Generative Adversarial Networks

presented by Ian Goodfellow

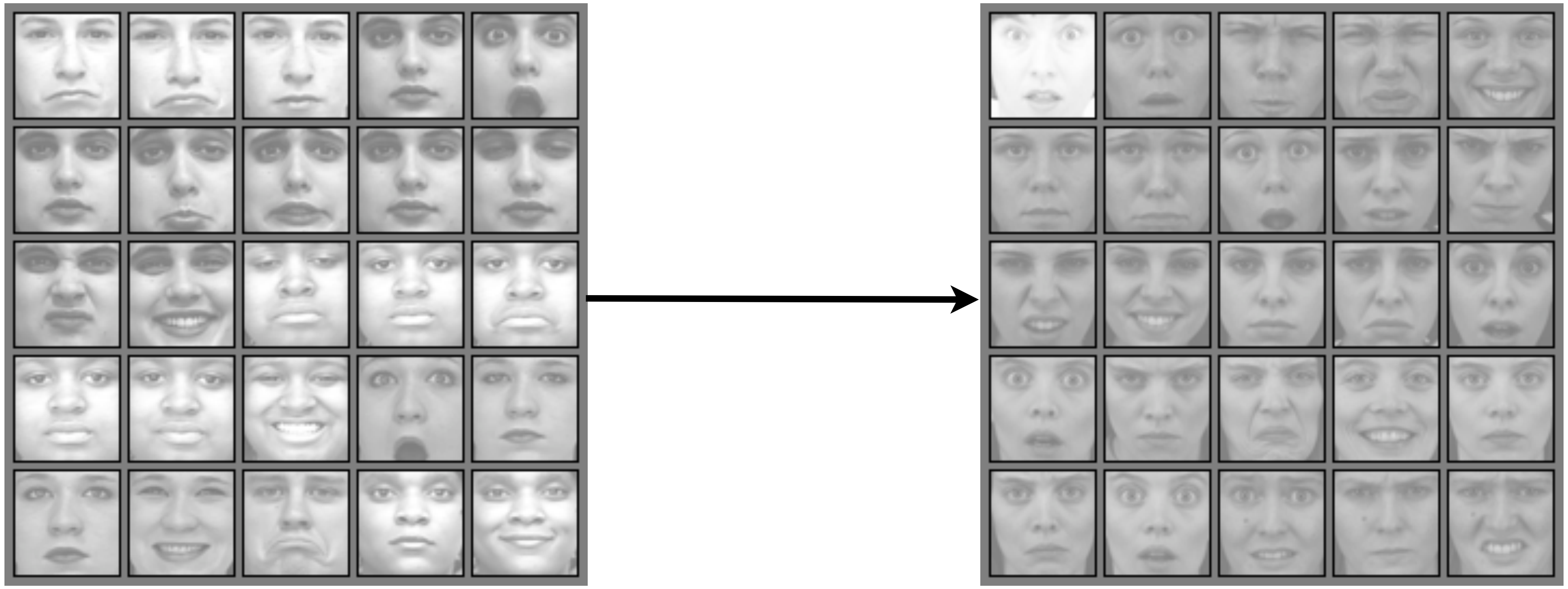
presentation co-developed with Aaron Courville

In today’s talk…

- “Generative Adversarial Networks” Goodfellow et al., NIPS 2014
- “Conditional Generative Adversarial Nets” Mirza and Osindero, NIPS Deep Learning Workshop 2014
- “On Distinguishability Criteria for Estimating Generative Models” Goodfellow, ICLR Workshop 2015
- “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks” Denton, Chintala, et al., ArXiv 2015

Generative modeling

- Have training examples $x \sim p_{\text{data}}(x)$
- Want a model that can draw samples: $x \sim p_{\text{model}}(x)$
- Where $p_{\text{model}} \approx p_{\text{data}}$



$x \sim p_{\text{data}}(x)$ $\qquad\qquad$ $x \sim p_{\text{model}}(x)$

Why generative models?

- Conditional generative models
  - Speech synthesis: Text ⇒ Speech
  - Machine Translation: French ⇒ English
    - French: Si mon tonton tond ton tonton, ton tonton sera tondu.
    - English: If my uncle shaves your uncle, your uncle will be shaved
  - Image ⇒ Image segmentation
- Environment simulator
  - Reinforcement learning
  - Planning
- Leverage unlabeled data?

Maximum likelihood: the dominant approach

- ML objective function

$$\theta^* = \max_{\theta} \frac{1}{m} \sum_{i=1}^{m} \log p\left(x^{(i)}; \theta\right)$$

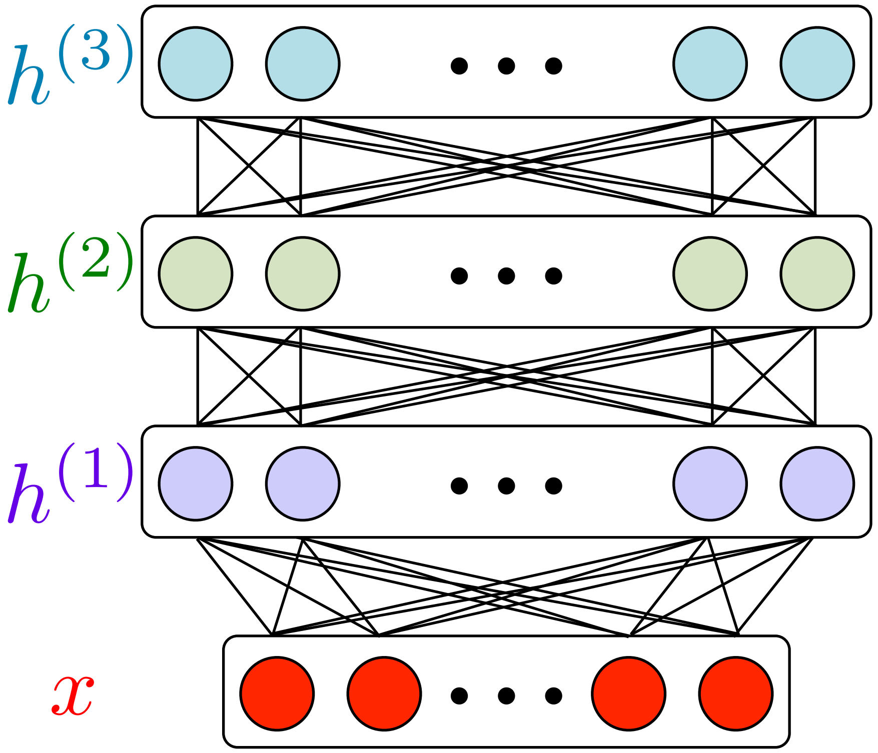
Undirected graphical models

- Flagship undirected graphical model: **Deep Boltzmann machines**
- Several "hidden layers" h

$$p(h, x) = \frac{1}{Z}\tilde{p}(h, x)$$

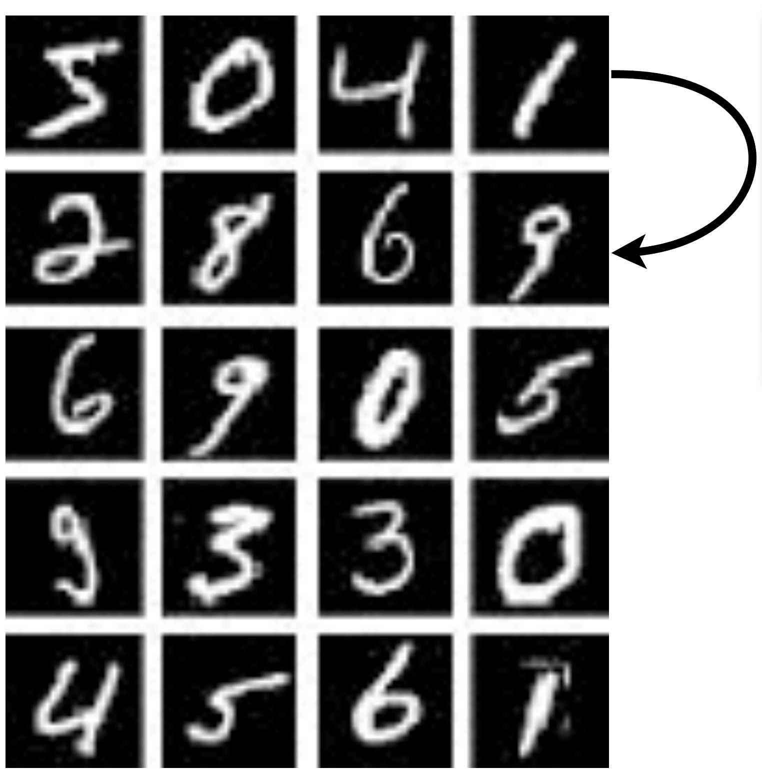
$$\tilde{p}(h, x) = \exp(-E(h, x))$$

$$Z = \sum_{h,x} \tilde{p}(h, x)$$



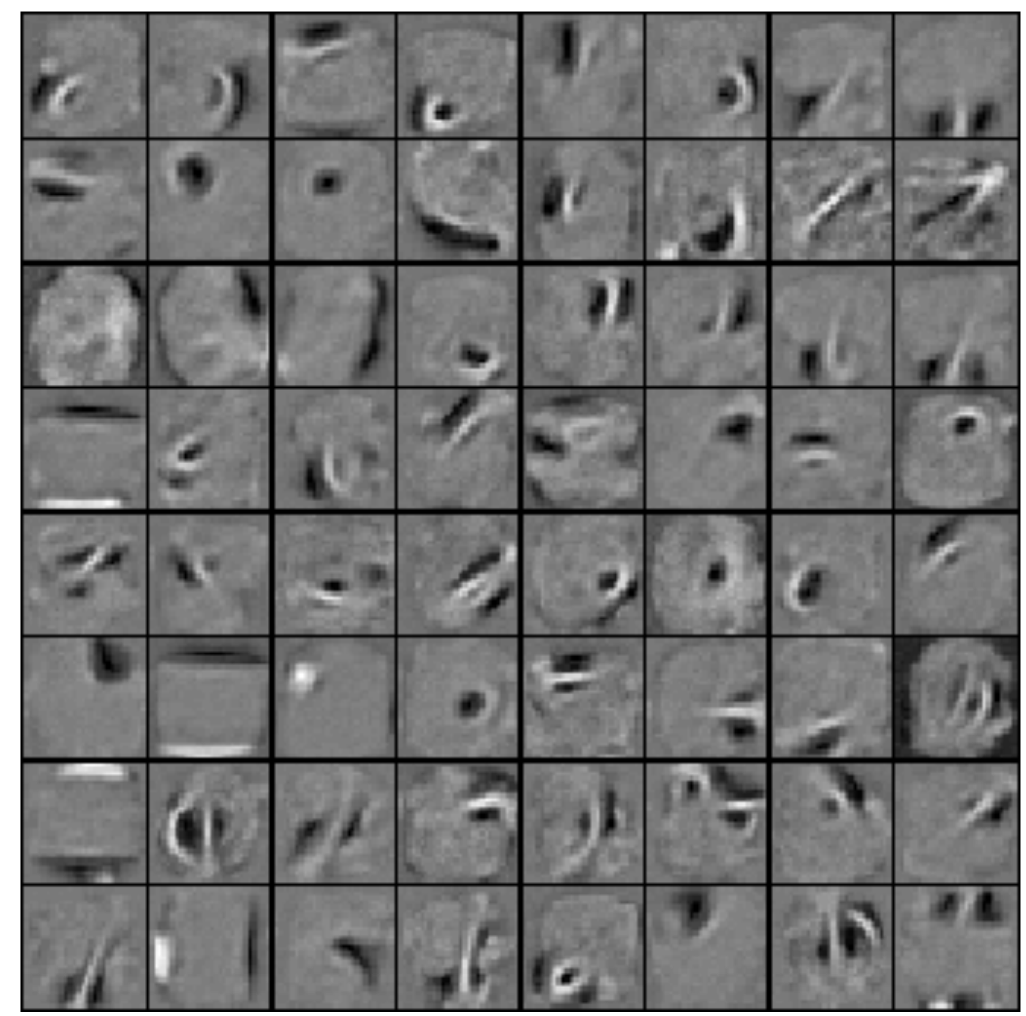
Boltzmann Machines: disadvantage

- Model is badly parameterized for learning high quality samples: peaked distributions -> slow mixing
- Why poor mixing?



MNIST dataset

Coordinated flipping of low-level features



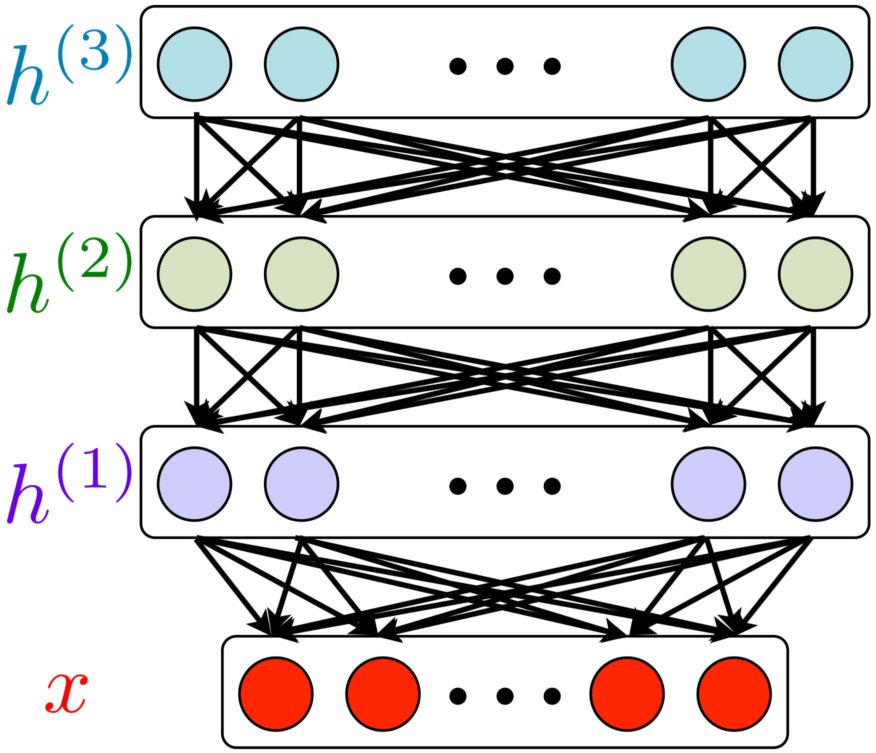
1st layer features (RBM)

Directed graphical models

$$p(x, h) = p(x \mid h^{(1)})p(h^{(1)} \mid h^{(2)}) \ldots p(h^{(L-1)} \mid h^{(L)})p(h^{(L)})$$

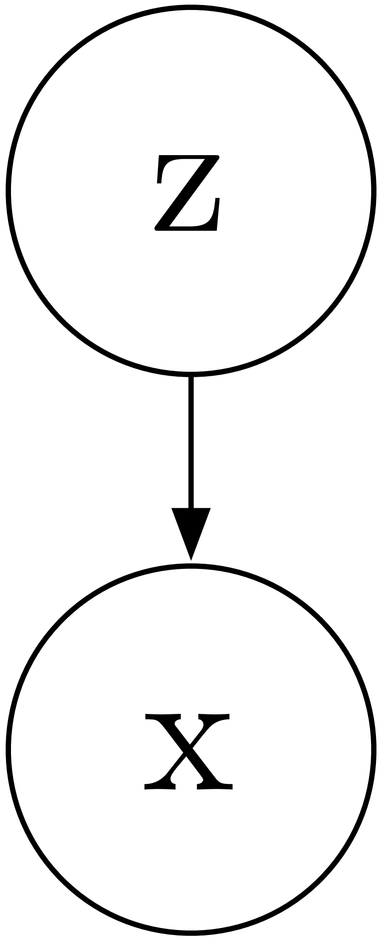
$$\frac{d}{d\theta_i} \log p(x) = \frac{1}{p(x)} \frac{d}{d\theta_i} p(x)$$

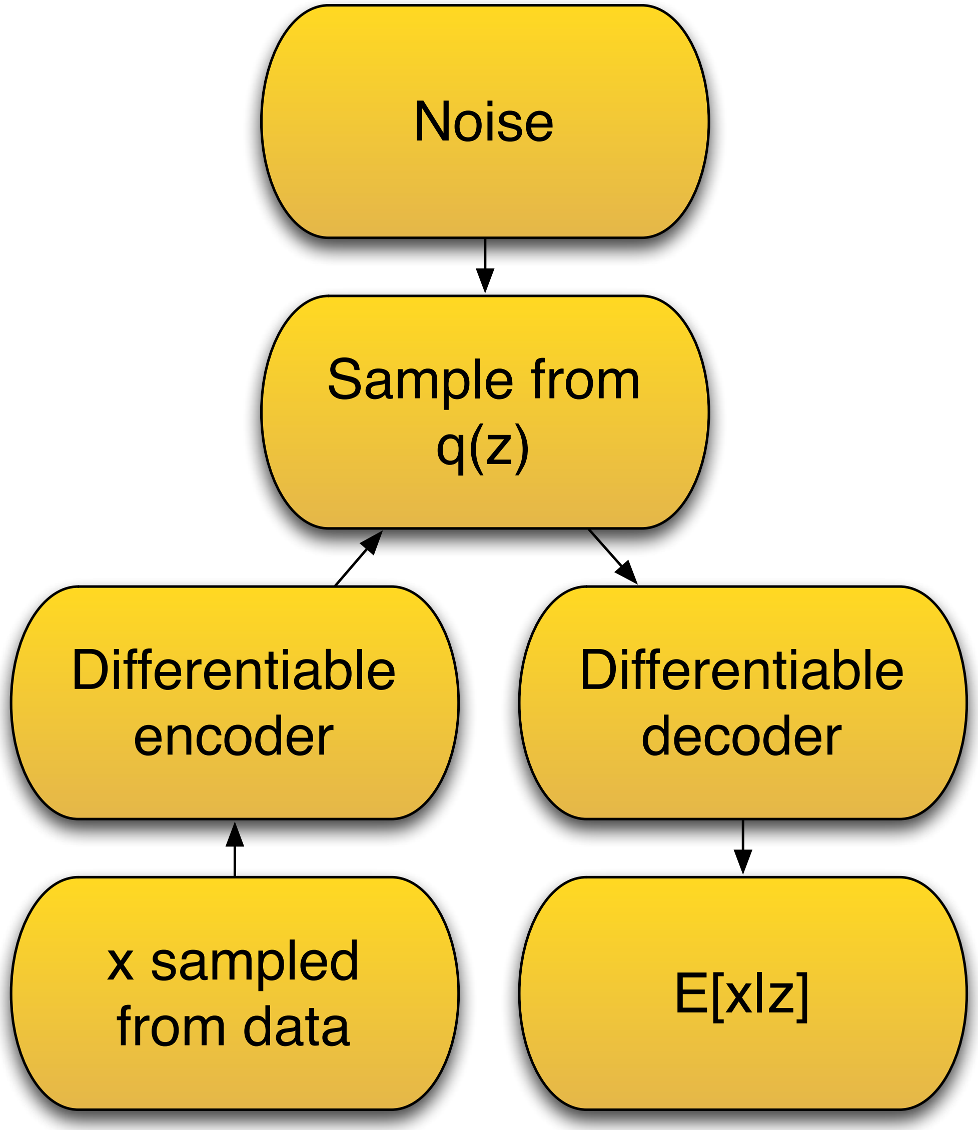
$$p(x) = \sum_h p(x \mid h)p(h)$$



- Two problems:
  1. Summation over exponentially many states in ***h***
  2. Posterior inference, i.e. calculating ***p*(*h* | *x*)**, is intractable.

Variational Autoencoder



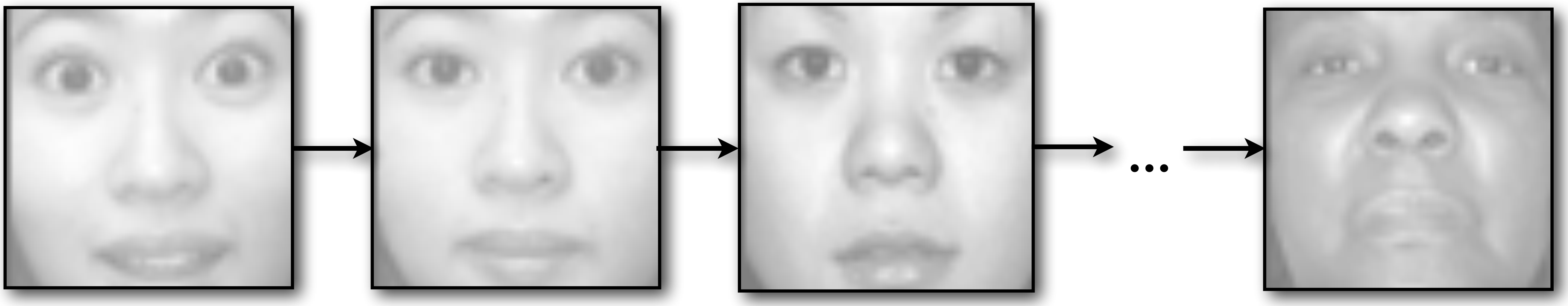


$$\text{Maximize} \log p(x) - D_{KL}\left(q(x) \| p(z \mid x)\right)$$

(Kingma and Welling, 2014, Rezende et al 2014)

Generative stochastic networks

- General strategy: Do not write a formula for $p(x)$, just learn to sample incrementally.



- Main issue: Subject to some of the same constraints on mixing as undirected graphical models.

(Bengio et al 2013)

Generative adversarial networks

- Don't write a formula for $p(x)$, just learn to sample directly.
- No Markov Chain
- No variational bound
- How? By playing a game.

Game theory: the basics

- N>1 players
- Clearly defined set of actions each player can take
- Clearly defined relationship between actions and outcomes
- Clearly defined value of each outcome
- Can't control the other player's actions

Two-player zero-sum game

- Your winnings + your opponent's winnings = 0
- Minimax theorem: a rational strategy exists for all such finite games

Two-player zero-sum game

- Strategy: specification of which moves you make in which circumstances.
- Equilibrium: each player's strategy is the best possible for their opponent's strategy.
- Example: Rock-paper-scissors:
  - *Mixed strategy equilibrium*
  - Choose your action at random

You \ Your opponent	Rock	Paper	Scissors
Rock	0	-1	1
Paper	1	0	-1
Scissors	-1	1	0

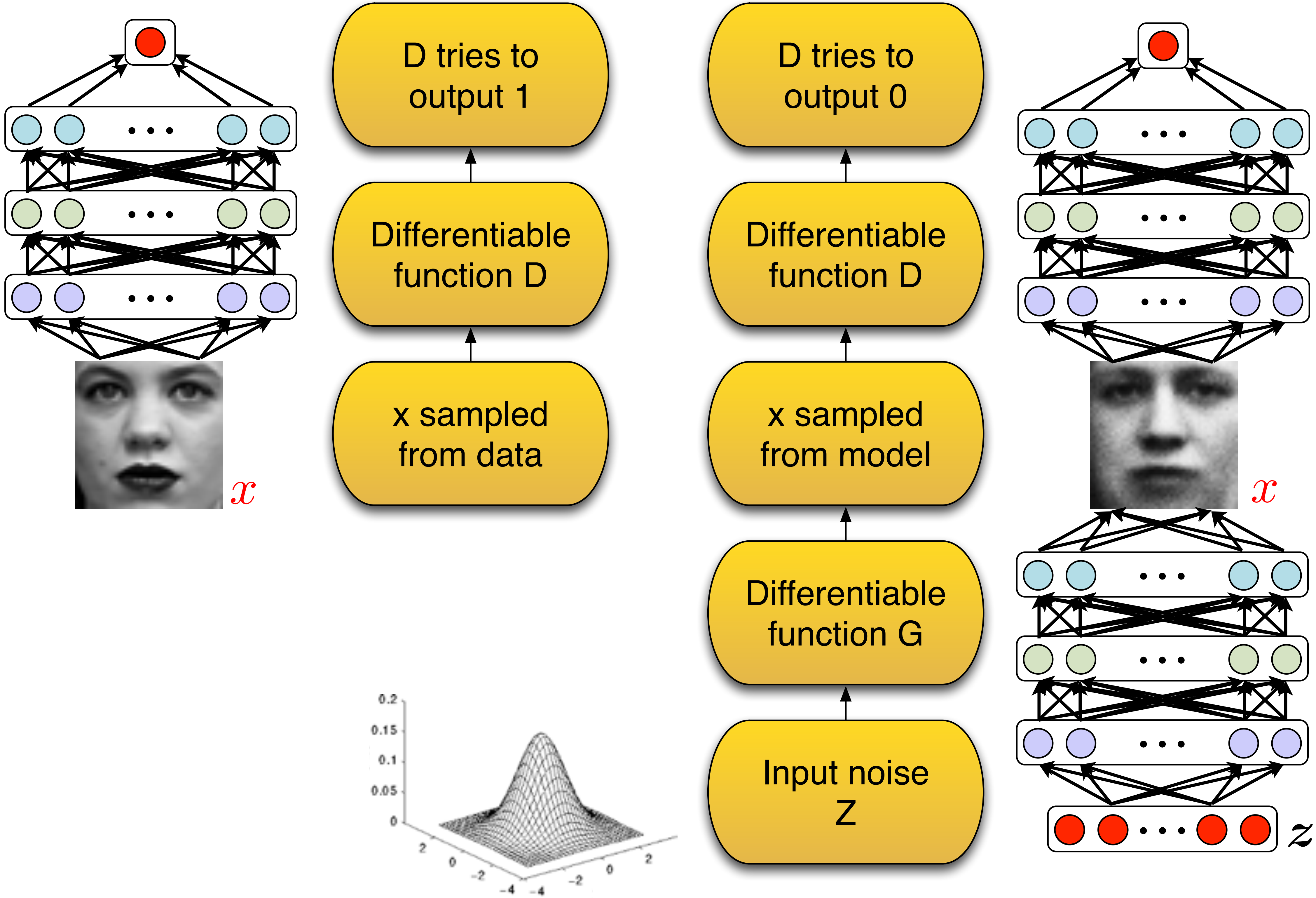
Adversarial nets framework

- A game between two players:
  1. Discriminator D
  2. Generator G

- D tries to discriminate between:
  - A sample from the data distribution.
  - And a sample from the generator G.

- G tries to “trick” D by generating samples that are hard for D to distinguish from data.

Adversarial nets framework



Zero-sum game

- Minimax value function:

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

Generator pushes down

Discriminator pushes up

Discriminator's ability to recognize data as being real

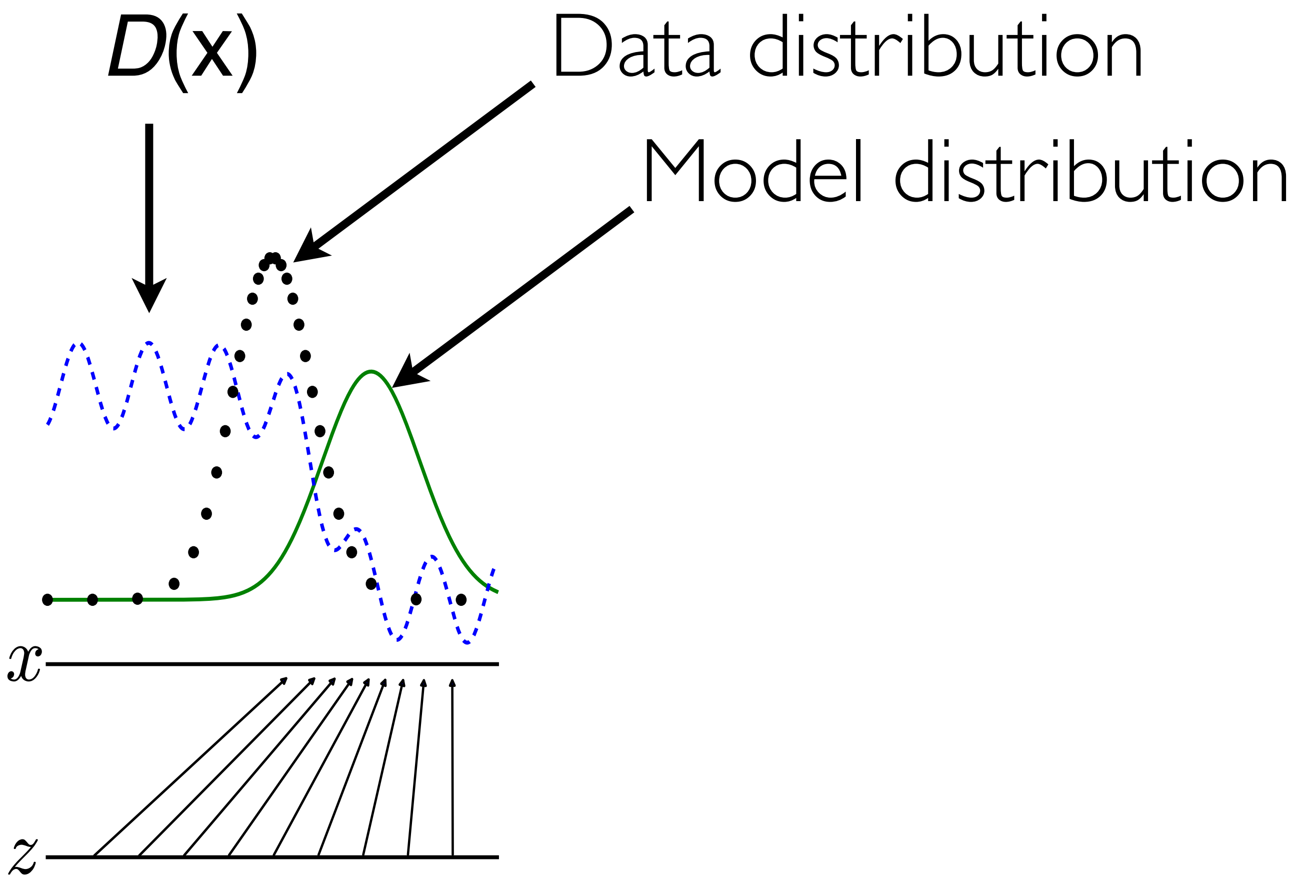
Discriminator's ability to recognize generator samples as being fake

Discriminator strategy

- Optimal strategy for any $p_{\text{model}}(x)$ is always

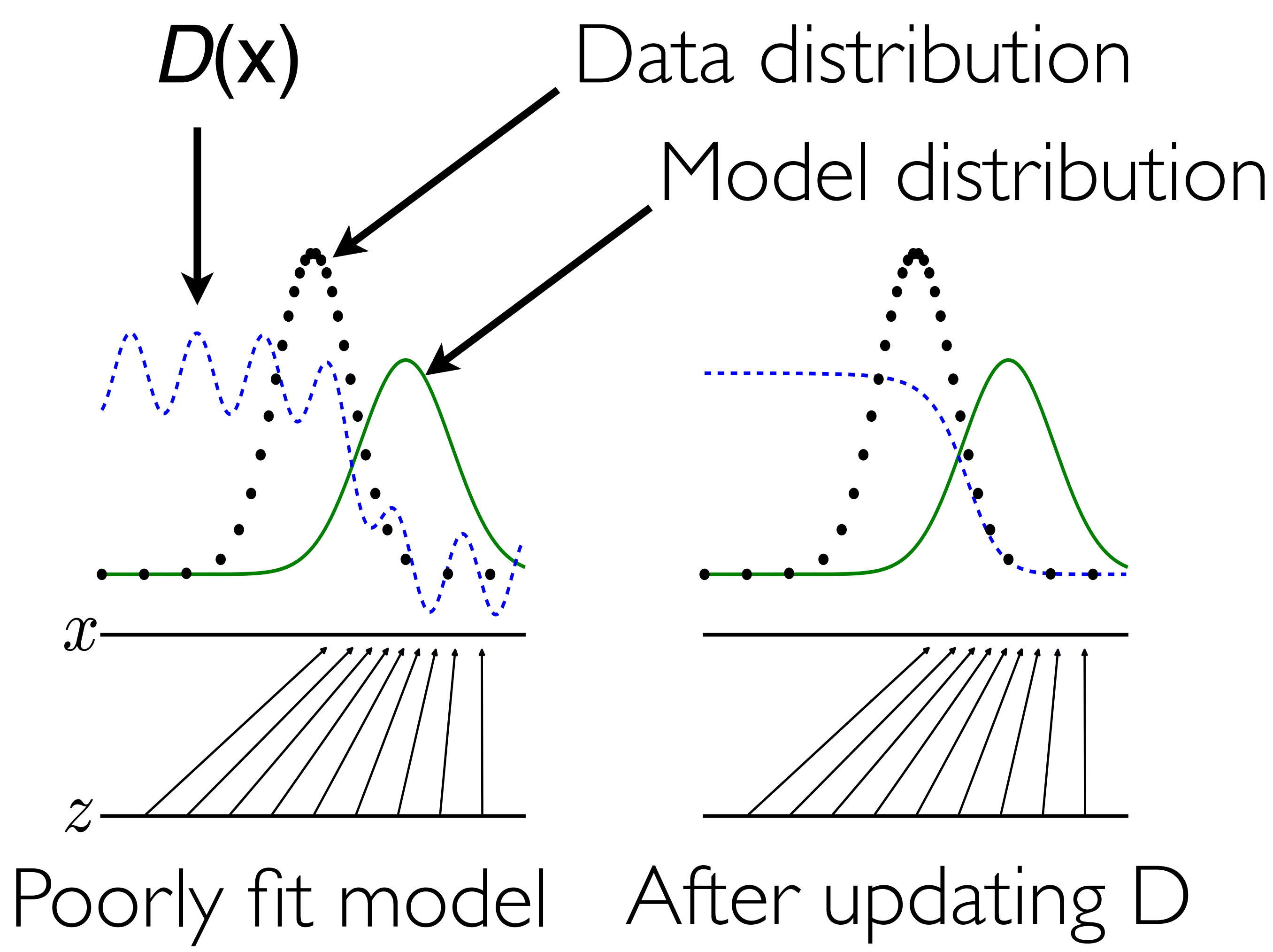
$$D(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_{\text{model}}(x)}$$

Learning process

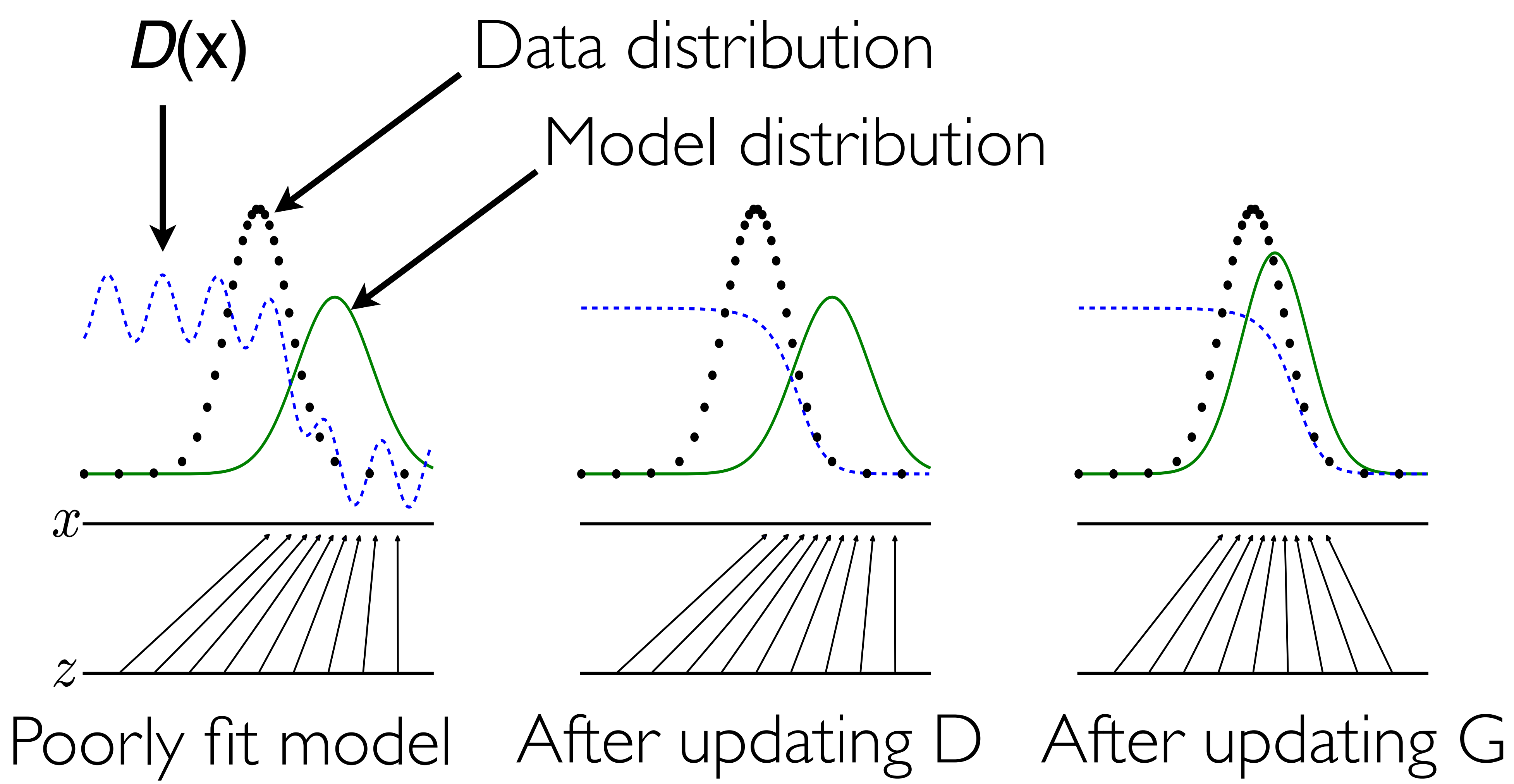


Poorly fit model

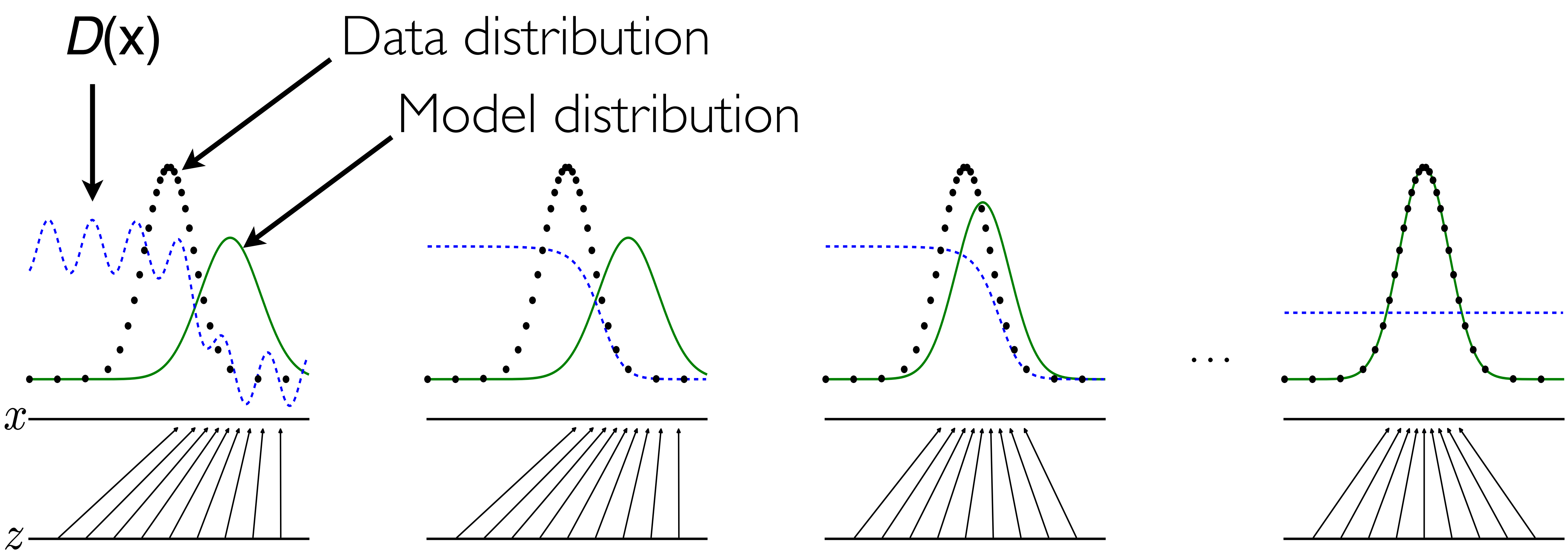
Learning process



Learning process



Learning process



Poorly fit model | After updating D | After updating G | Mixed strategy equilibrium

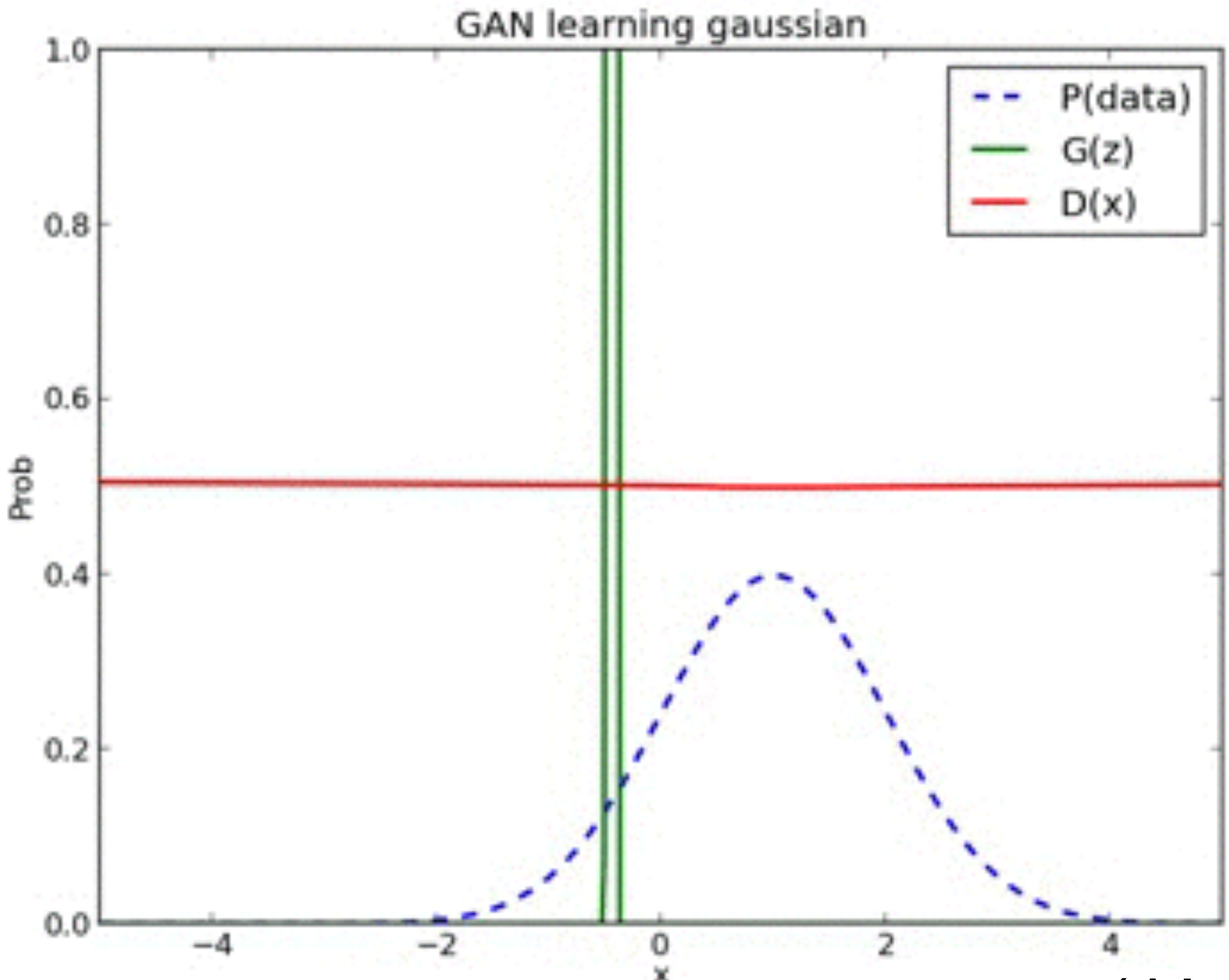
Theoretical properties

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

- Theoretical properties (assuming infinite data, infinite model capacity, direct updating of generator's distribution):
  - Unique global optimum.
  - Optimum corresponds to data distribution.
  - Convergence to optimum guaranteed.

In practice: no proof that SGD *converges*

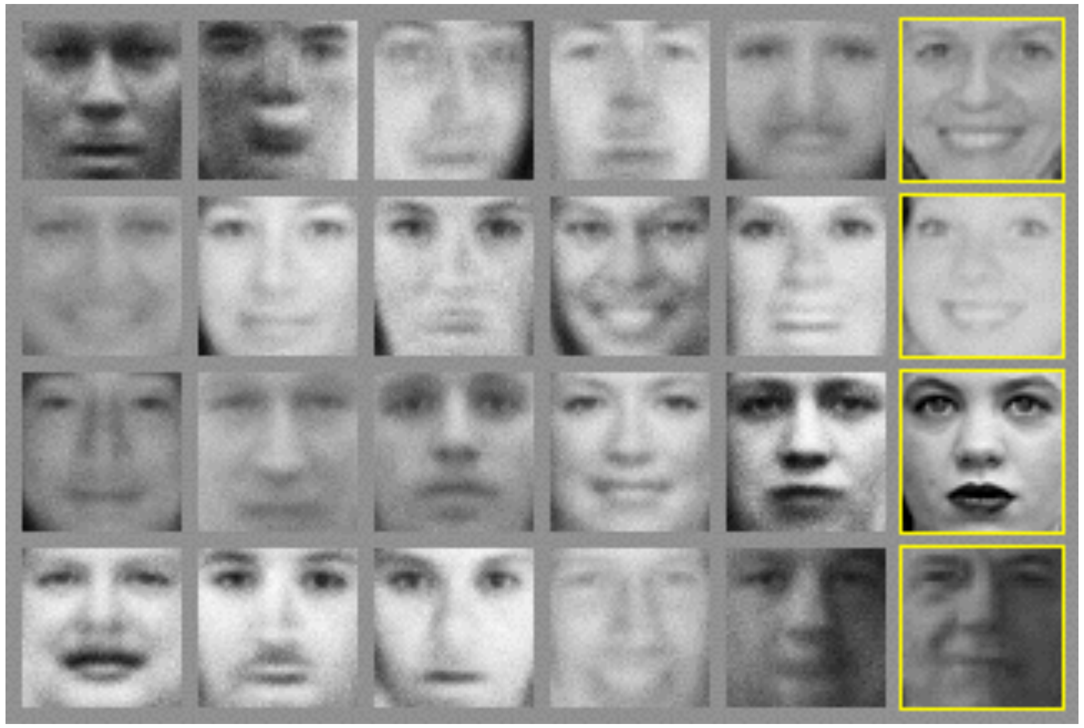
Oscillation



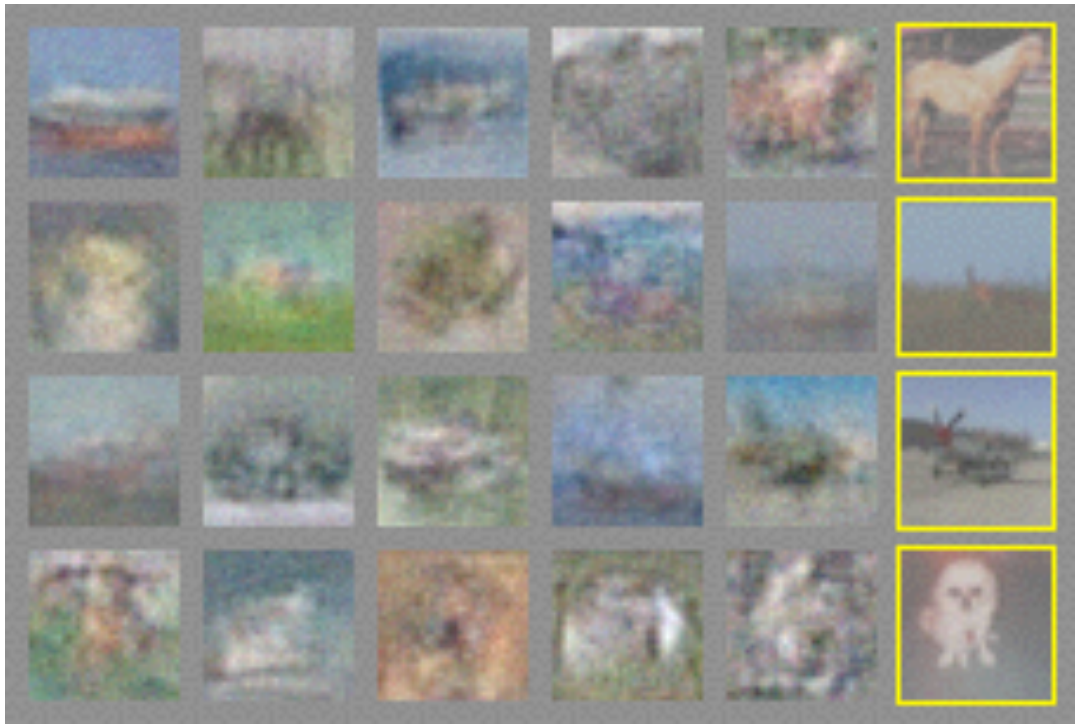
(Alec Radford)

Visualization of model samples

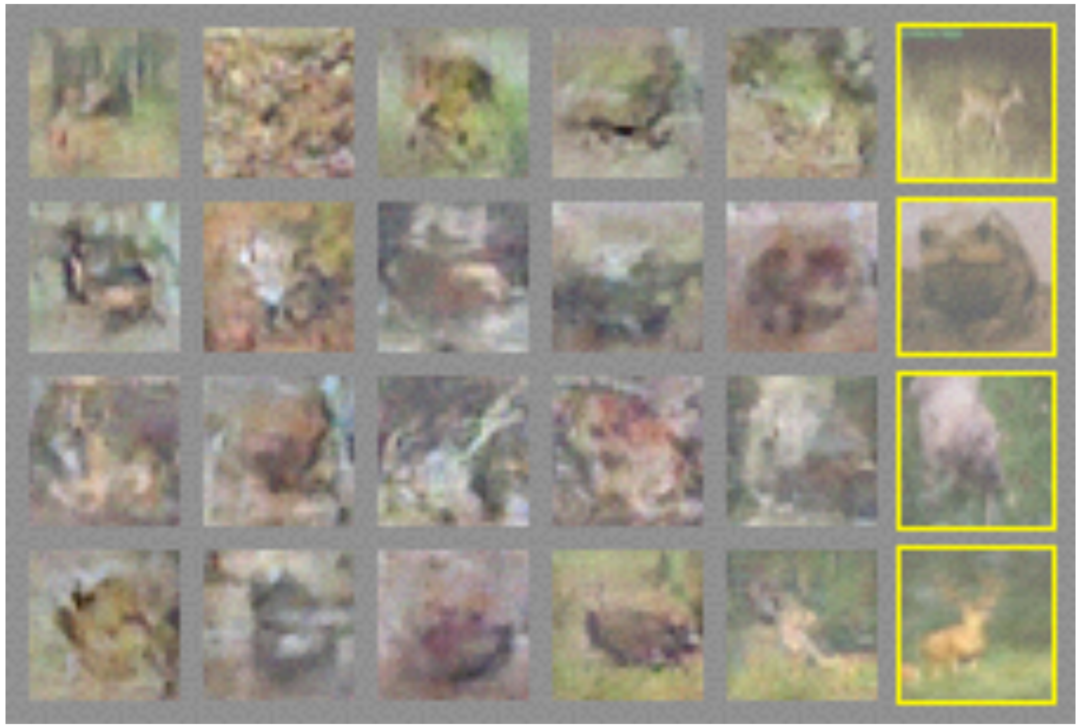
MNIST



TFD

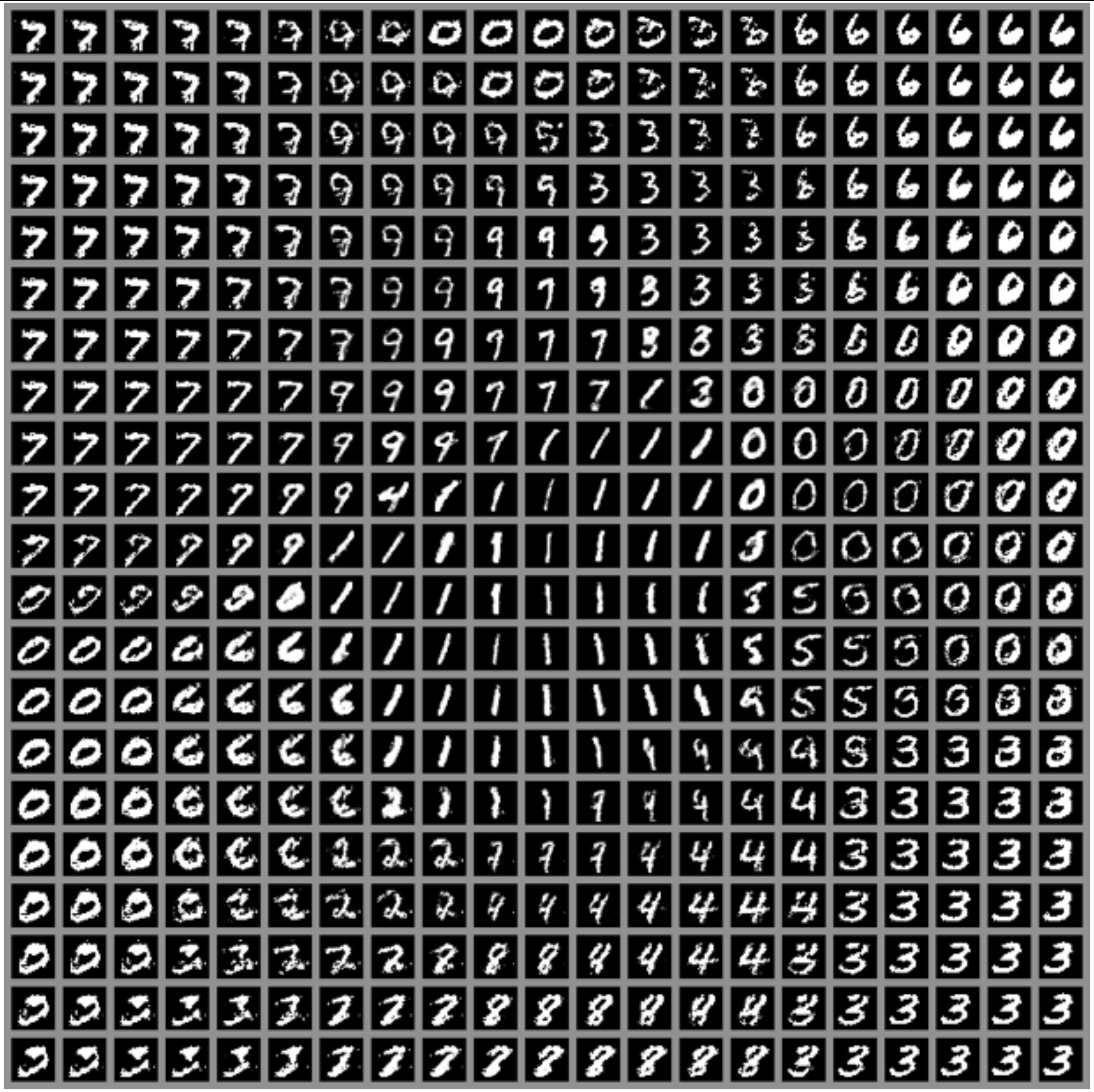


CIFAR-10 (fully connected)



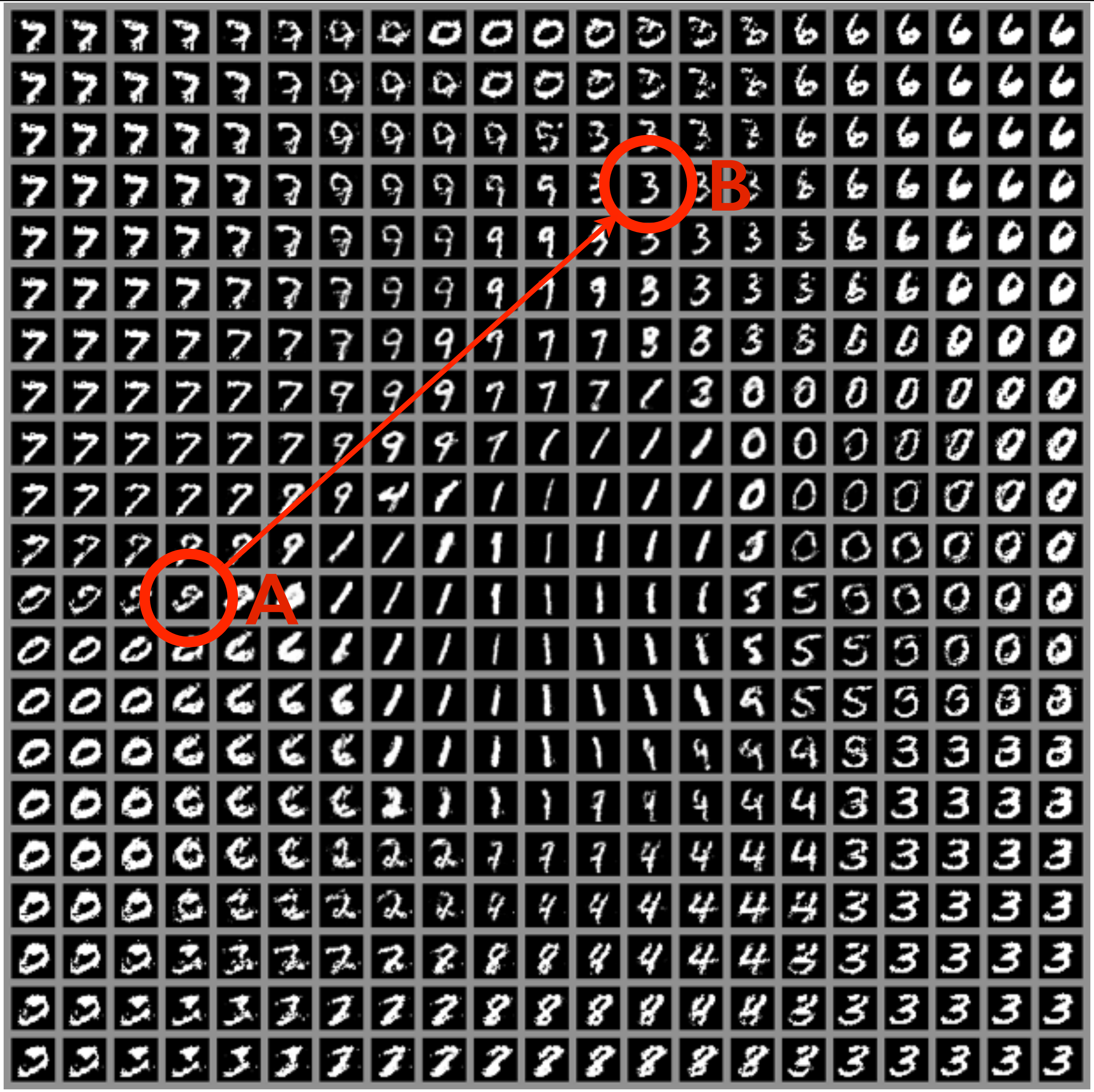
CIFAR-10 (convolutional)

Learned 2-D manifold of MNIST

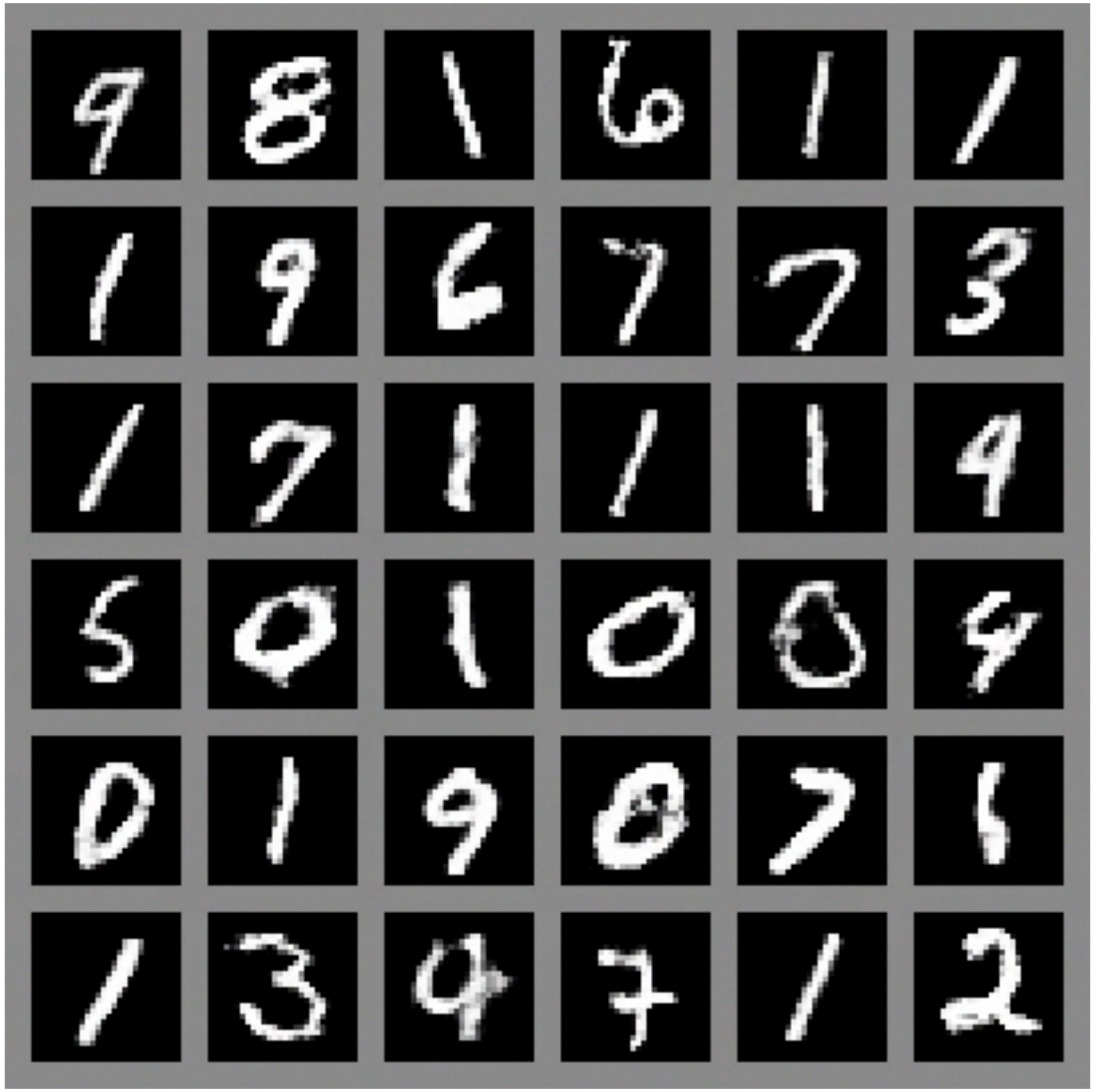


Visualizing trajectories

1. Draw sample (A)
2. Draw sample (B)
3. Simulate samples along the path between A and B
4. Repeat steps 1-3 as desired.



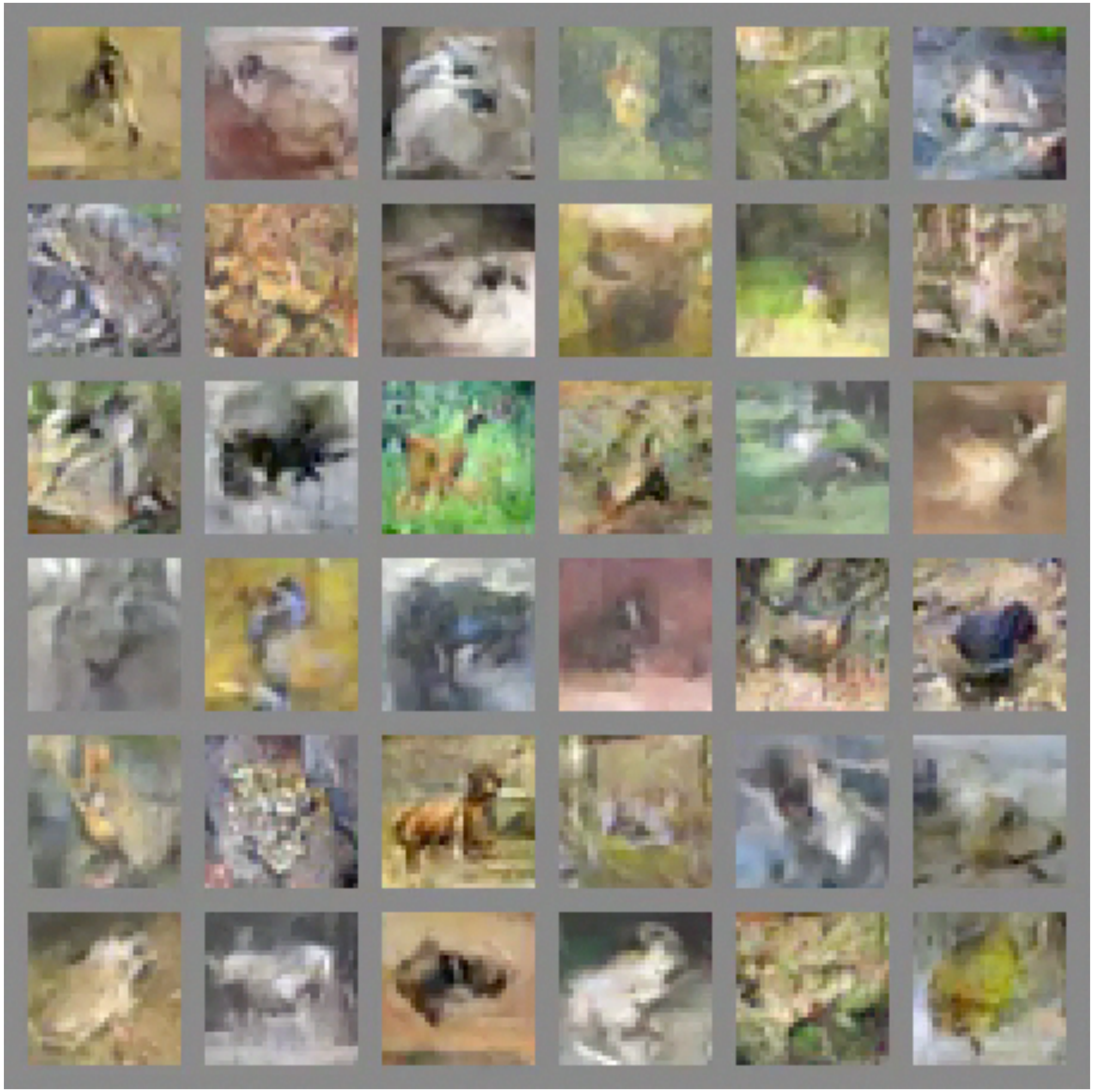
Visualization of model trajectories



MNIST digit dataset

Toronto Face Dataset (TFD)

Visualization of model trajectories



CIFAR-10
(convolutional)

GANs vs VAEs

- Both use backprop through continuous random number generation
- VAE:
  - generator gets direct output target
  - need REINFORCE to do discrete latent variables
  - possible underfitting due to variational approximation
  - gets global image composition right but blurs details
- GAN:
  - generator never sees the data
  - need REINFORCE to do discrete visible variables
  - possible underfitting due to non-convergence
  - gets local image features right but not global structure

VAE + GAN

VAE

VAE+GAN

-Reduce VAE blurriness
-Reduce GAN oscillation

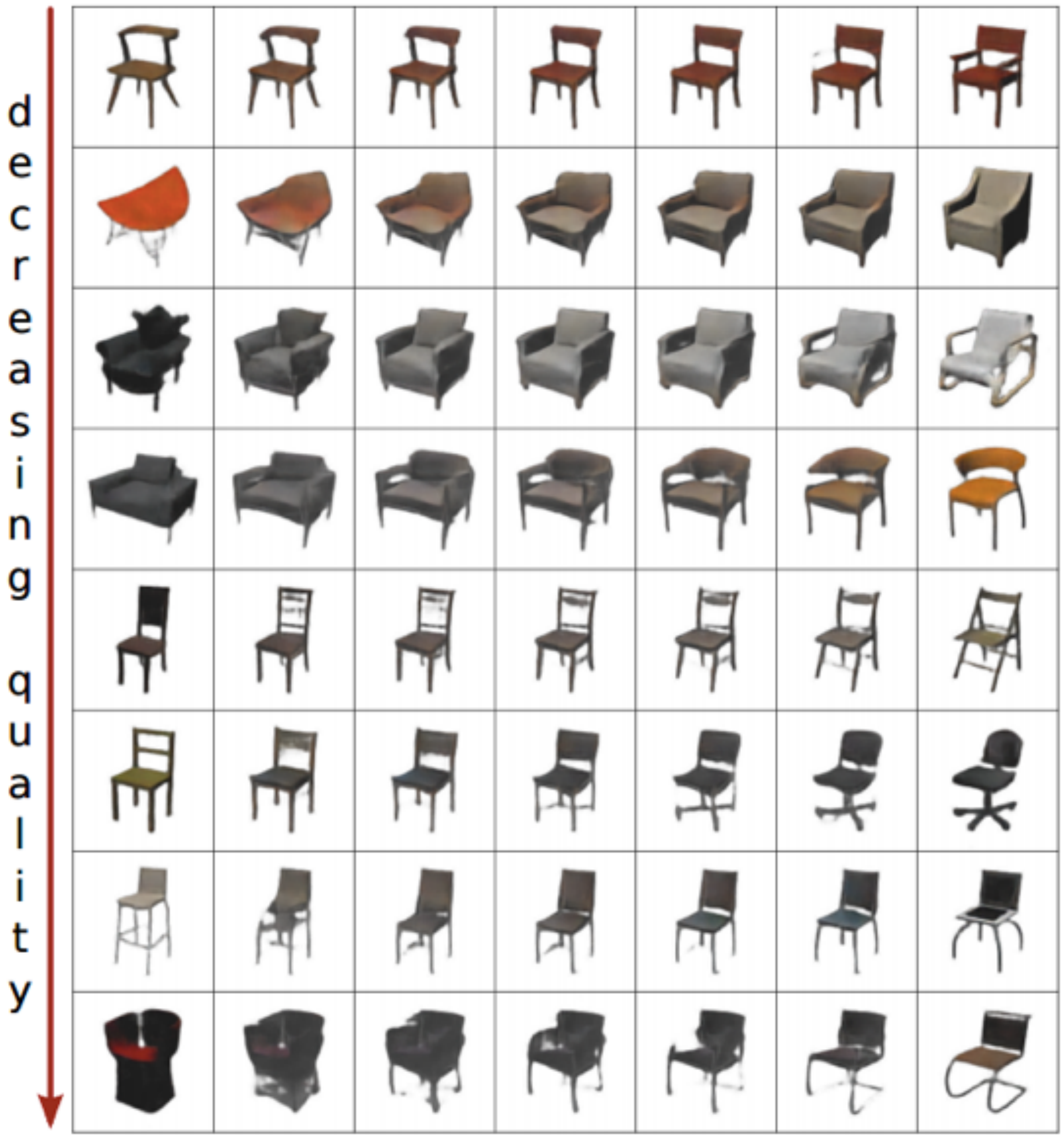
(Alec Radford, 2015)

MMD-based generator nets

(Li et al 2015)

(Dziugaite et al 2015)

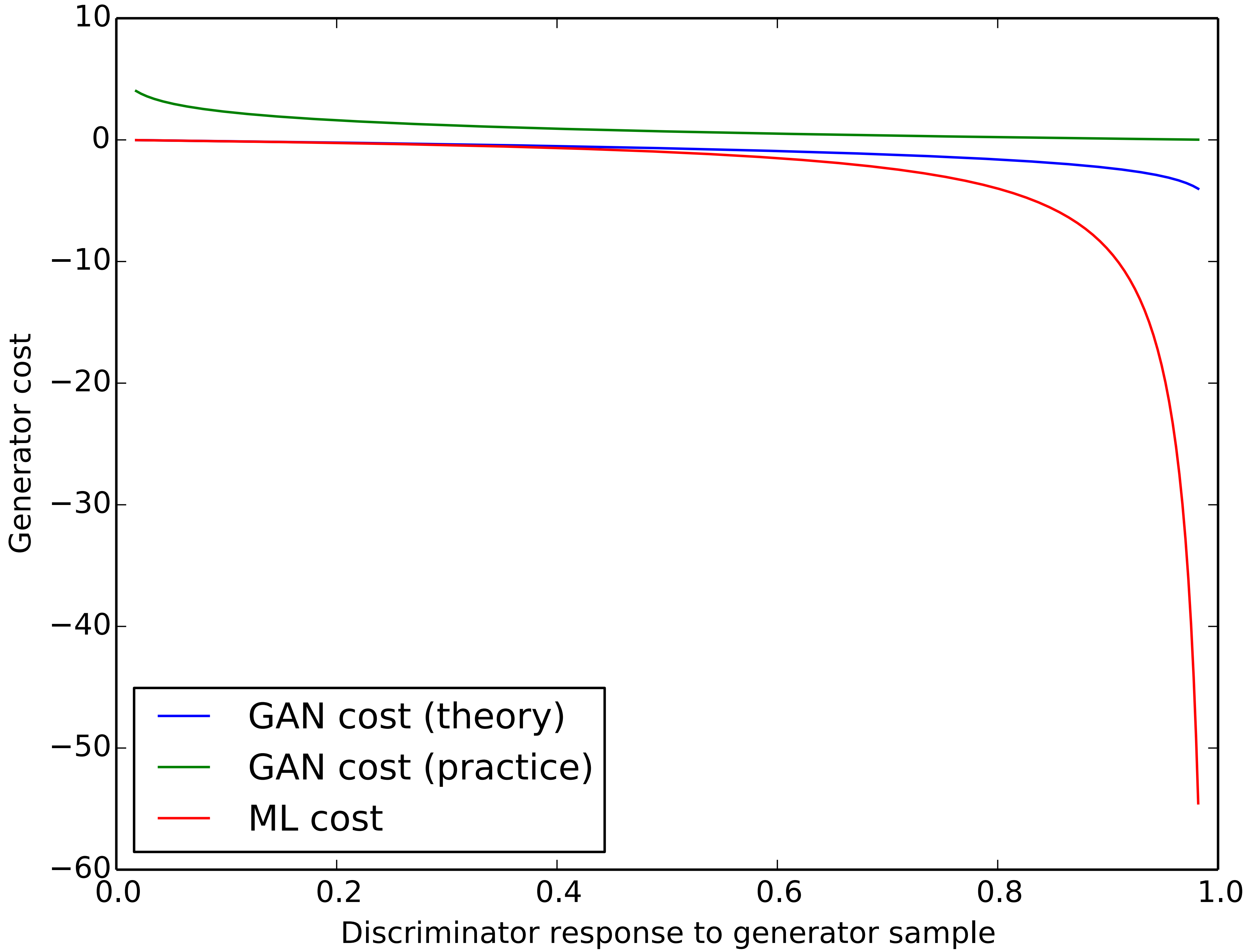
Supervised Generator Nets



Generator nets are powerful—it is our ability to infer a mapping from an unobserved space that is limited.

(Dosovitskiy et al 2014)

General game



Extensions

- Inference net:
  - Learn a network to model p(z | x)
  - Wake/Sleep style approach
    - Sample z from prior
    - Sample x from p(z|x)
    - Learn mapping from x to z
  - Infinite training set!

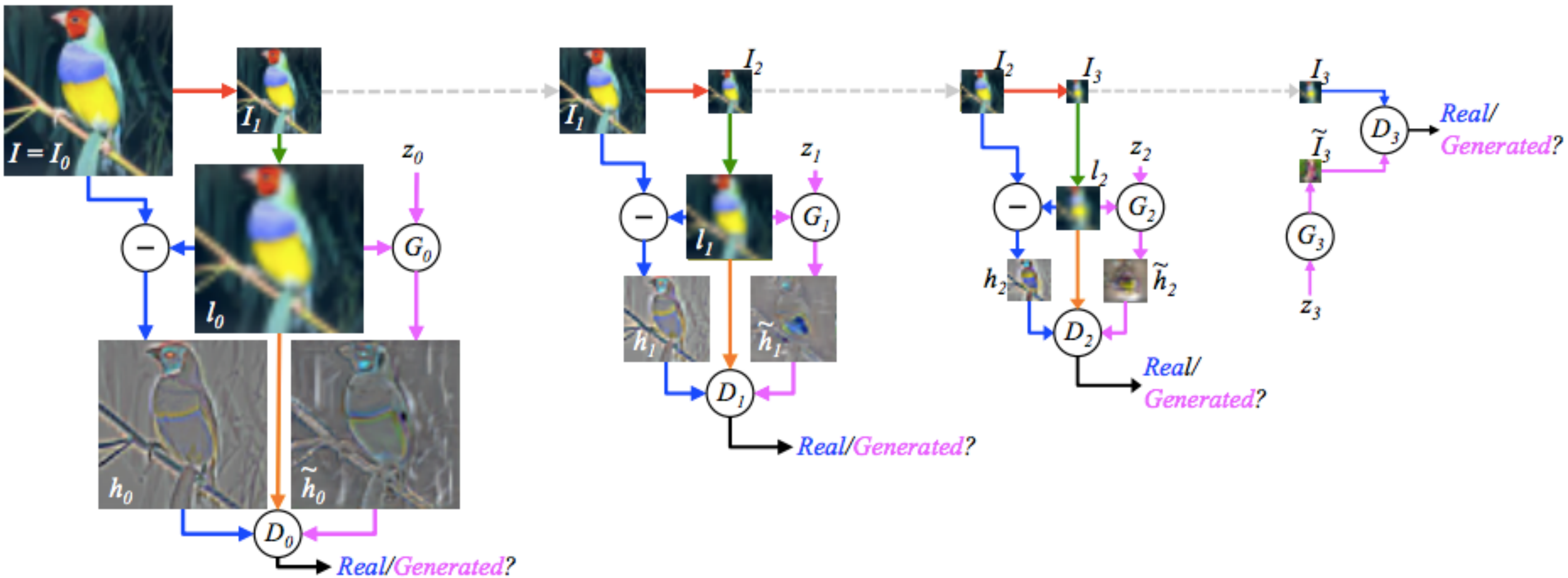
Extensions

- Conditional model:
  - Learn $p(x \mid y)$
  - Discriminator is trained on (x, y) pairs
  - Generator net gets y and z as input
  - Useful for: Translation, speech synth, image segmentation.

	User tags + annotations	Generated tags
	montanha, trem, inverno, frio, people, male, plant life, tree, structures, transport, car	taxi, passenger, line, transportation, railway station, passengers, railways, signals, rail, rails
	food, raspberry, delicious, homemade	chicken, fattening, cooked, peanut, cream, cookie, house made, bread, biscuit, bakes
	water, river	creek, lake, along, near, river, rocky, treeline, valley, woods, waters

(Mirza and Osindero, 2014)

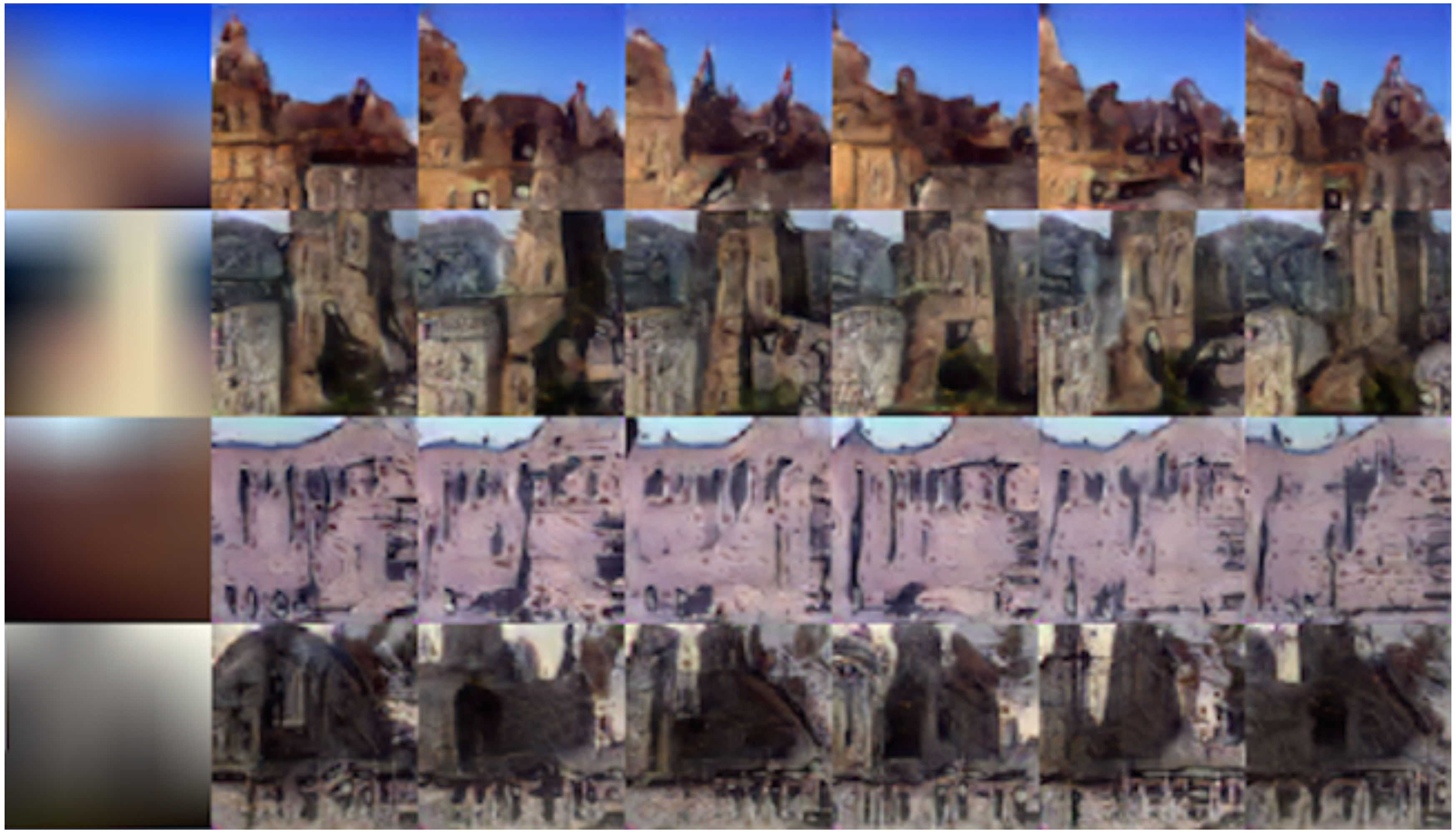
Laplacian Pyramid



(Denton + Chintala, et al 2015)

LAPGAN results

- 40% of samples mistaken *by humans* for real photos



(Denton + Chintala, et al 2015)

Open problems

- Is non-convergence a serious problem in practice?
- If so, how can we prevent non-convergence?
- Is there a better loss function for the generator?

Thank You.

Questions?