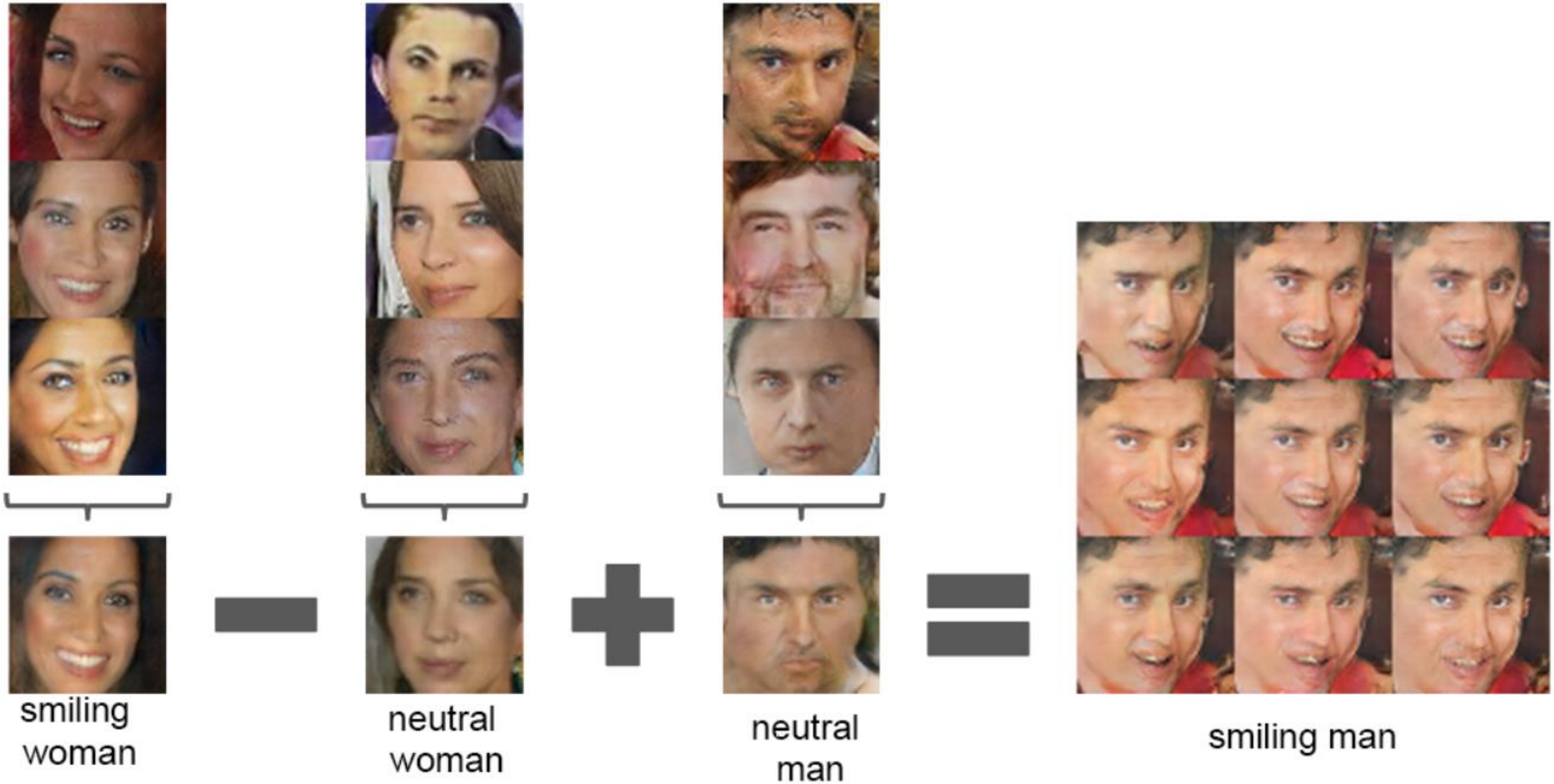
- Replace FC hidden layers with Convolutions
  - **Generator:** Fractional-Strided convolutions
- Use Batch Normalization after each layer
- **Inside Generator**
  - Use ReLU for hidden layers
  - Use Tanh for the output layer

Generated bedrooms after one training pass through the dataset. Theoretically, the model could learn to memorize training examples, but this is experimentally unlikely as we train with a small learning rate and minibatch SGD. We are aware of no prior empirical evidence demonstrating memorization with SGD and a small learning rate.

Generated bedrooms after five epochs of training. There appears to be evidence of visual under-fitting via repeated noise textures across multiple samples such as the base boards of some of the beds.



Random filters **Trained filters**


smiling
woman
neutral
woman
neutral
man
smiling man

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

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

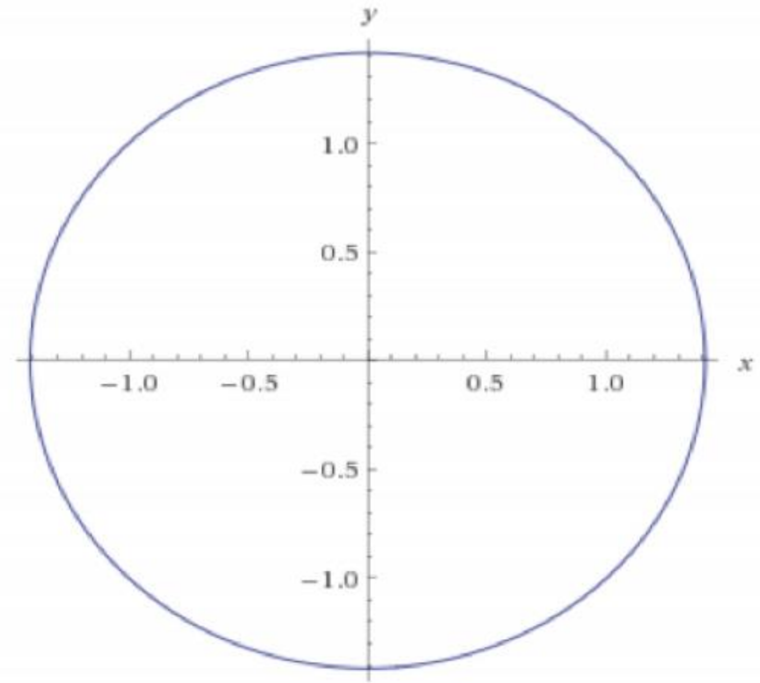
Let $V(x, y) = xy$

• 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

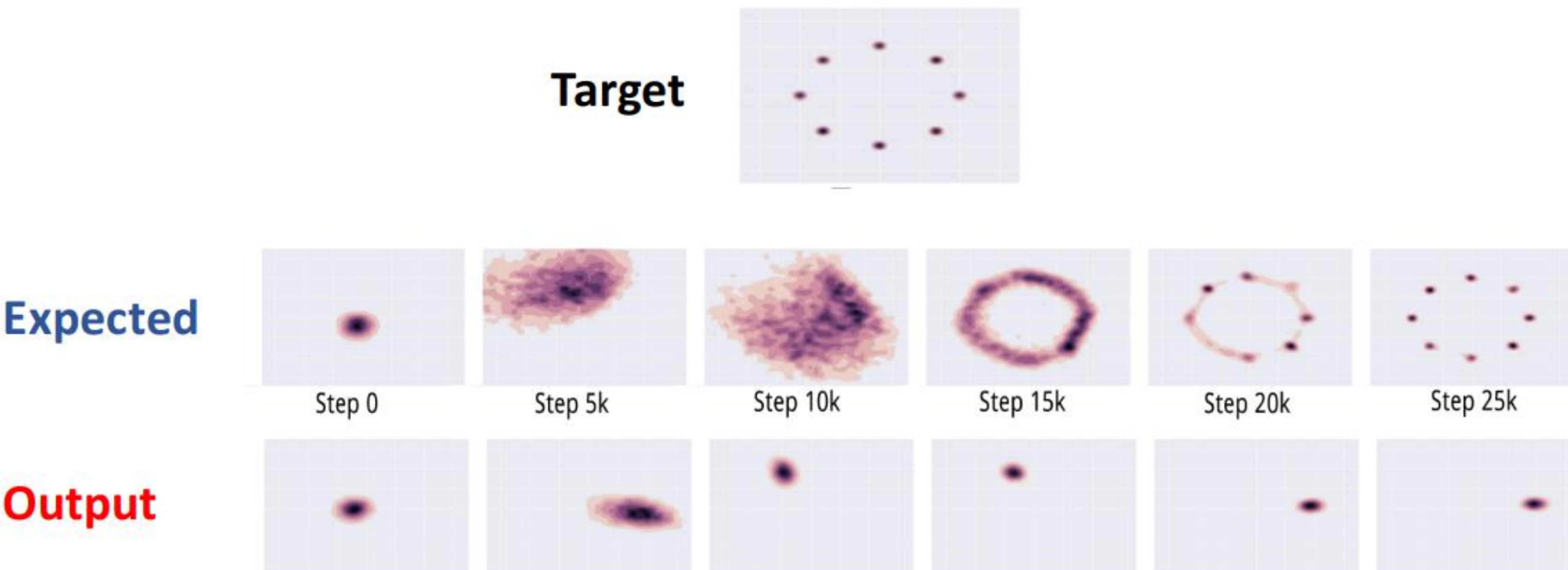
$$\min_x \max_y \; xy$$

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



Mode-Collapse

- **Generator fails to output diverse samples**



Some real examples

Salimans, Tim, et al. "Improved techniques for training gans." NIPS2016.

Some Solutions

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

- **Some recent attempts :-**
  - Unrolled GANs
  - W-GANs

How to reward **sample diversity**?

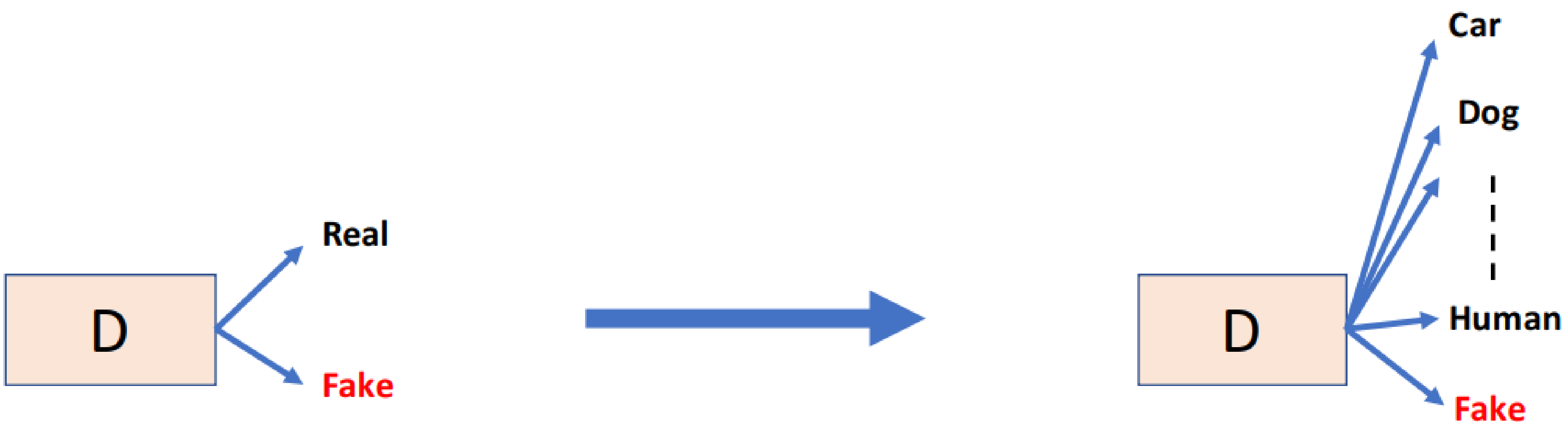
- **At Mode Collapse,**
  - Generator produces good samples, but a very few of them.
  - Thus, Discriminator can't tag them as fake.
- **To address this problem,**
  - Let the Discriminator know about this edge-case.
- **More formally,**
  - Let the Discriminator look at the entire batch instead of single 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

Supervision with Labels

- Label information of the real data might help



- Empirically generates much better samples

Zhao, Junbo, Michael Mathieu, and Yann LeCun. "Energy-based generative adversarial network." arXiv preprint arXiv:1609.03126 (2016)

Alternate view of GANs

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

$$D^* = \arg\max_D V(D,G)$$

$$G^* = \arg\min_G V(D,G)$$

- In this formulation, Discriminator's strategy was $D(x) \rightarrow 1, \; D(G(z)) \rightarrow 0$

- Alternatively, we can flip the binary classification labels i.e. **Fake = 1**, **Real = 0**

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

- In this new formulation, Discriminator's strategy will be $D(x) \rightarrow 0, \; D(G(z)) \rightarrow 1$

Alternate view of GANs (Contd.)

- If all we want to encode is $D(x) \to 0, \quad D\big(G(z)\big) \to 1$

$$\boxed{D^* = argmax_D \; \mathbb{E}_{x\sim p(x)}\big[\log\big(1 - D(x)\big)\big] + \mathbb{E}_{z\sim q(z)}\big[\log\big(D(G(z))\big)\big]}$$

We can use this

$$\boxed{D^* = argmin_D \; \mathbb{E}_{x\sim p(x)} \log\big(D(x)\big) + \mathbb{E}_{z\sim q(z)}\left[\log\left(1 - D\big(G(z)\big)\right)\right]}$$

- Now, we can replace cross-entropy with any loss function **(Hinge Loss)**

$$\boxed{D^* = argmin_D \; \mathbb{E}_{x\sim p(x)} D(x) + \mathbb{E}_{z\sim q(z)} \max\left(0, m - D\big(G(z)\big)\right)}$$

- And thus, instead of outputting probabilities, Discriminator just has to output :-
  - High values for fake samples
  - Low values for real samples

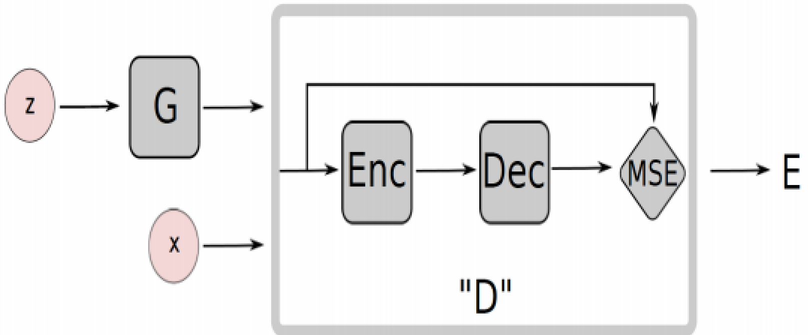
Energy-Based GANs

- Modified game plans
  - **Generator** will try to generate samples with low values
  - **Discriminator** will try to assign high scores to fake values

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

- **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



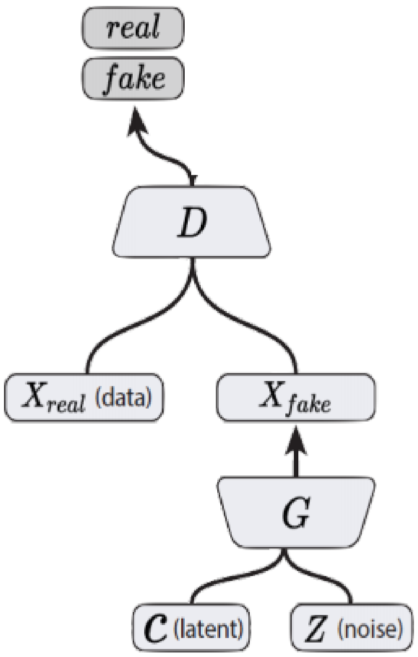
Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. InfoGAN: Interpretable Representation Learning by Information Maximization Generative Adversarial Nets, NIPS (2016).

InfoGAN

- We want to maximize the mutual information I between $\boldsymbol{c}$ and $\mathbf{x} = \boldsymbol{G}(\boldsymbol{z}, \boldsymbol{c})$

- Incorporate in the value function of the minimax game.

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



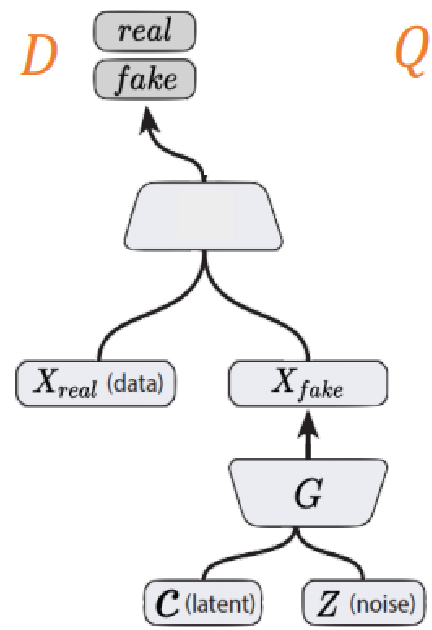
Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. InfoGAN: Interpretable Representation Learning by Information Maximization Generative Adversarial Nets, NIPS (2016).

InfoGAN

Mutual Information's Variational Lower bound

$$I\big(c; G(z,c)\big) = H(c) - H(c|G(z,c))$$
$$= \mathbb{E}_{x\sim G(z,c)}\left[\mathbb{E}_{c'\sim P(c|x)}[\log P(c'|x)]\right] + H(c)$$
$$= \mathbb{E}_{x\sim G(z,c)}\left[D_{KL}(P||Q) + \mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]\right] + H(c)$$
$$\geq \mathbb{E}_{x\sim G(z,c)}\left[\mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]\right] + H(c)$$

$$\geq \mathbb{E}_{c\sim P(c),\ x\sim G(z,c)}[\log Q(c|x)] + H(c)$$



Mirza, Mehdi, and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014).

Conditional GANs

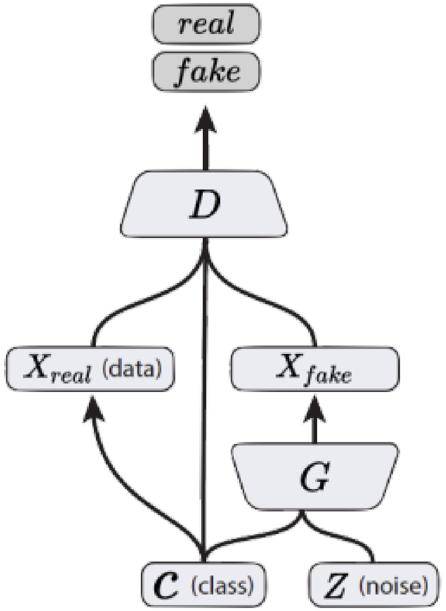
MNIST digits generated conditioned on their class label.

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)

Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. “Image-to-image translation with conditional adversarial networks”. arXiv preprint arXiv:1611.07004. (2016).

Image-to-Image Translation

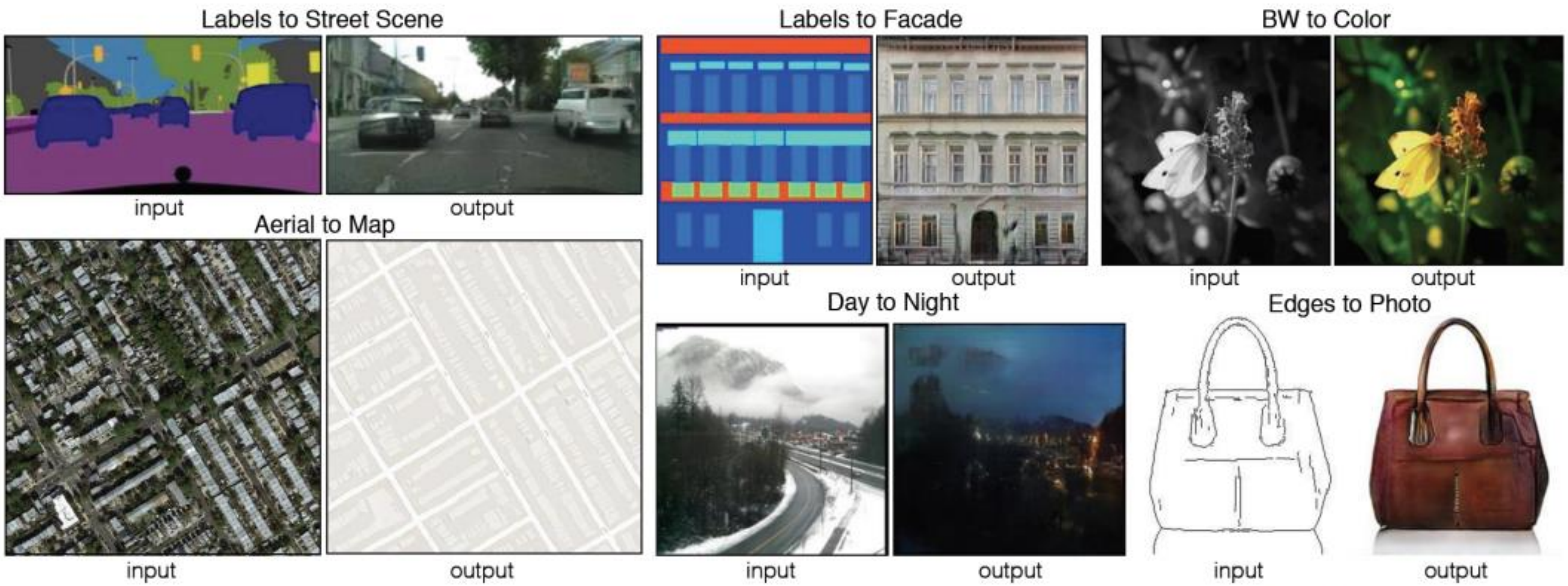


Figure 1 in the original paper.

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.

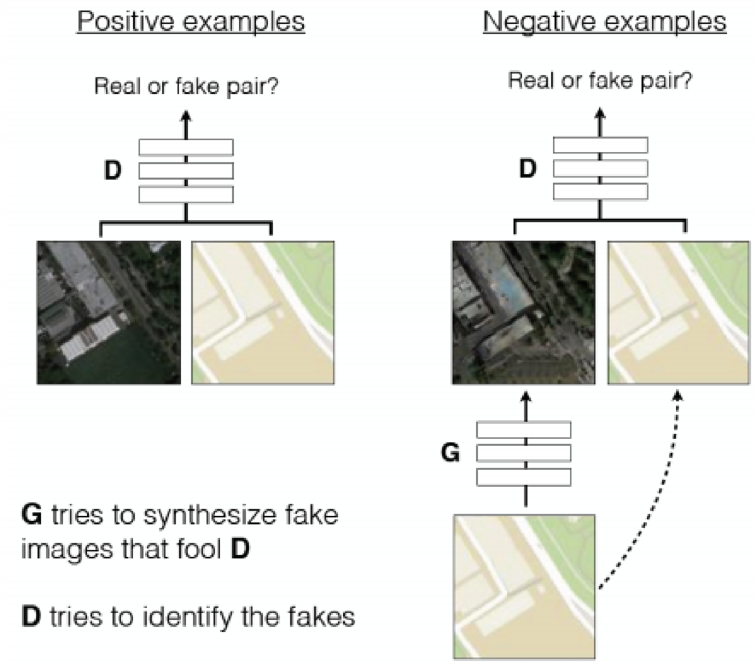


Figure 2 in the original paper.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. “Generative adversarial text to image synthesis”. ICML (2016).

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.

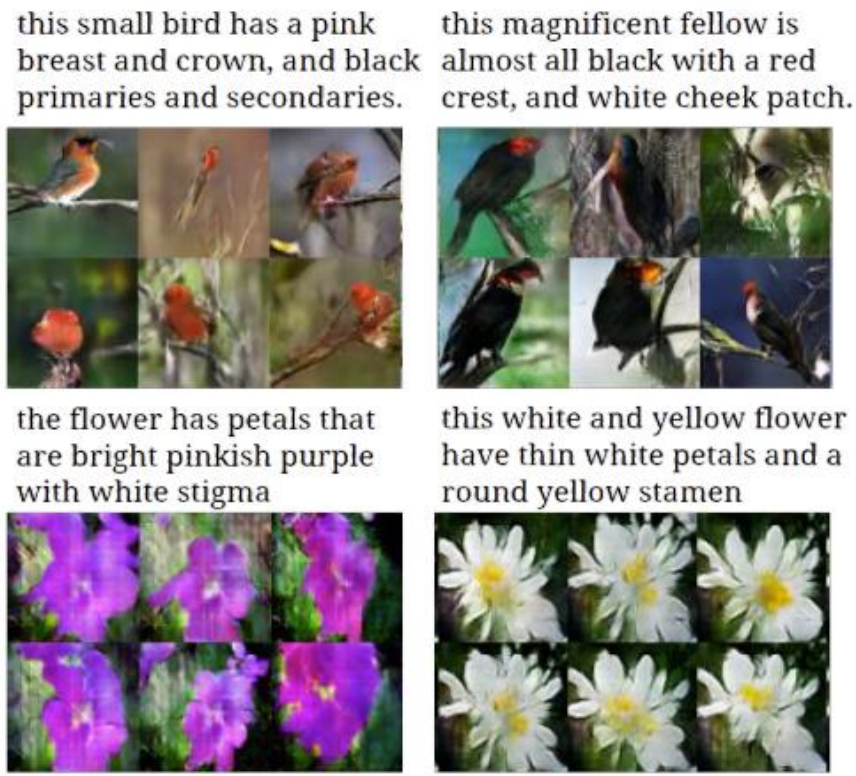


Figure 1 in the original paper.

Text-to-Image Synthesis

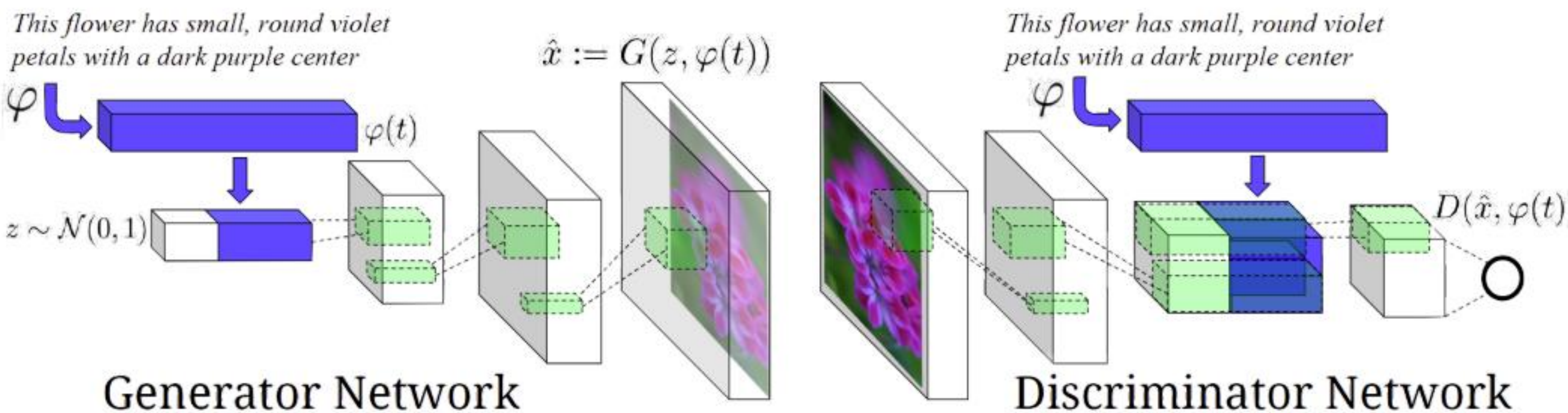


Figure 2 in the original paper.

Positive Example:

Real Image, Right Text

Negative Examples:

Real Image, Wrong Text

Fake Image, Right Text

Antipov, G., Baccouche, M., & Dugelay, J. L. (2017). “Face Aging With Conditional Generative Adversarial Networks”. arXiv preprint arXiv:1702.01983.

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.

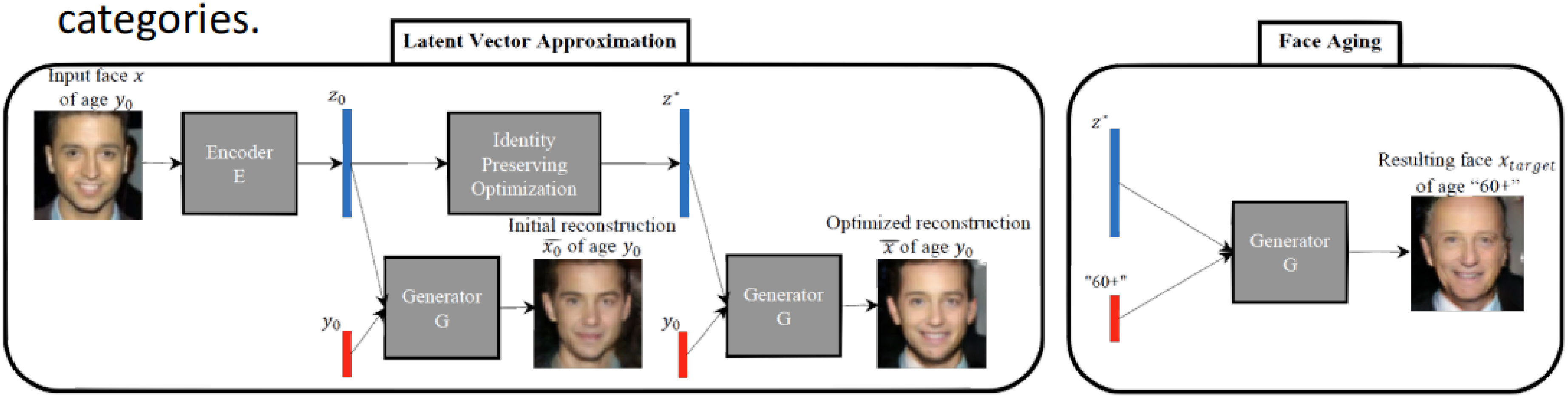


Figure 1 in the original paper.

Liu, Ming-Yu, and Oncel Tuzel. "Coupled generative adversarial networks". NIPS (2016).

Coupled GANs

- Architecture

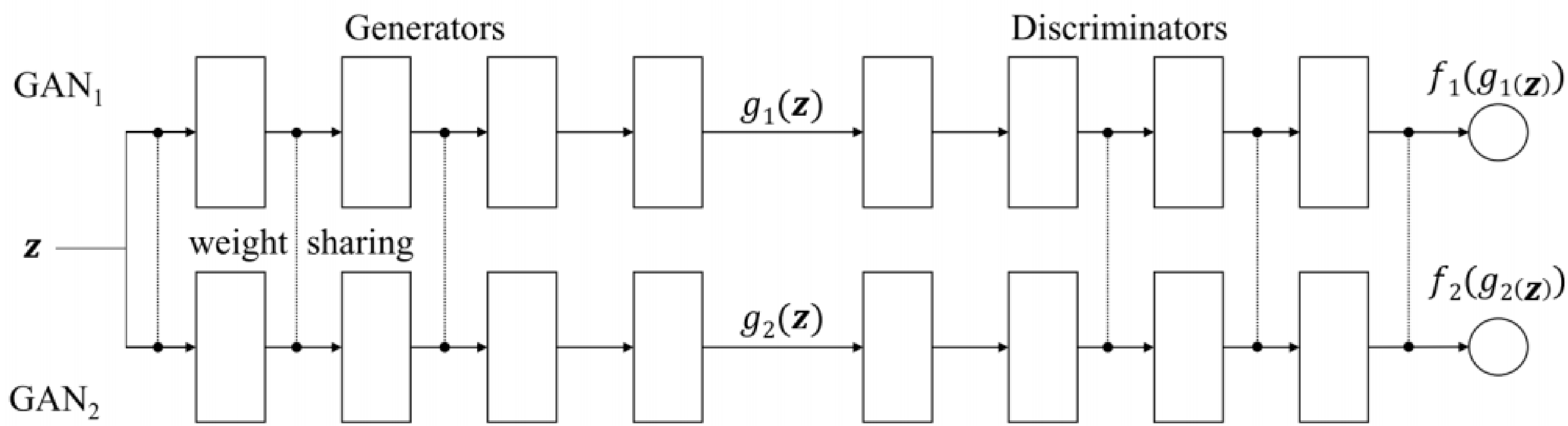


Figure 1 of the original paper.

Weight-sharing constraints the network to learn a *joint distribution* without corresponding supervision.

Coupled GANs

- Some examples of generating facial images across different feature domains.
- Corresponding images in a column are generate from the same latent code z

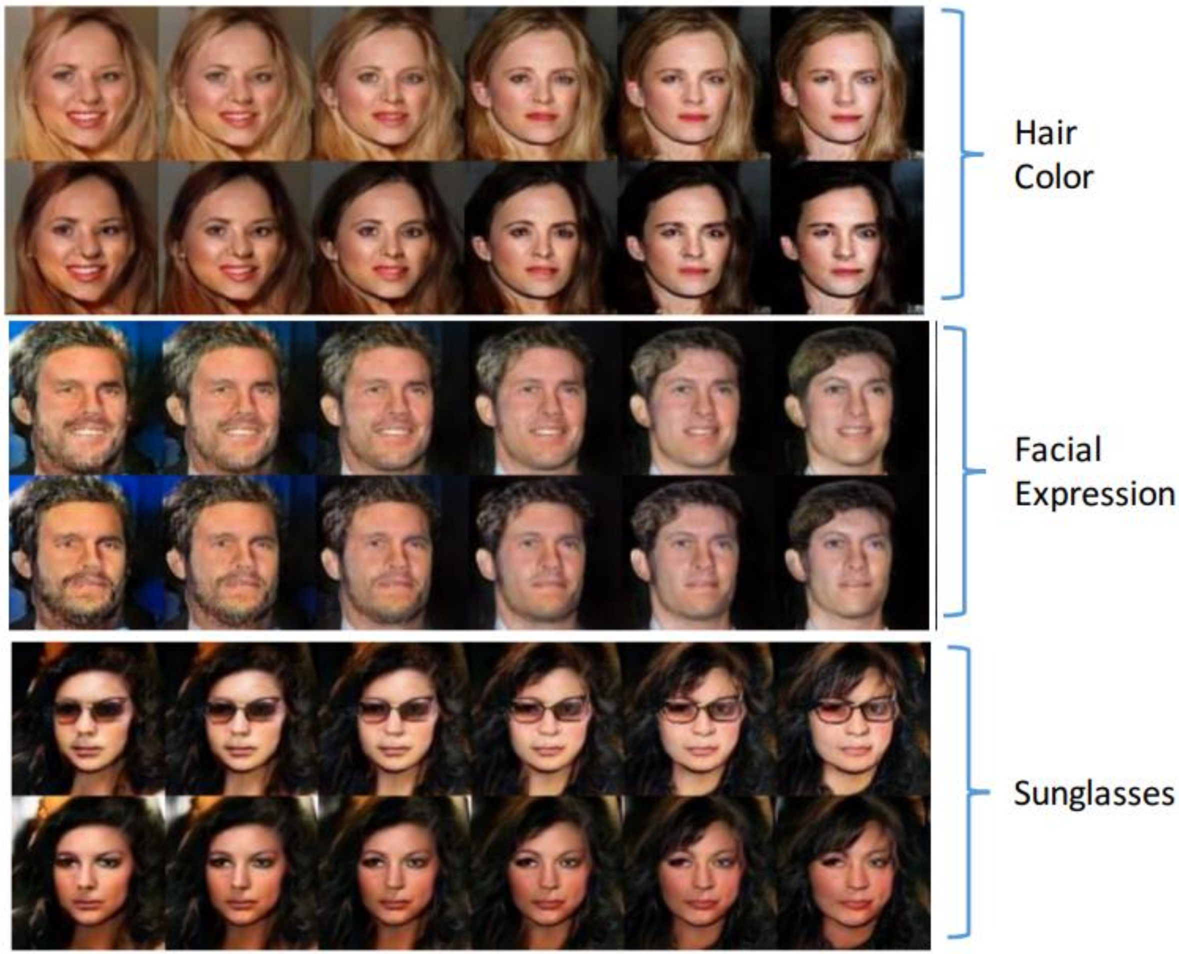


Figure 4 in the original paper.

Denton, E.L., Chintala, S. and Fergus, R., 2015. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”. NIPS (2015)

Laplacian Pyramid of Adversarial Networks

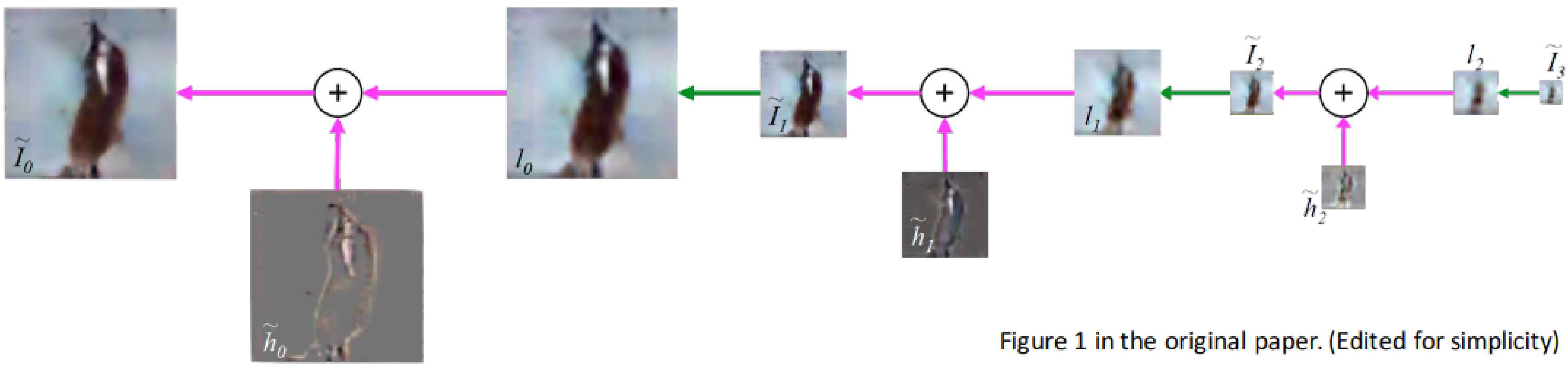


Figure 1 in the original paper. (Edited for simplicity)

- Based on the Laplacian Pyramid representation of images. (1983)
- Generate high resolution (dimension) images by using a hierarchical system of GANs
- Iteratively increase image resolution and quality.

Laplacian Pyramid of Adversarial Networks

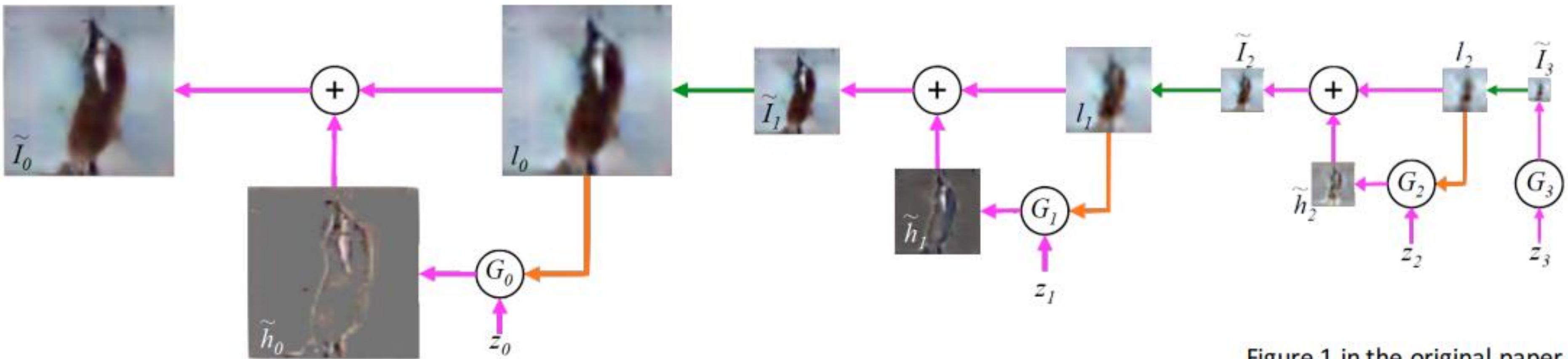


Figure 1 in the original paper.

Image Generation using a LAPGAN

- Generator G_3 generates the base image $\widetilde{I_3}$ from random noise input z_3.
- Generators (G_2, G_1, G_0) iteratively generate the *difference image* $(\widehat{h})$ conditioned on previous small image (l).
- This *difference image* is added to an up-scaled version of previous smaller image.

Laplacian Pyramid of Adversarial Networks

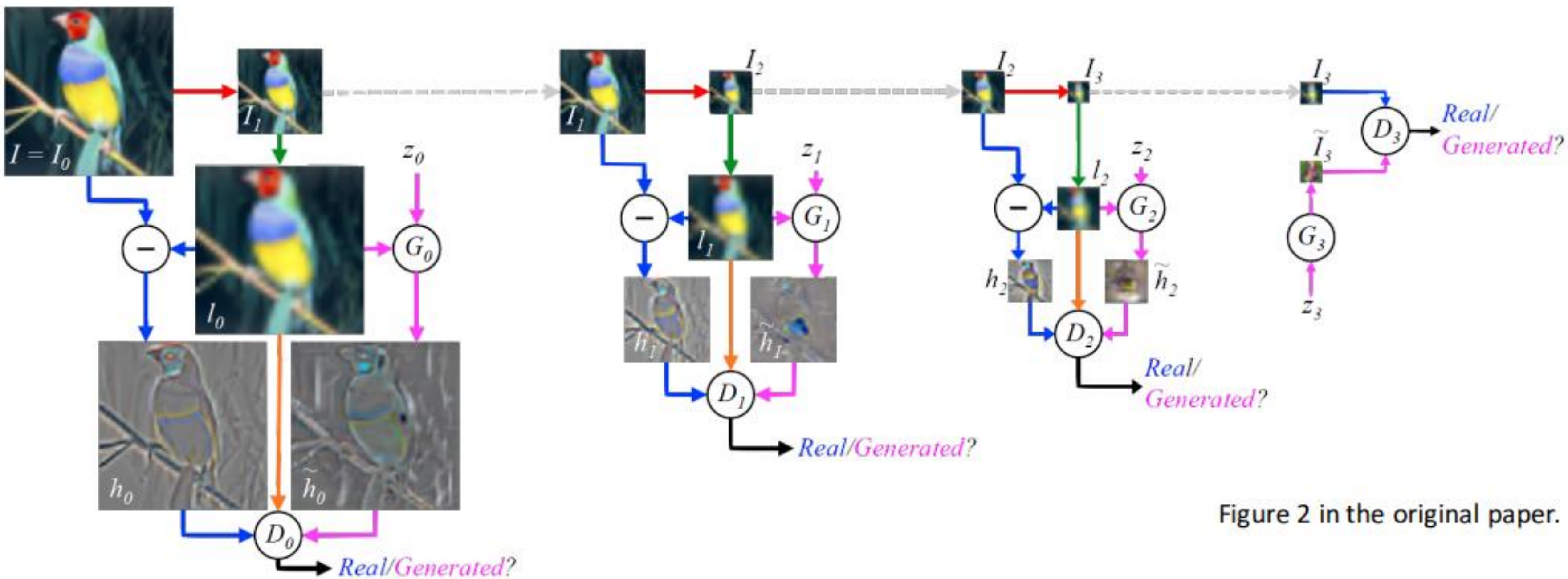


Figure 2 in the original paper.

Training Procedure:

Models at each level are trained independently to learn the required representation.