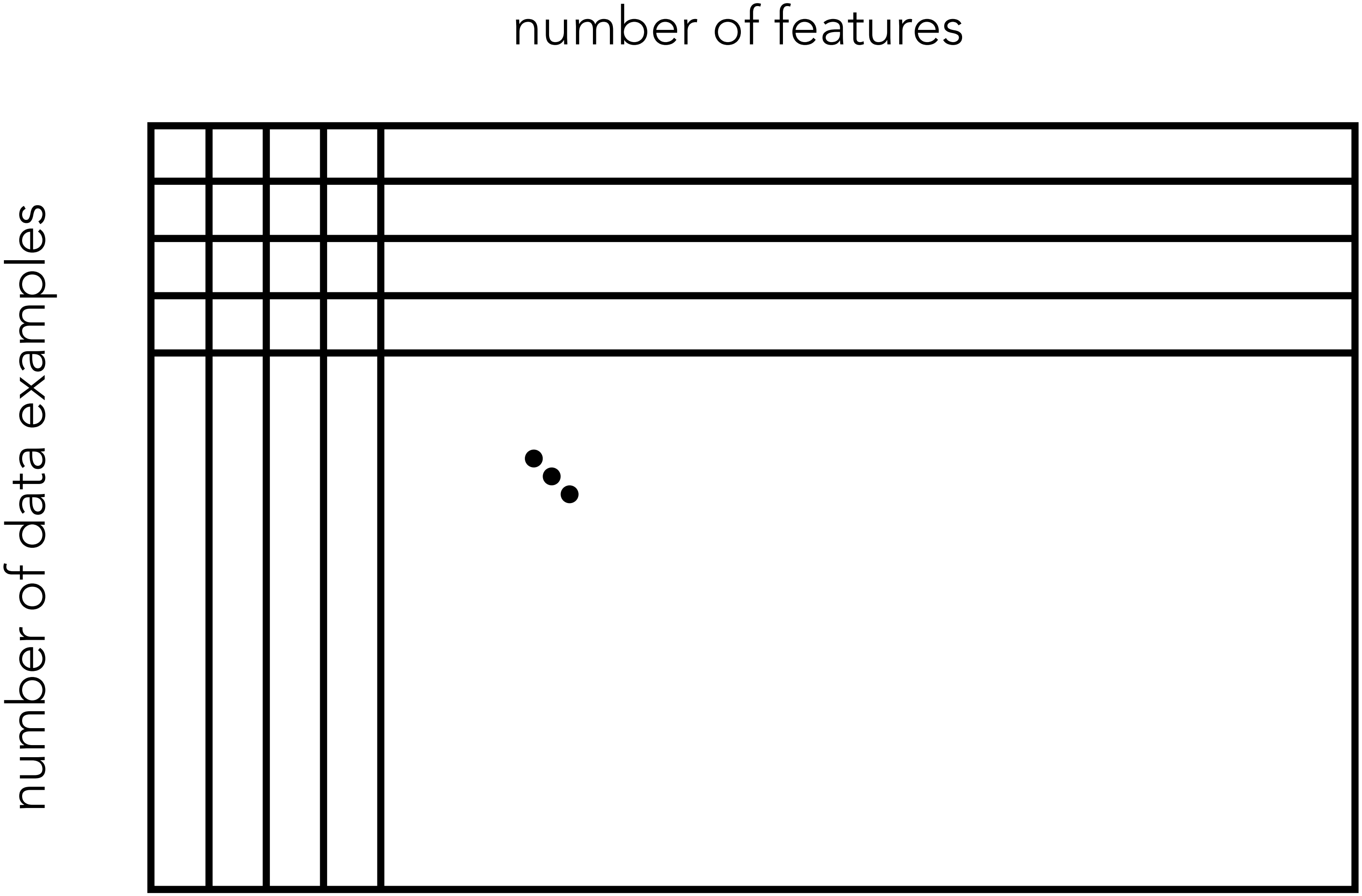
DEEP LEARNING

PART THREE - *DEEP GENERATIVE MODELS*

CS/CNS/EE 155 - MACHINE LEARNING & DATA MINING

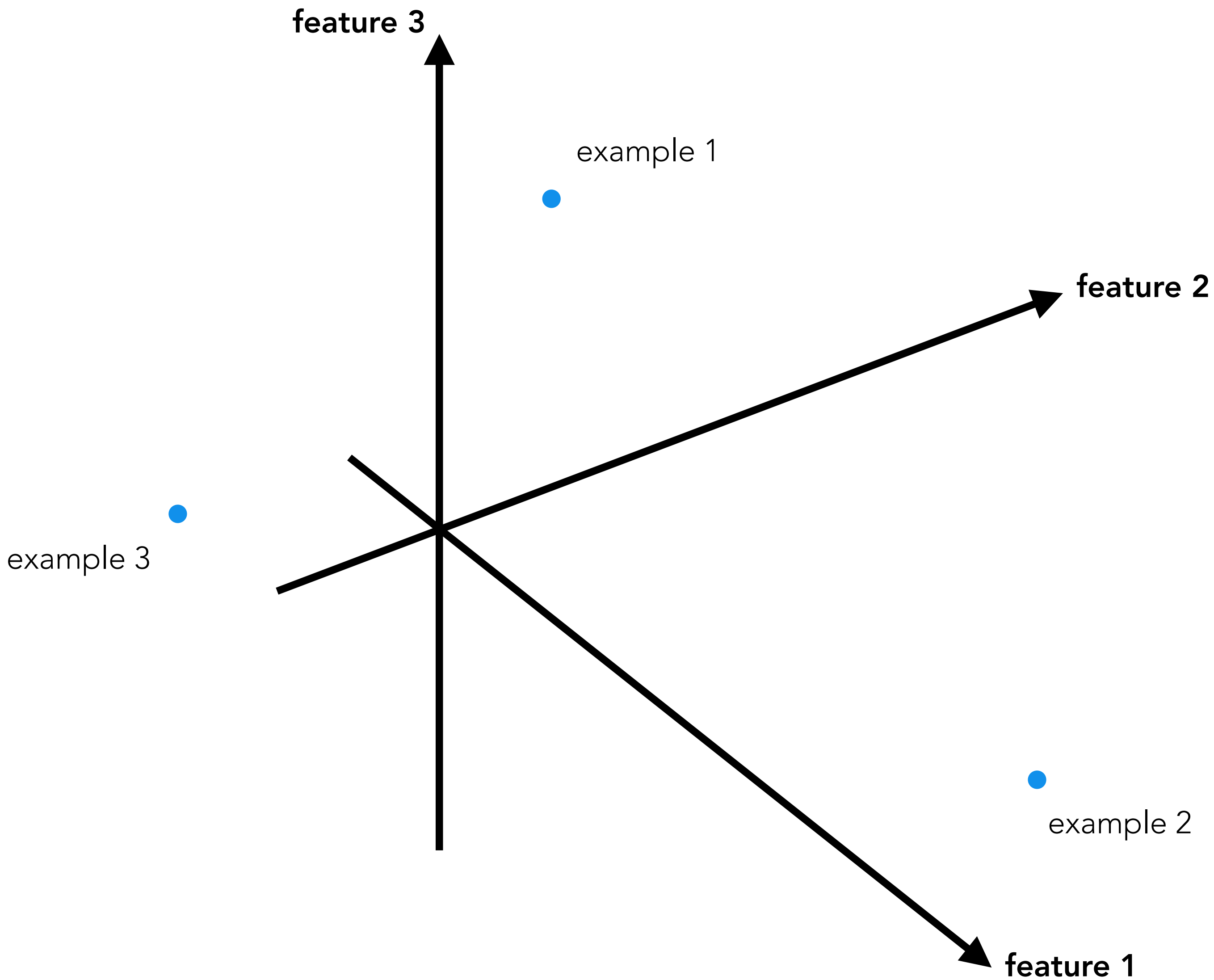
GENERATIVE MODELS



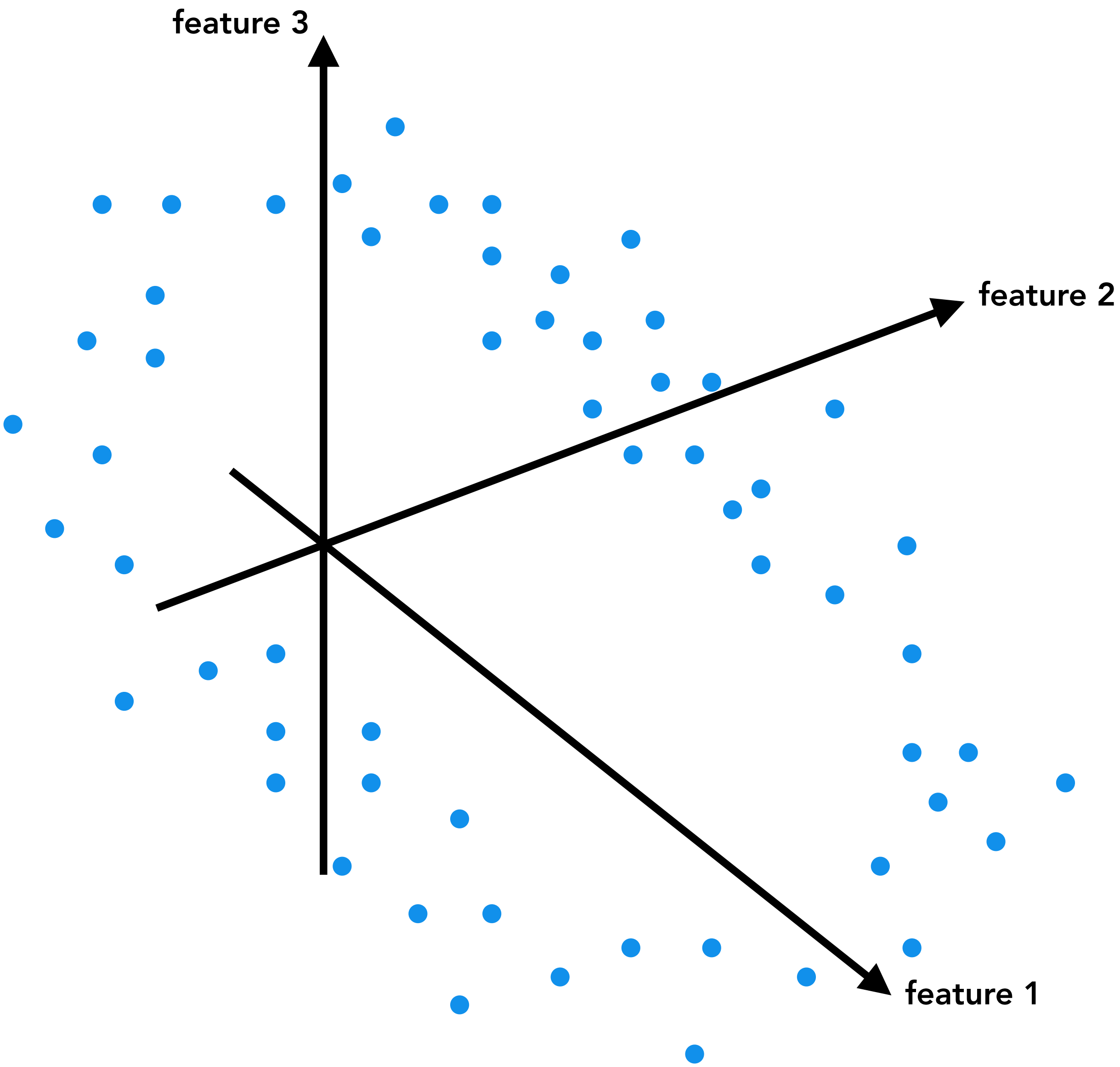
number of features

number of data examples

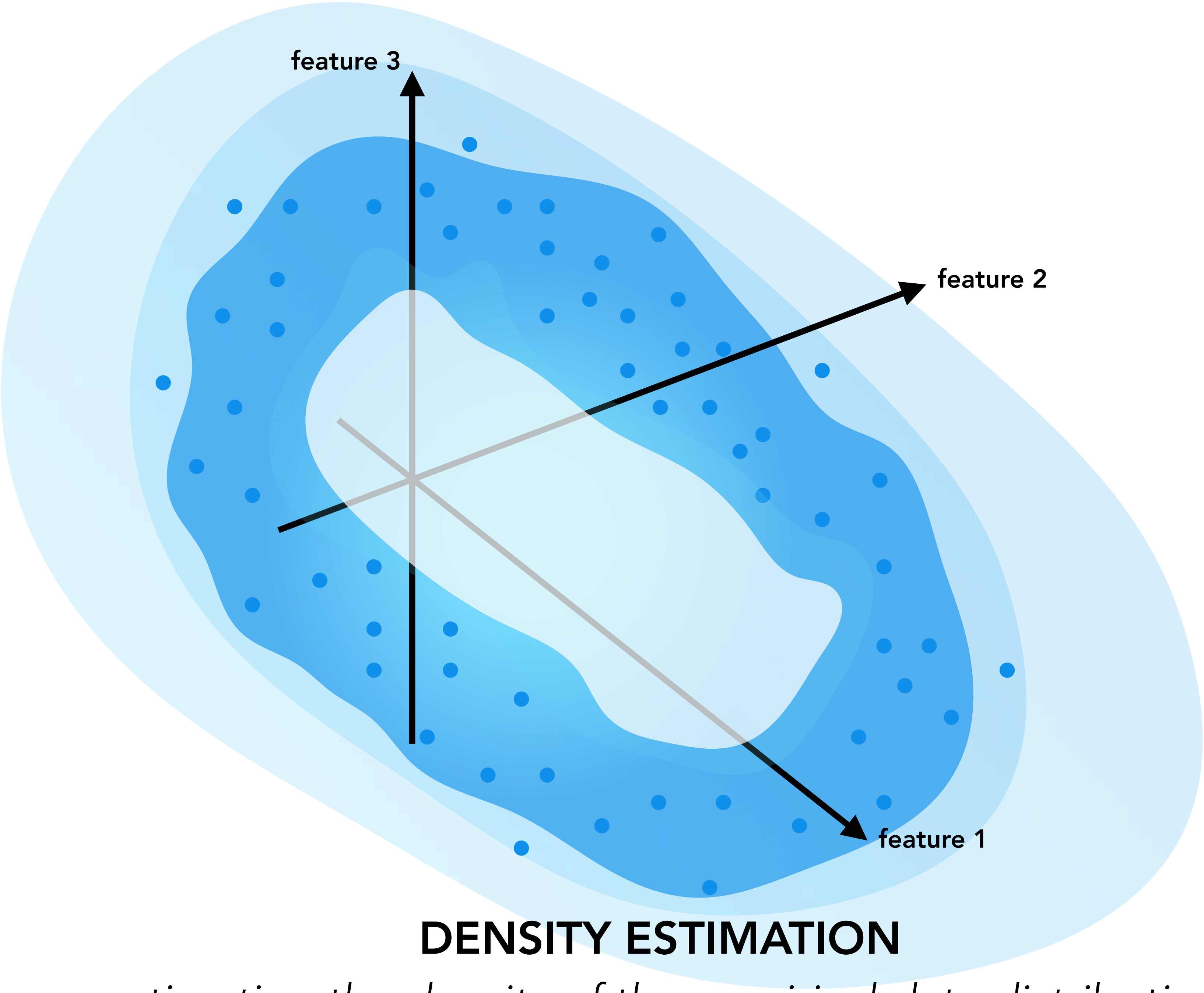
DATA



DATA DISTRIBUTION



EMPIRICAL DATA DISTRIBUTION



DENSITY ESTIMATION

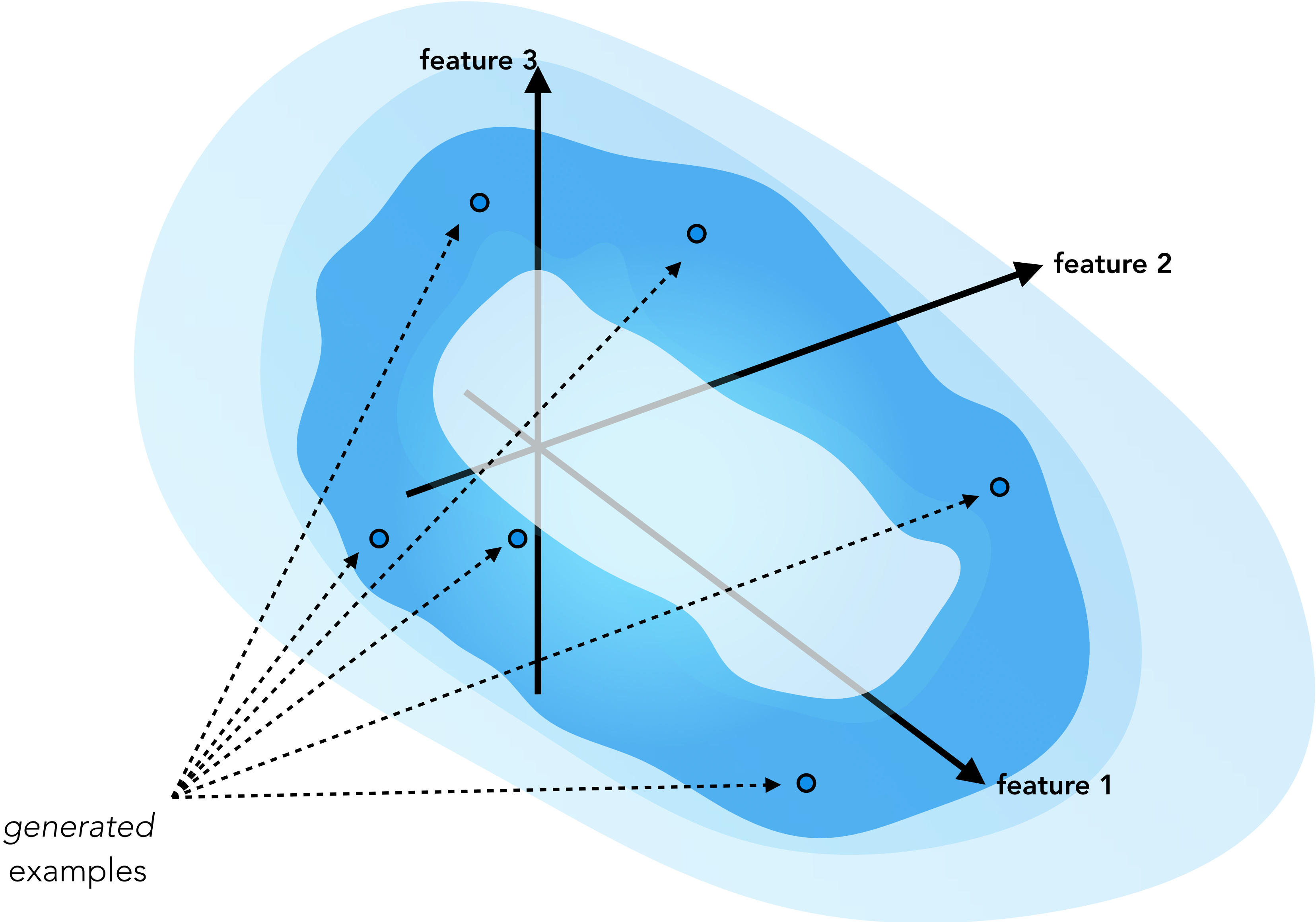
estimating the density of the empirical data distribution

GENERATIVE MODEL

a model of the density of the data distribution

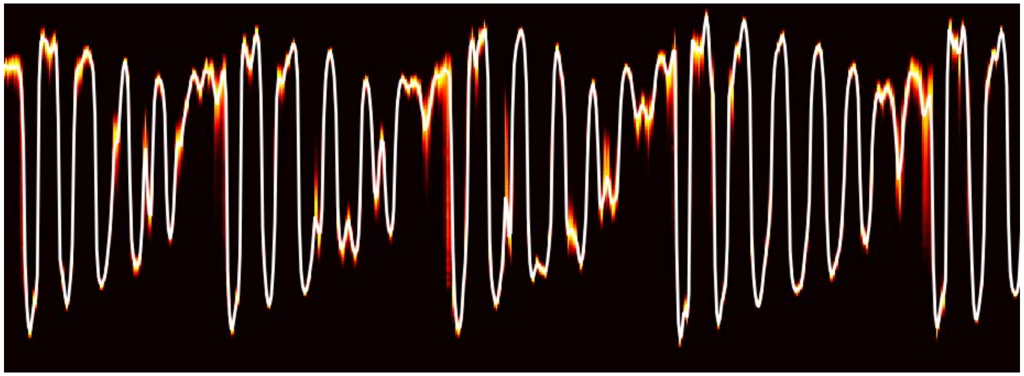
why learn a generative model?

generative models can ***generate new data examples***



Glow, Kingma & Dhariwal, 2018

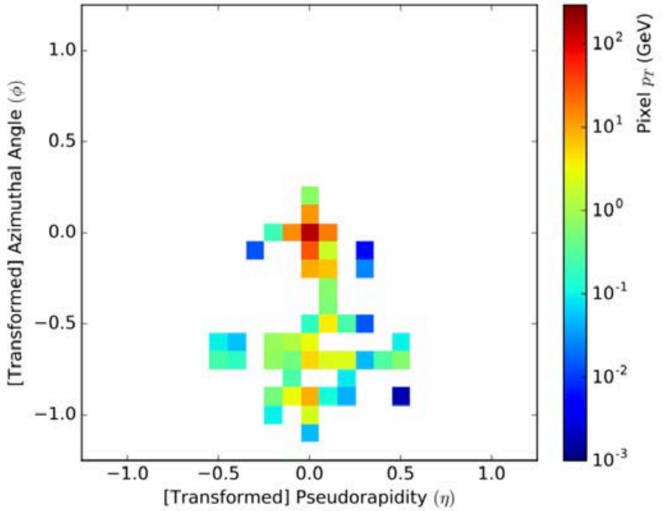
BigGan, Brock *et al.*, 2019



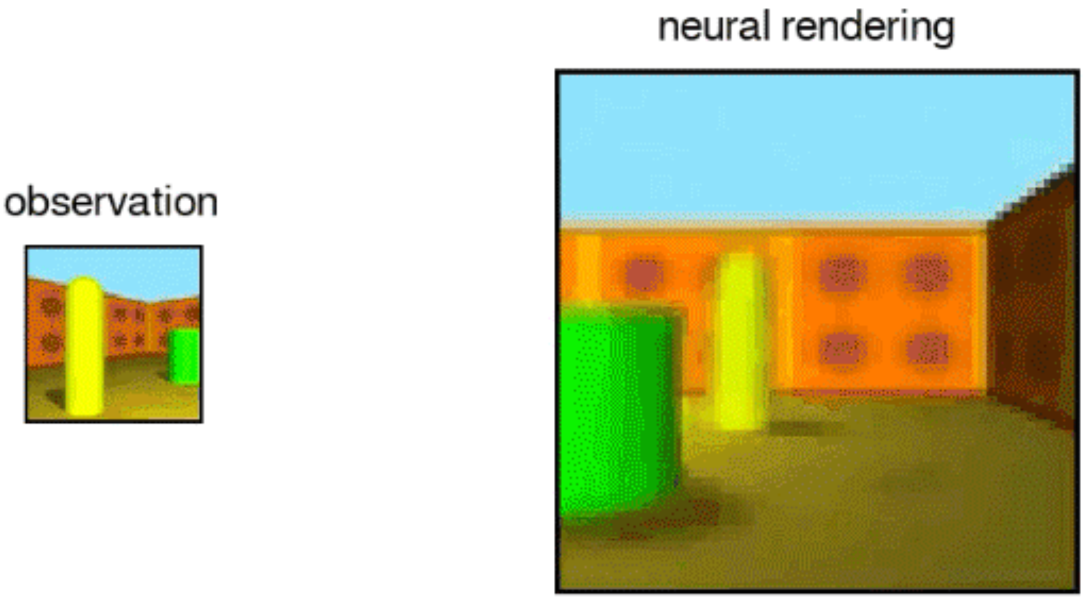
WaveNet, van den Oord *et al.*, 2016

(b) MidiNet model 2

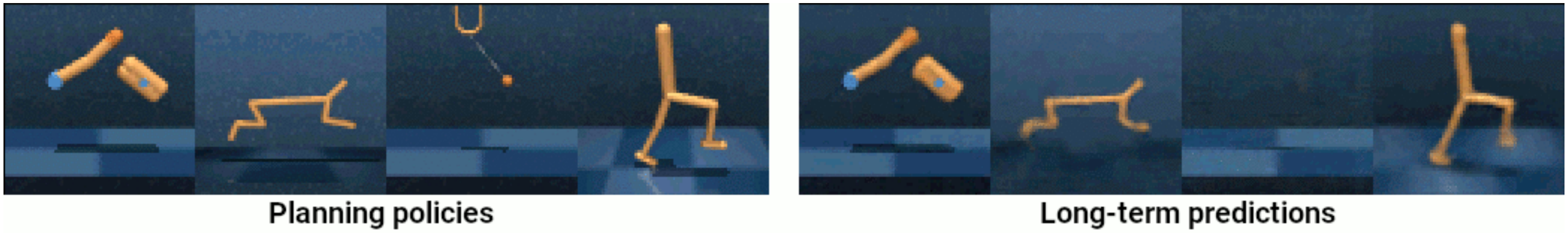
MidiNet, Yang *et al.*, 2017



Learning Particle Physics by Example, *de Oliveira et al.*, 2017

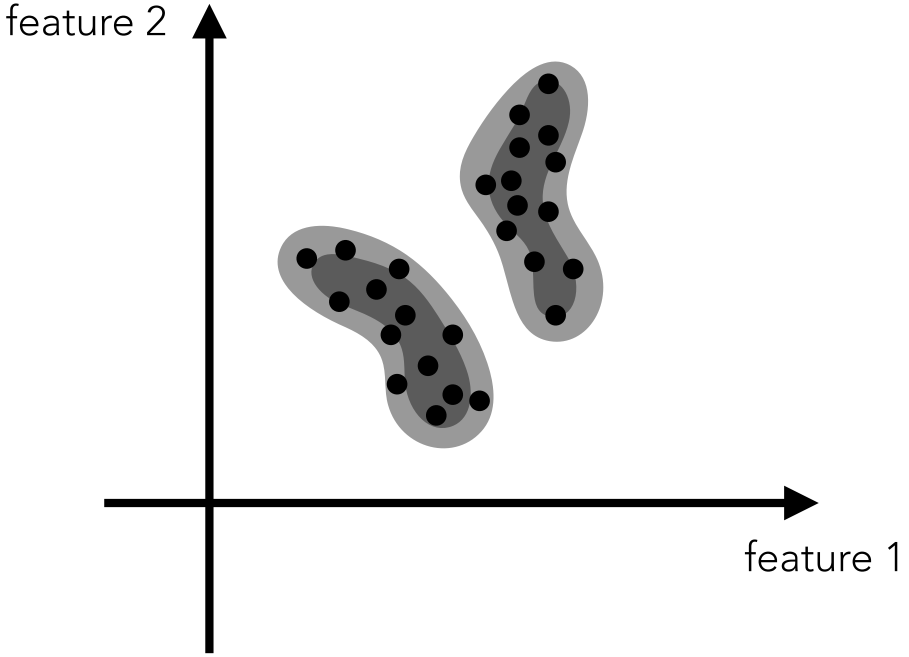


GQN, Eslami *et al.*, 2018

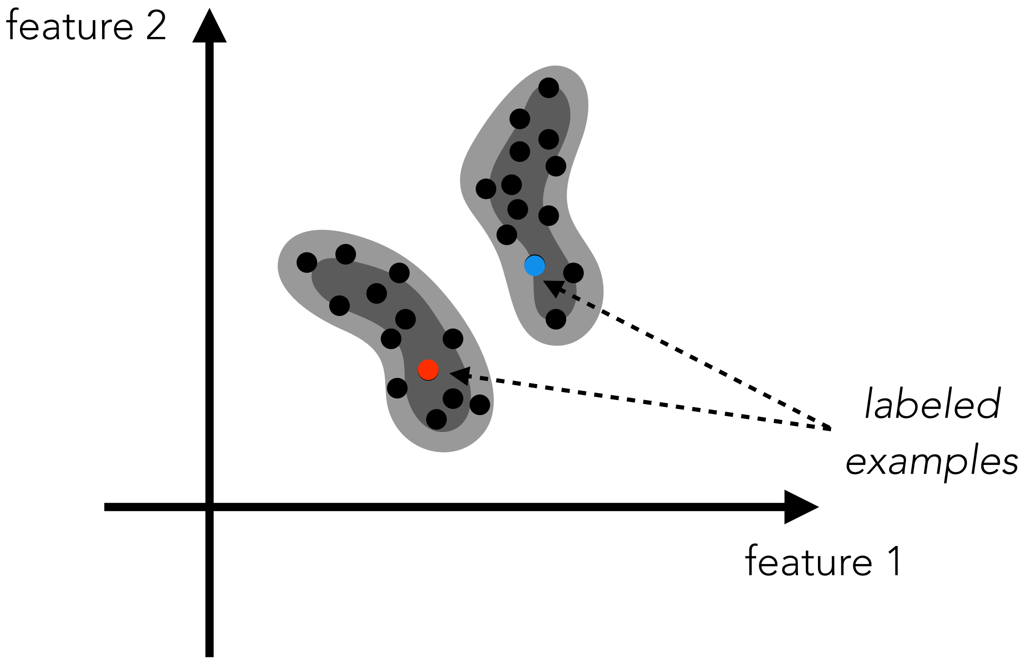


PlaNet, Hafner *et al.*, 2018

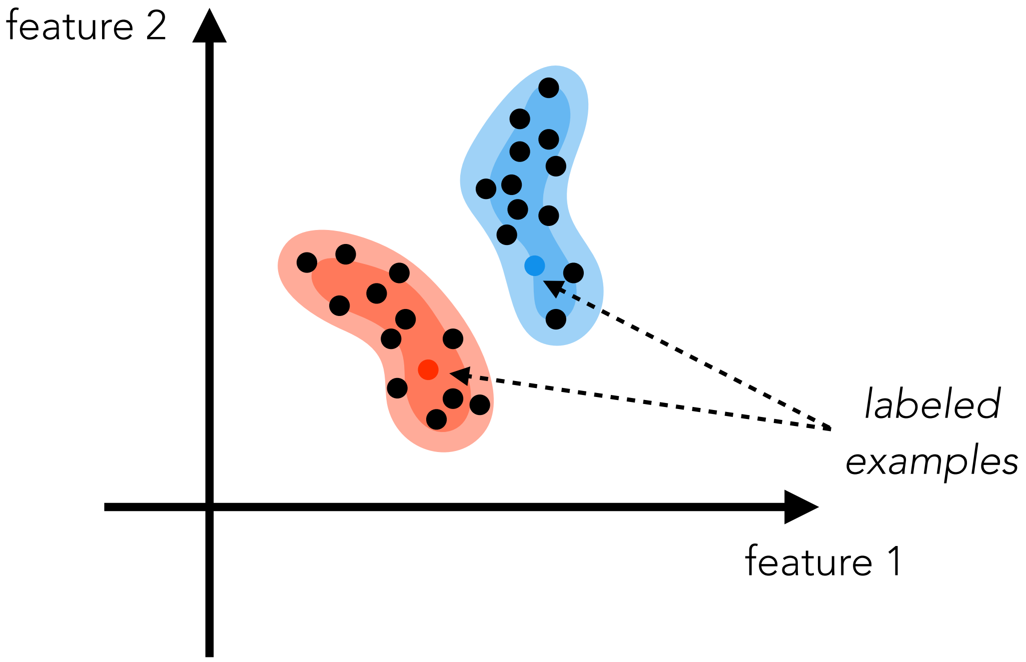
generative models can ***extract structure from data***



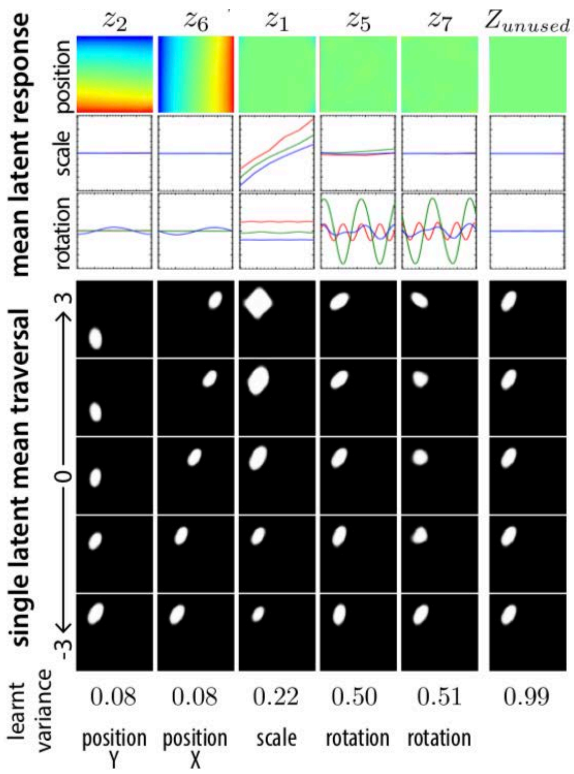
generative models can ***extract structure from data***



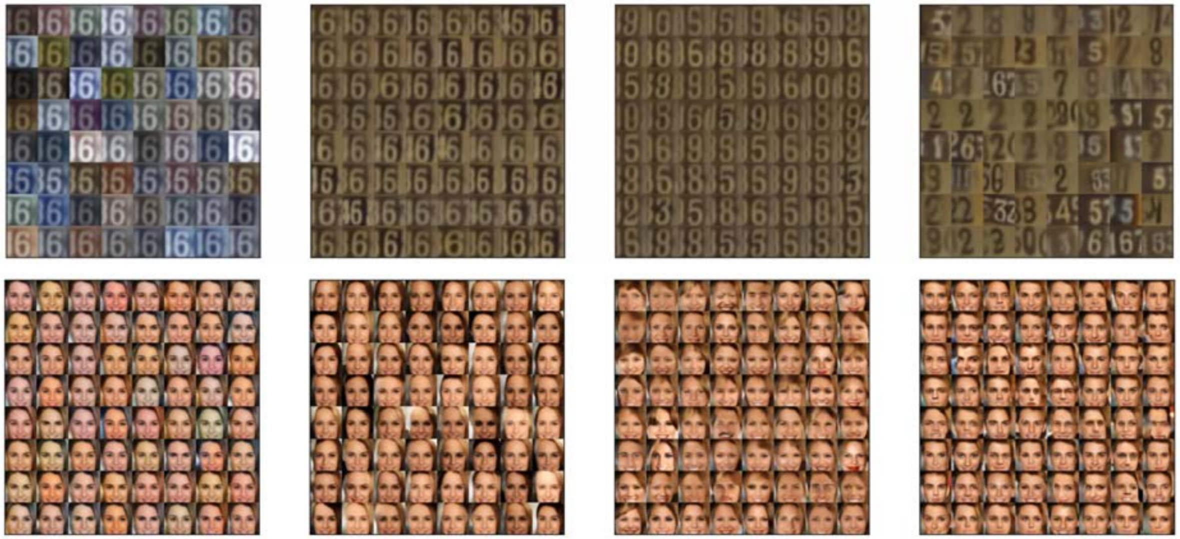
generative models can ***extract structure from data***



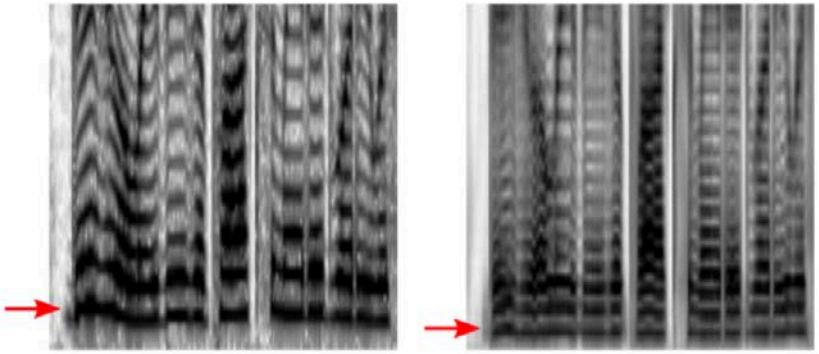
can make it easier to learn and generalize on new tasks



beta-VAE, Higgins *et al.*, 2016



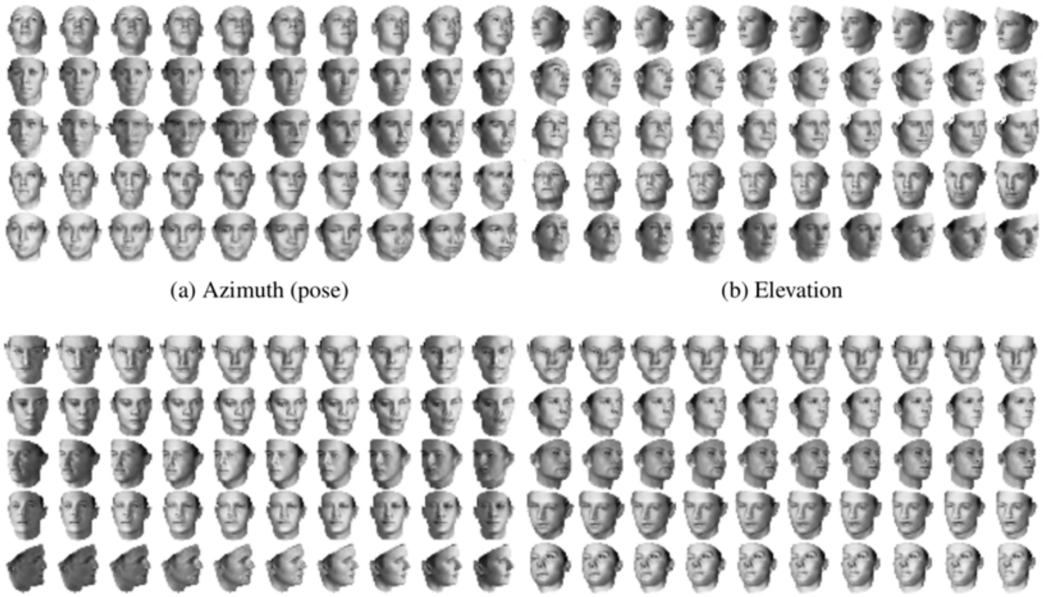
VLAE, Zhao *et al.*, 2017



(a) female speech (original)

(b) female to male

Disentangled Sequential Autoencoder,
Li & Mandt, 2018



(a) Azimuth (pose)

(b) Elevation

(c) Lighting

(d) Wide or Narrow

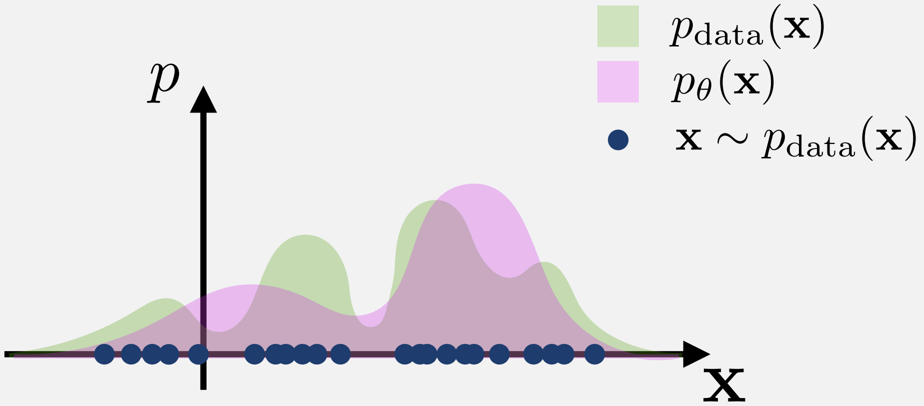
InfoGAN, Chen *et al.*, 2016

modeling the data distribution

data: $p_{\text{data}}(\mathbf{x})$

model: $p_\theta(\mathbf{x})$

parameters: θ

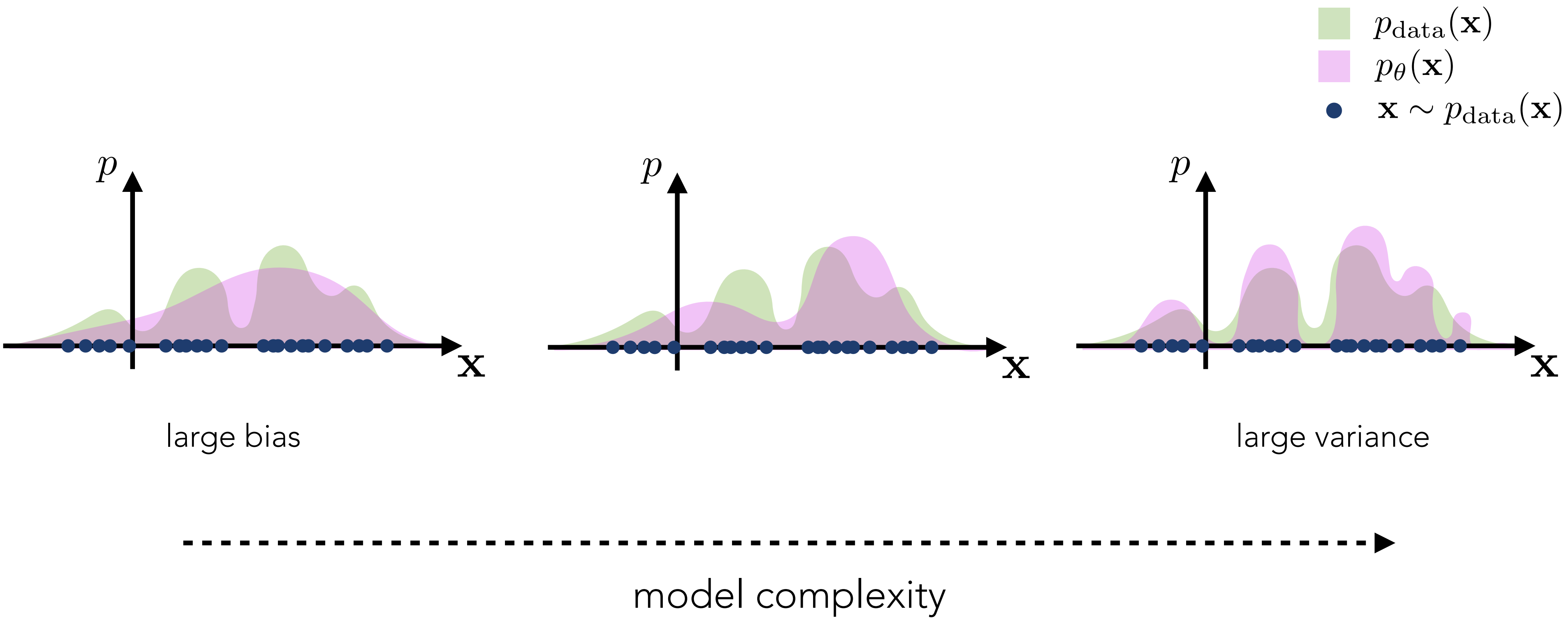


maximum likelihood estimation

find the model that assigns the *maximum likelihood* to the data

$$\begin{aligned}
\theta^* &= \arg\min_\theta \; D_{KL}(p_{\text{data}}(\mathbf{x}) || p_\theta(\mathbf{x})) \\
&= \arg\min_\theta \; \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[\log p_{\text{data}}(\mathbf{x}) - \log p_\theta(\mathbf{x})\right] \\
&= \arg\max_\theta \; \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[\log p_\theta(\mathbf{x})\right] \approx \frac{1}{N} \sum_{i=1}^{N} \log p_\theta(\mathbf{x}^{(i)})
\end{aligned}$$

bias-variance trade-off



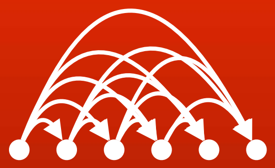
large bias

large variance

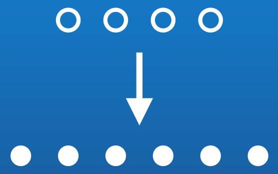
model complexity

deep generative model

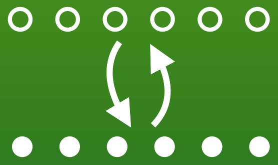
a generative model that uses deep neural networks
to model the data distribution



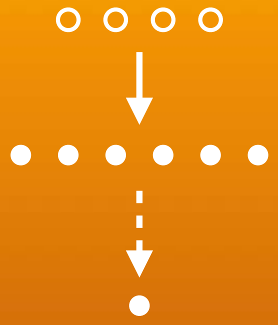
autoregressive models



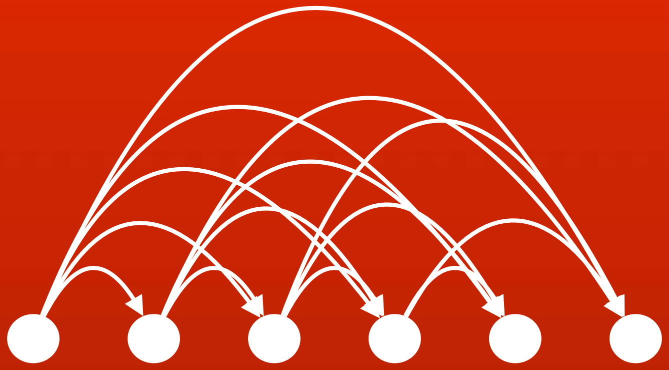
explicit latent variable models



invertible explicit latent variable models



implicit latent variable models

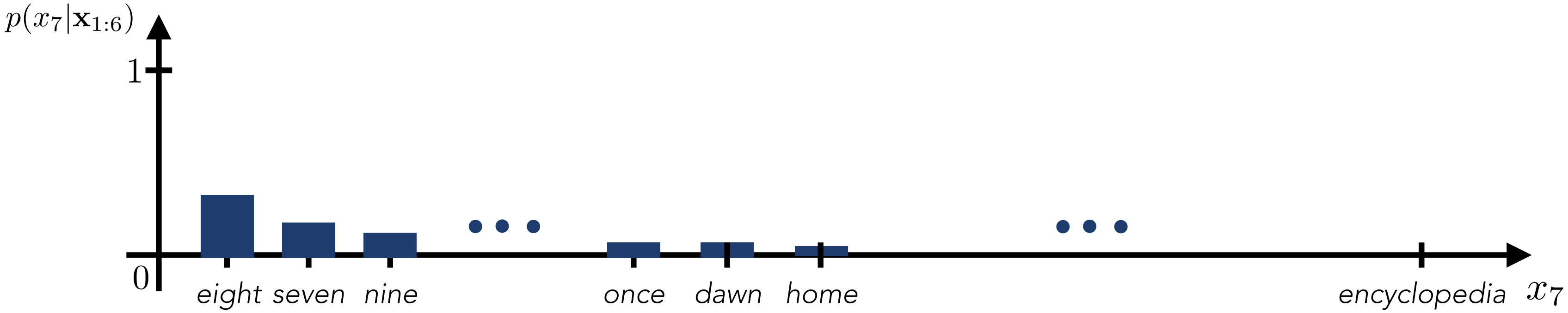


autoregressive models

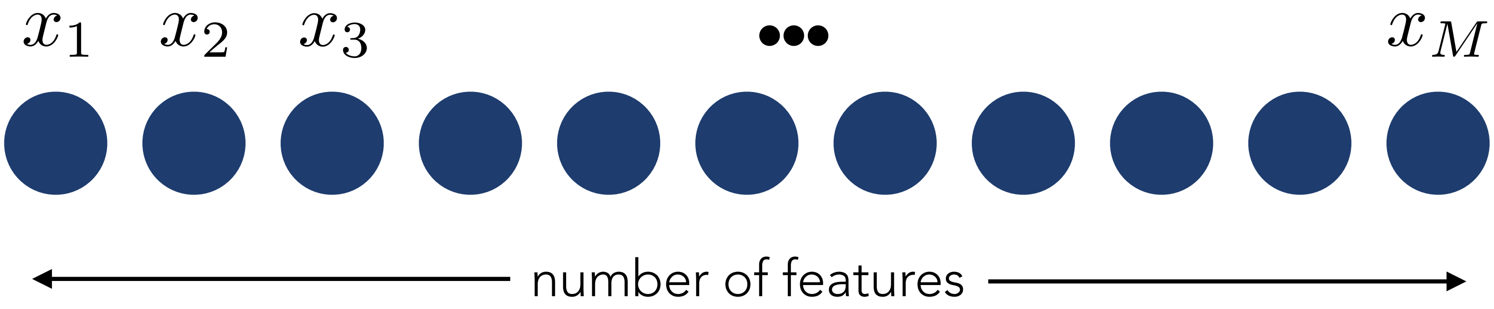
conditional probability distributions

This	*morning*	*I*	*woke*	*up*	*at*	
x_1	x_2	x_3	x_4	x_5	x_6	x_7

What is $p(x_7|\mathbf{x}_{1:6})$?



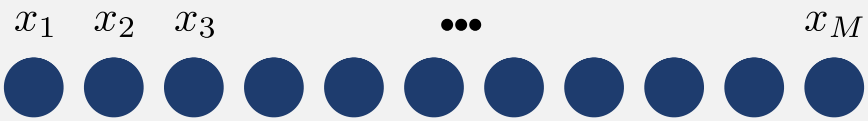
a data example



$$p(\mathbf{x}) = p(x_1, x_2, \ldots, x_M)$$

chain rule of probability

split the joint distribution into a product of conditional distributions



$$p(\mathbf{x}) = p(x_1, x_2, \ldots, x_M)$$

$$p(a|b) = \frac{p(a,b)}{p(b)} \longrightarrow p(a,b) = p(a|b)p(b)$$

definition of conditional probability

recursively apply to $p(x_1, x_2, \ldots, x_M)$:

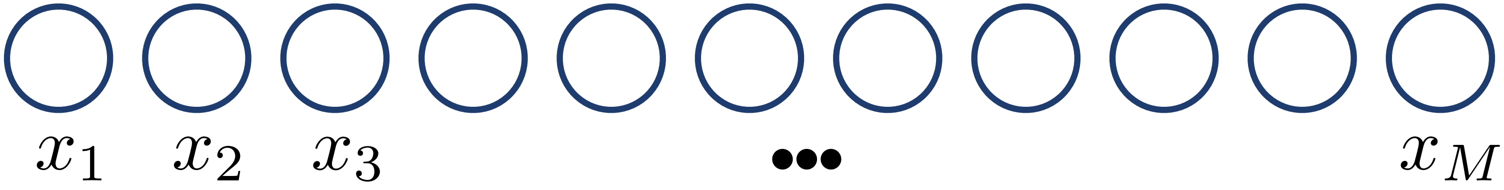
$$\begin{aligned} p(x_1, x_2, \ldots, x_M) &= p(x_1)p(x_2, \ldots, x_M|x_1) \\ &\vdots \\ &= p(x_1)p(x_2|x_1)\ldots p(x_M|x_1, \ldots, x_{M-1}) \end{aligned}$$

$$\boxed{p(x_1, \ldots, x_M) = \prod_{j=1}^{M} p(x_j|x_1, \ldots, x_{j-1})}$$

note: conditioning order is arbitrary

model the conditional distributions of the data

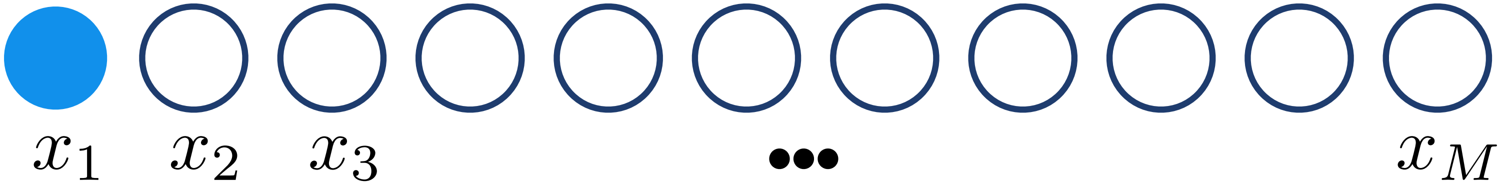
learn to ***auto-regress*** each value



model the conditional distributions of the data

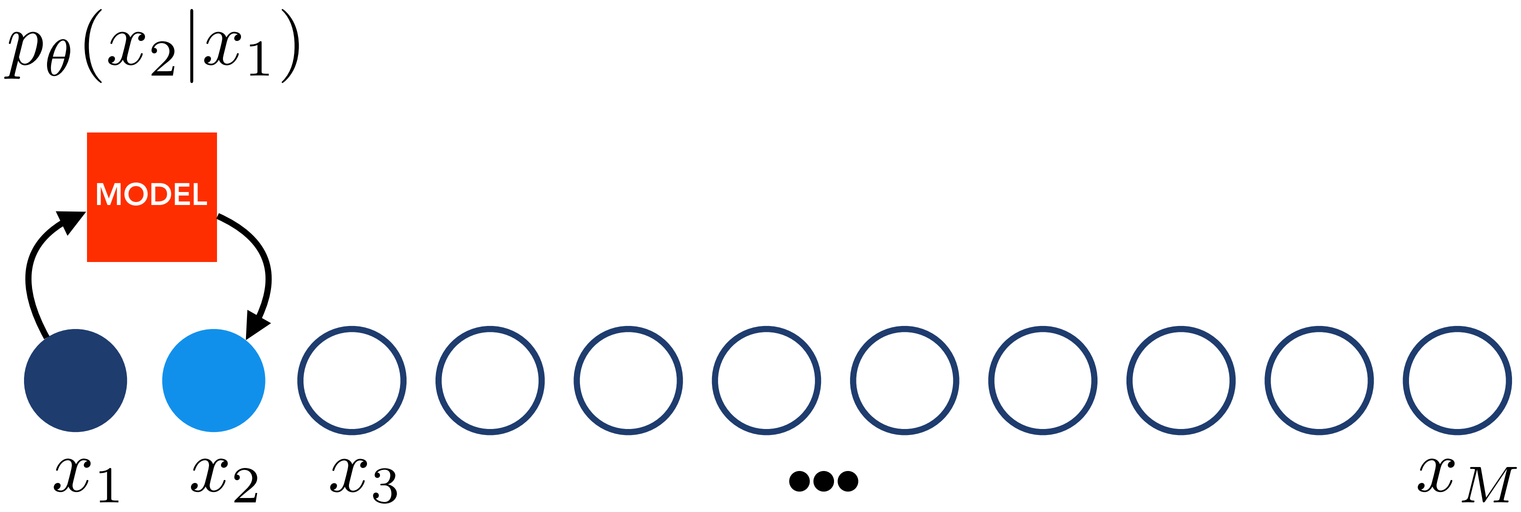
learn to ***auto-regress*** each value

$p_\theta(x_1)$



model the conditional distributions of the data

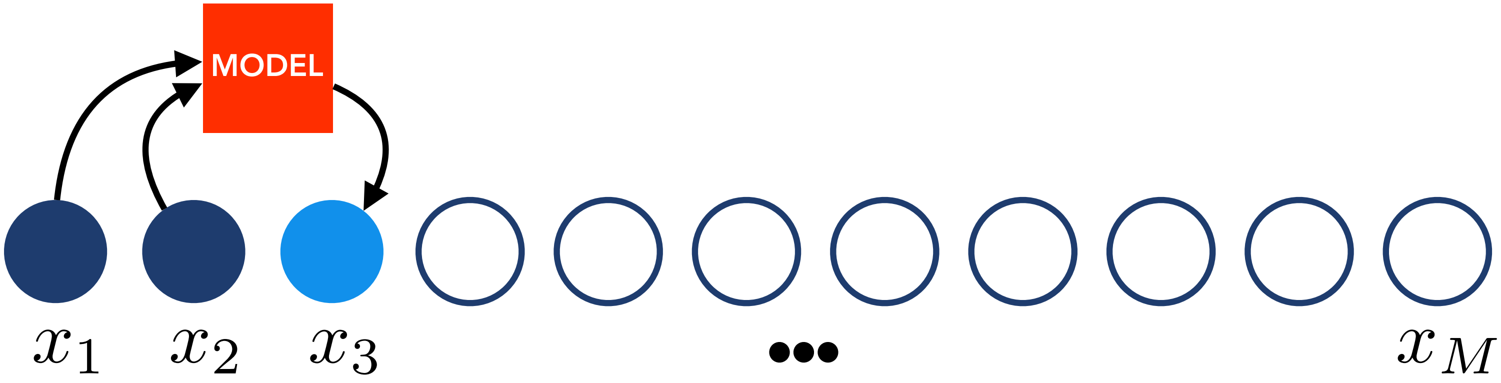
learn to ***auto-regress*** each value



model the conditional distributions of the data

learn to ***auto-regress*** each value

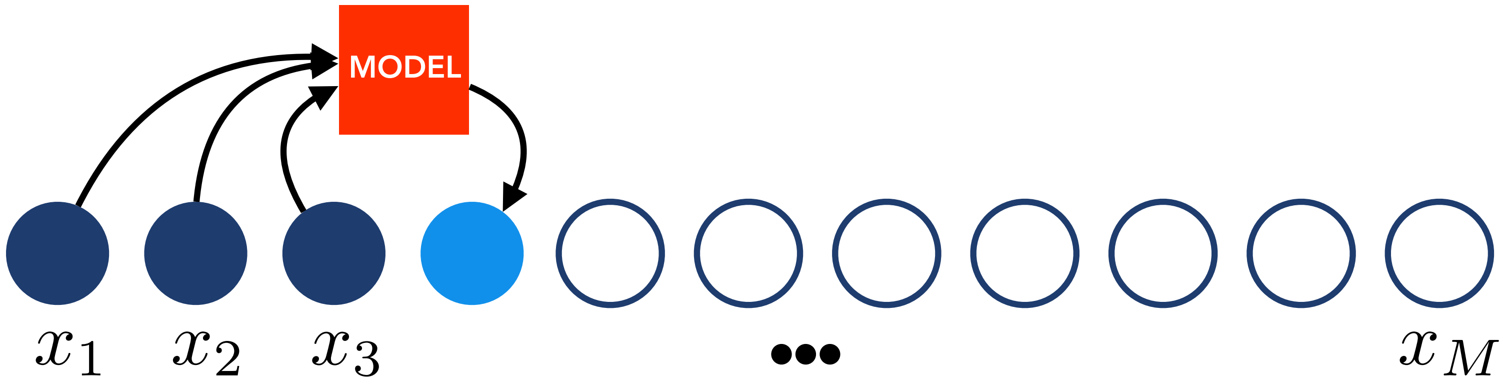
$p_\theta(x_3|x_1, x_2)$



model the conditional distributions of the data

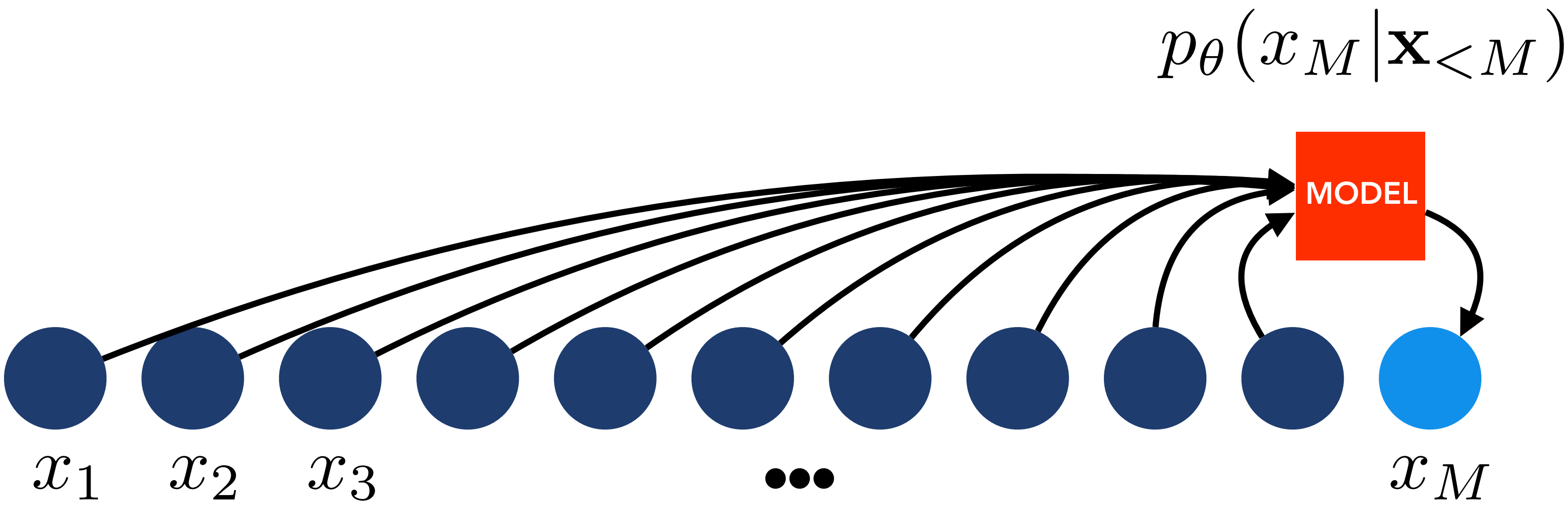
learn to ***auto-regress*** each value

$$p_\theta(x_4|x_1, x_2, x_3)$$



model the conditional distributions of the data

learn to ***auto-regress*** each value



maximum likelihood estimation

maximize the log-likelihood (under the model) of the true data examples

$$\theta^* = \arg\max_\theta \ \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] \approx \frac{1}{N}\sum_{i=1}^{N} \log p_\theta(\mathbf{x}^{(i)})$$

for auto-regressive models:

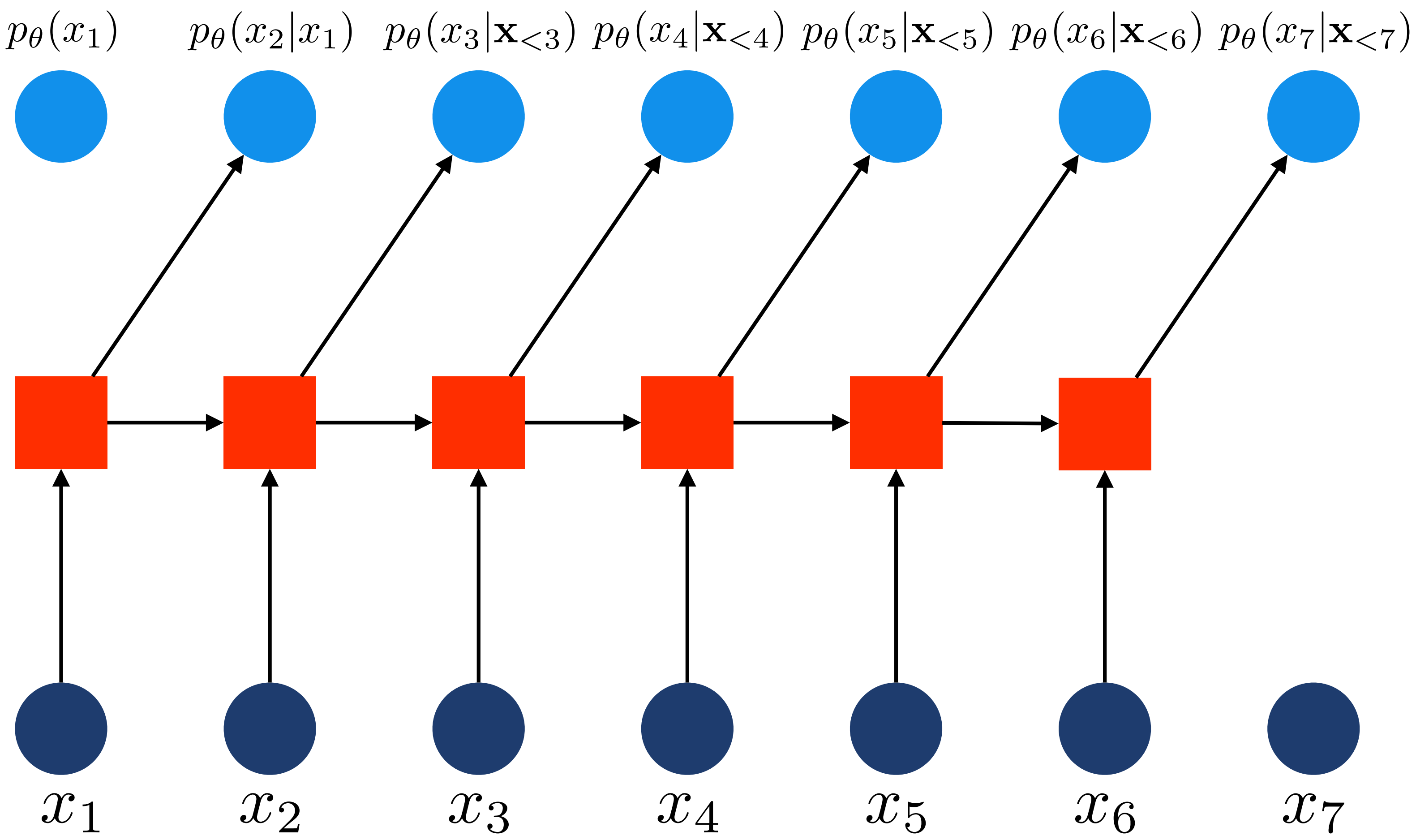
$$\log p_\theta(\mathbf{x}) = \log\left(\prod_{j=1}^{M} p_\theta(x_j|\mathbf{x}_{<j})\right)$$

$$= \sum_{j=1}^{M} \log p_\theta(x_j|\mathbf{x}_{<j})$$

$$\theta^* = \arg\max_\theta \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} \log p_\theta(x_j^{(i)}|\mathbf{x}_{<j}^{(i)})$$

models

can parameterize conditional distributions using a **recurrent neural network**

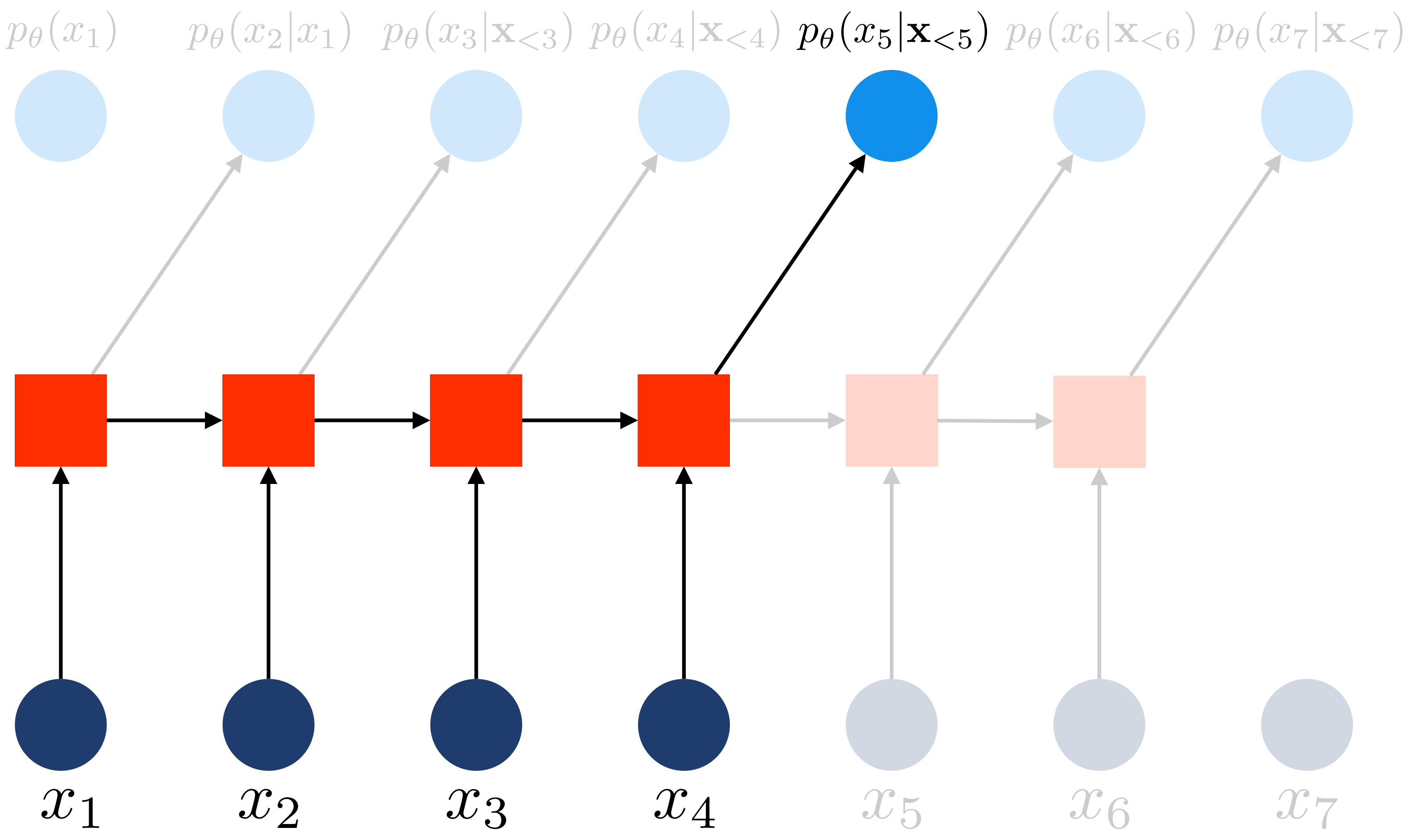


$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{<3})$
$p_\theta(x_4|\mathbf{x}_{<4})$
$p_\theta(x_5|\mathbf{x}_{<5})$
$p_\theta(x_6|\mathbf{x}_{<6})$
$p_\theta(x_7|\mathbf{x}_{<7})$
x_1
x_2
x_3
x_4
x_5
x_6
x_7

see **Deep Learning** (Chapter 10), *Goodfellow et al.*, 2016

models

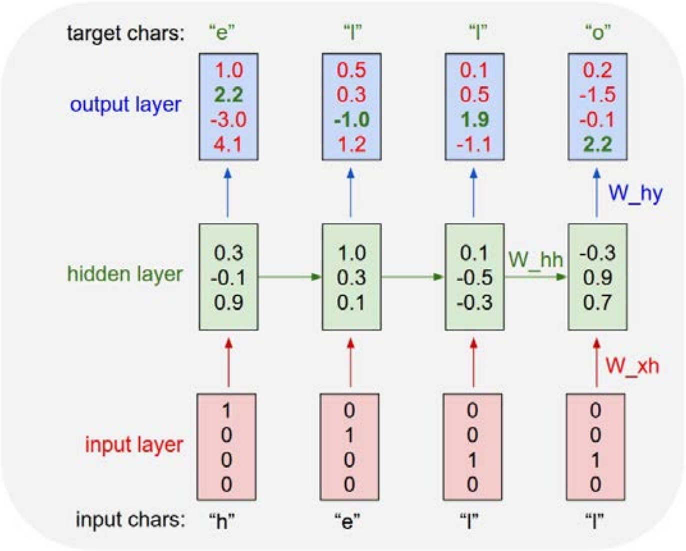
can parameterize conditional distributions using a **recurrent neural network**


$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{<3})$
$p_\theta(x_4|\mathbf{x}_{<4})$
$p_\theta(x_5|\mathbf{x}_{<5})$
$p_\theta(x_6|\mathbf{x}_{<6})$
$p_\theta(x_7|\mathbf{x}_{<7})$
x_1
x_2
x_3
x_4
x_5
x_6
x_7

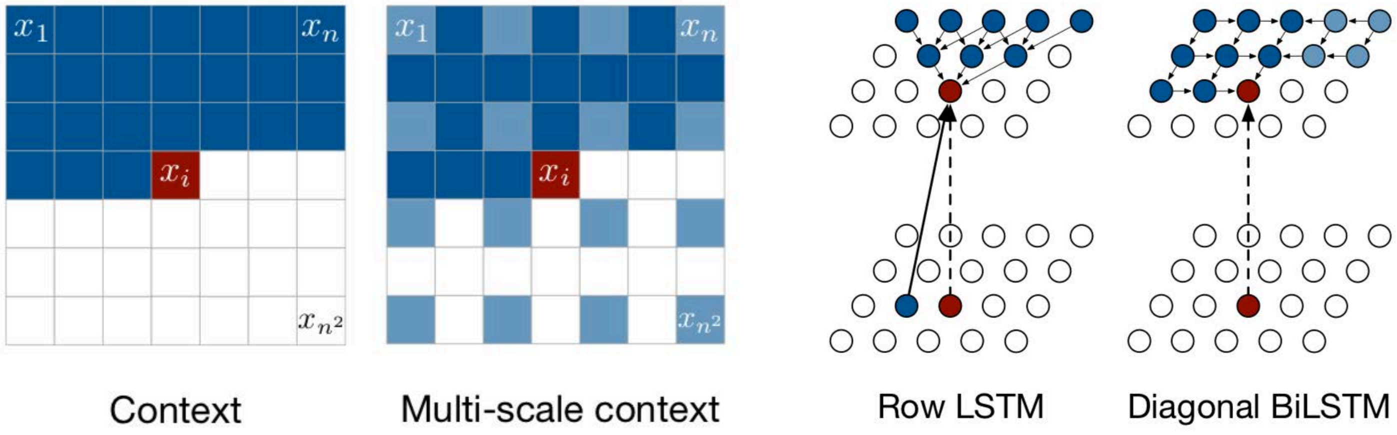
see **Deep Learning** (Chapter 10), *Goodfellow et al.*, 2016

models

can parameterize conditional distributions using a **recurrent neural network**



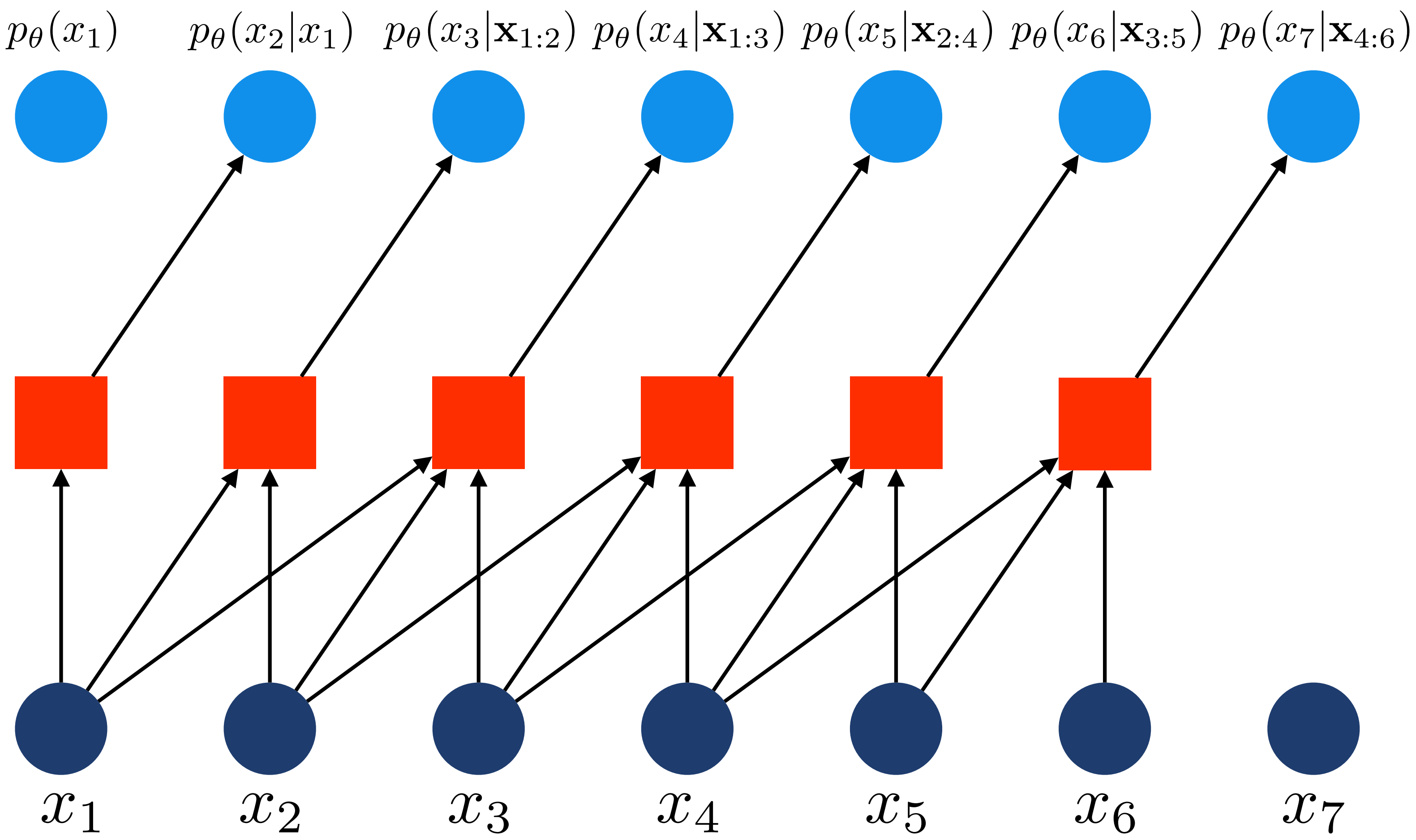
The Unreasonable Effectiveness of Recurrent Neural Networks, *Karpathy*, 2015



Pixel Recurrent Neural Networks, *van den Oord et al.*, 2016

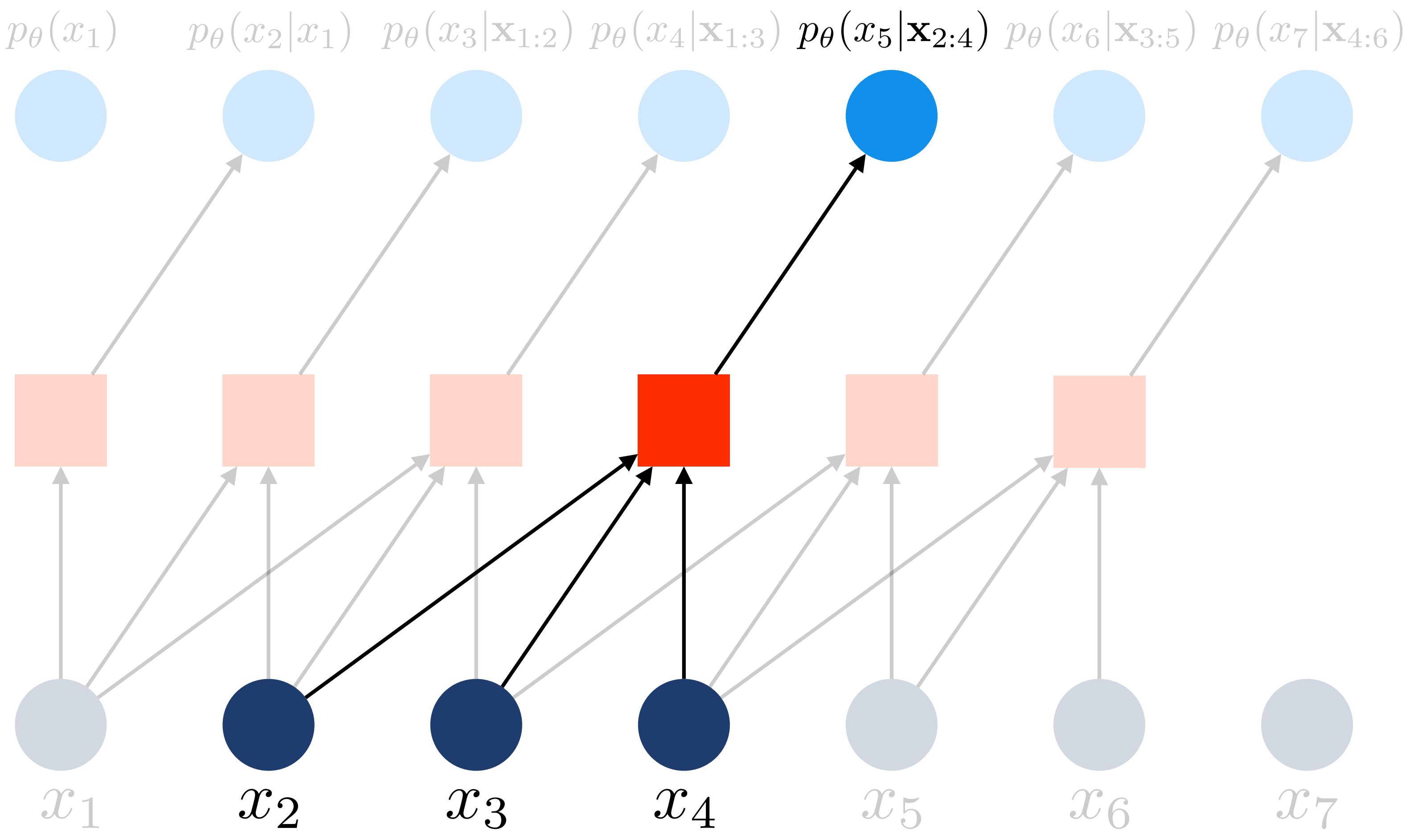
models

can condition on a local window using **convolutional neural networks**


$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{1:2})$
$p_\theta(x_4|\mathbf{x}_{1:3})$
$p_\theta(x_5|\mathbf{x}_{2:4})$
$p_\theta(x_6|\mathbf{x}_{3:5})$
$p_\theta(x_7|\mathbf{x}_{4:6})$
x_1
x_2
x_3
x_4
x_5
x_6
x_7

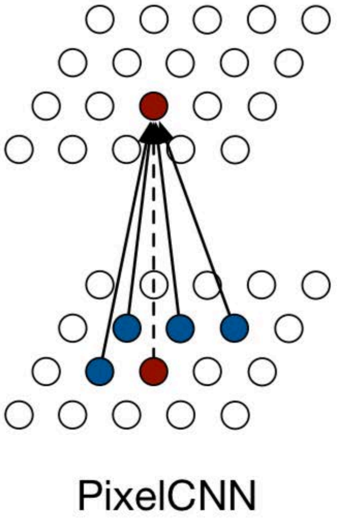
models

can condition on a local window using **convolutional neural networks**

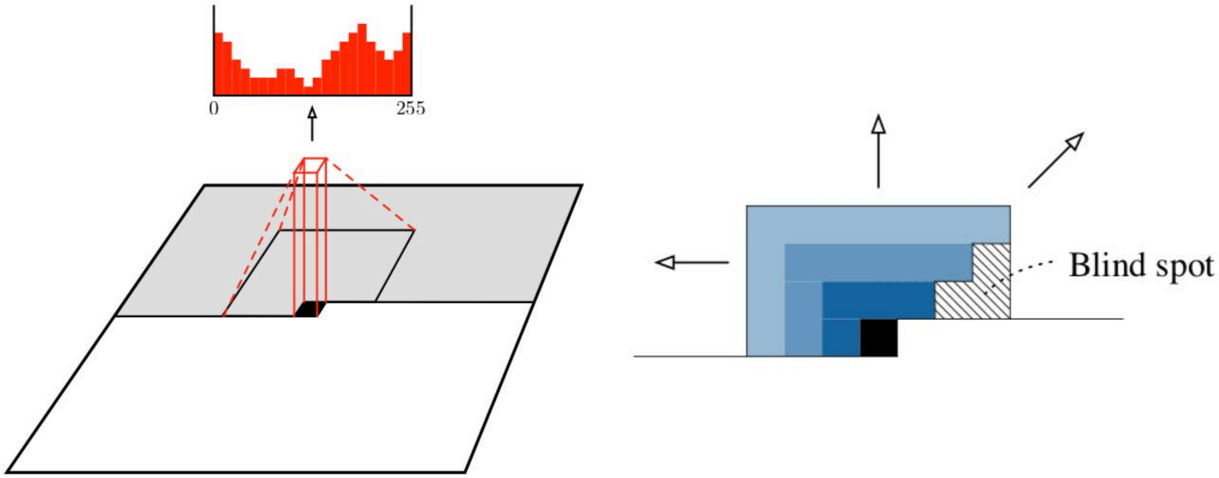

$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{1:2})$
$p_\theta(x_4|\mathbf{x}_{1:3})$
$p_\theta(x_5|\mathbf{x}_{2:4})$
$p_\theta(x_6|\mathbf{x}_{3:5})$
$p_\theta(x_7|\mathbf{x}_{4:6})$
x_1
x_2
x_3
x_4
x_5
x_6
x_7

models

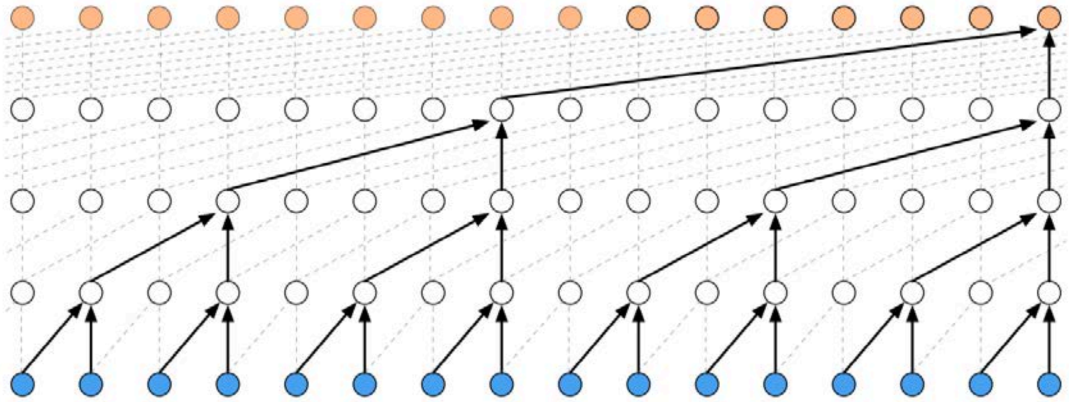
can condition on a local window using **convolutional neural networks**



Pixel Recurrent Neural Networks, *van den Oord et al.*, 2016



Conditional Image Generation with PixelCNN Decoders, *van den Oord et al.*, 2016



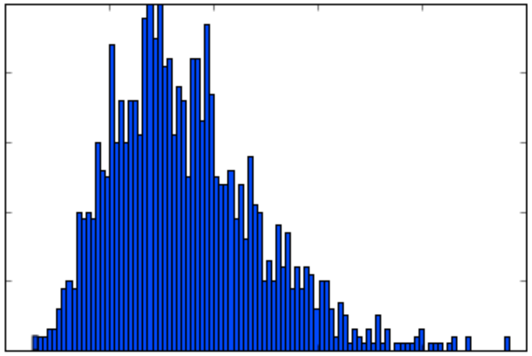
WaveNet: A Generative Model for Raw Audio, *van den Oord et al.*, 2016

output distributions

need to choose a form for the conditional **output distribution**,
i.e. how do we express $p(x_j|x_1,...,x_{j-1})$?

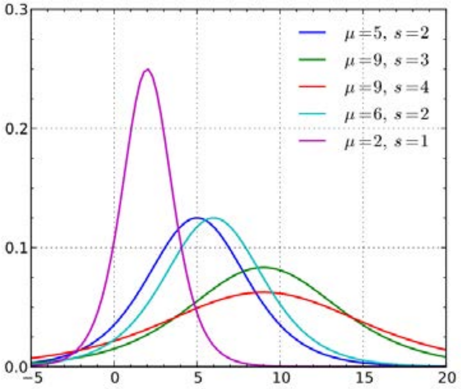
*model the data as **discrete** variables*

→ *categorical output*



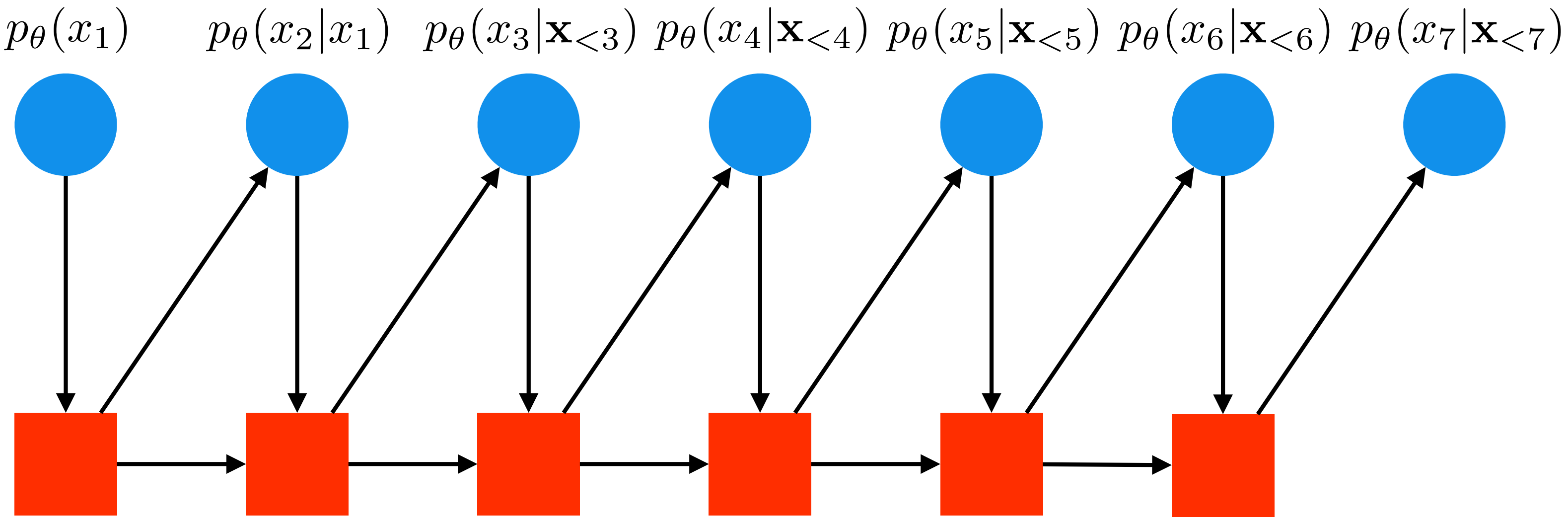
*model the data as **continuous** variables*

→ *Gaussian, logistic, etc. output*



sampling

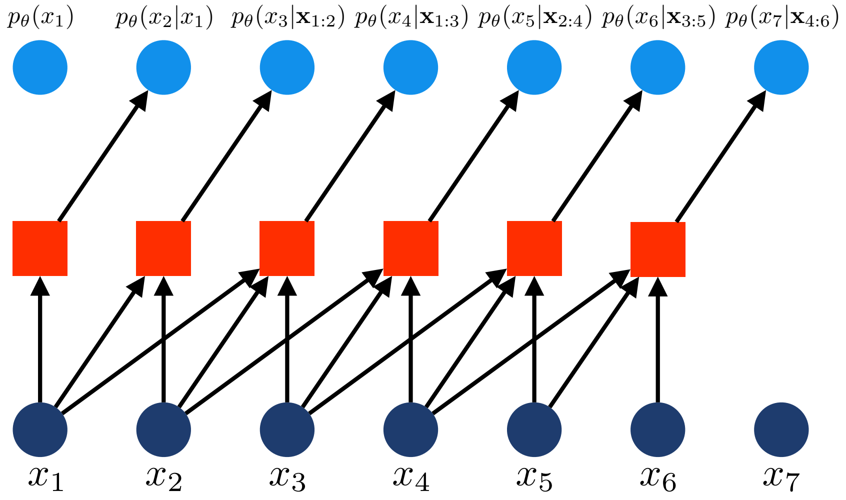
sample from the model by drawing from the output distribution


$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{<3})$
$p_\theta(x_4|\mathbf{x}_{<4})$
$p_\theta(x_5|\mathbf{x}_{<5})$
$p_\theta(x_6|\mathbf{x}_{<6})$
$p_\theta(x_7|\mathbf{x}_{<7})$

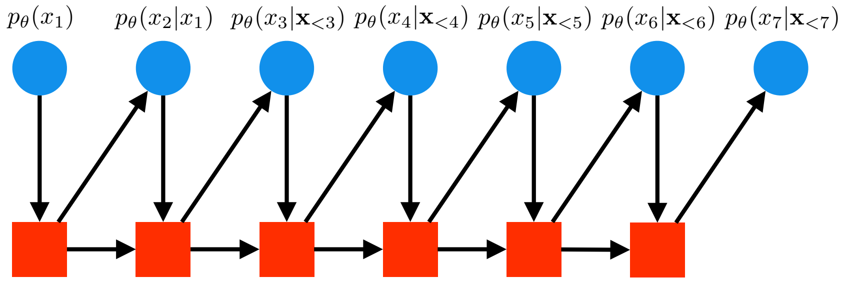
question

what issues might arise with sampling from the model?

training


$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{1:2})$
$p_\theta(x_4|\mathbf{x}_{1:3})$
$p_\theta(x_5|\mathbf{x}_{2:4})$
$p_\theta(x_6|\mathbf{x}_{3:5})$
$p_\theta(x_7|\mathbf{x}_{4:6})$
x_1
x_2
x_3
x_4
x_5
x_6
x_7

sampling


$p_\theta(x_1)$
$p_\theta(x_2|x_1)$
$p_\theta(x_3|\mathbf{x}_{<3})$
$p_\theta(x_4|\mathbf{x}_{<4})$
$p_\theta(x_5|\mathbf{x}_{<5})$
$p_\theta(x_6|\mathbf{x}_{<6})$
$p_\theta(x_7|\mathbf{x}_{<7})$

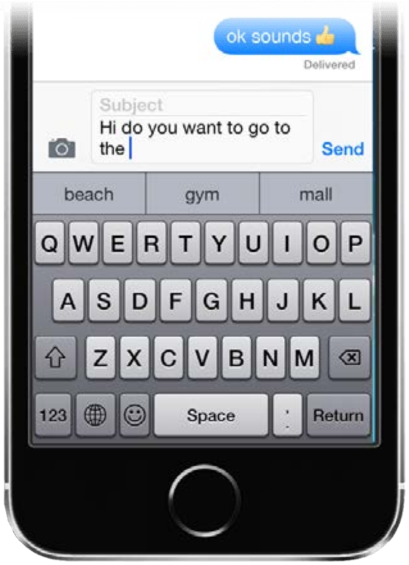
errors in the model distribution can accumulate, leading to poor samples

see *teacher forcing*

Deep Learning (Chapter 10), *Goodfellow et al.*, 2016

example applications

text



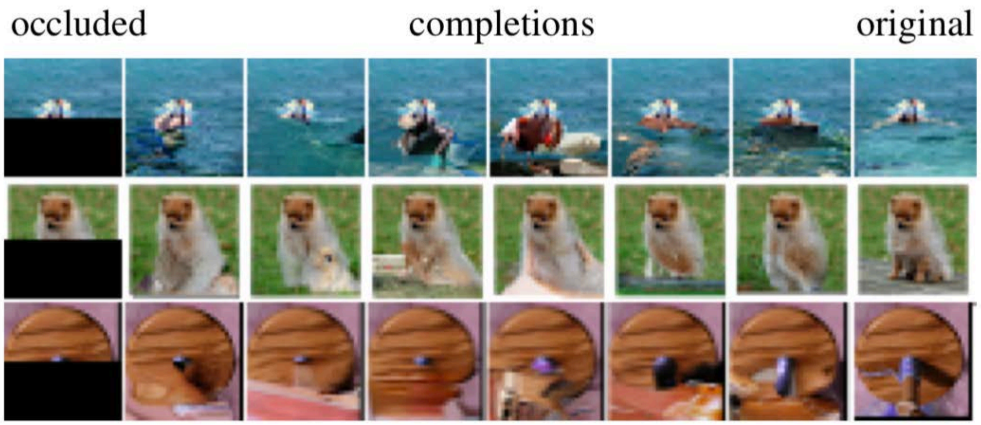
ok sounds 👍

Delivered

Subject

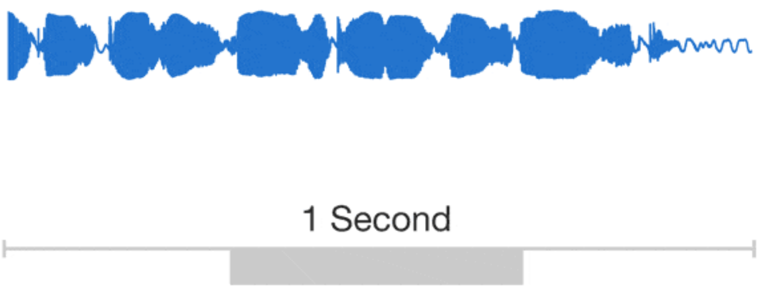
Hi do you want to go to the

images

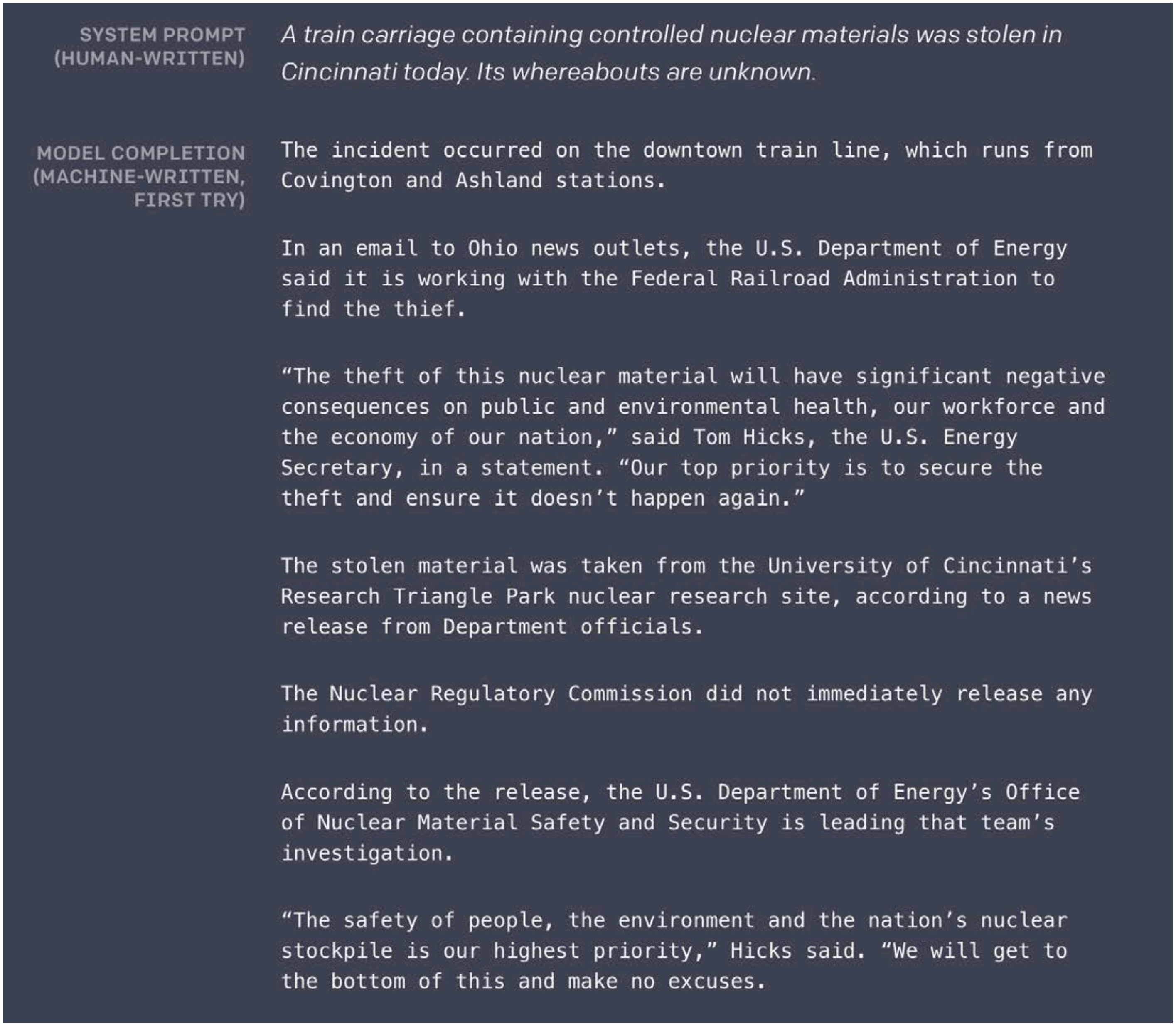


Pixel Recurrent Neural Networks, *van den Oord et al.*, 2016

speech



WaveNet: A Generative Model for Raw Audio, *van den Oord et al.*, 2016



SYSTEM PROMPT (HUMAN-WRITTEN)

A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.

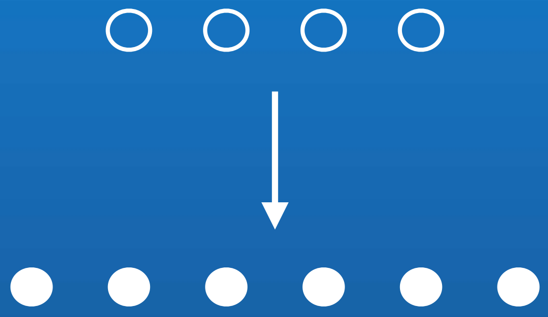
MODEL COMPLETION (MACHINE-WRITTEN, FIRST TRY)

```
The incident occurred on the downtown train line, which runs from
Covington and Ashland stations.

In an email to Ohio news outlets, the U.S. Department of Energy
said it is working with the Federal Railroad Administration to
find the thief.

“The theft of this nuclear material will have significant negative
consequences on public and environmental health, our workforce and
the economy of our nation,” said Tom Hicks, the U.S. Energy
Secretary, in a statement. “Our top priority is to secure the
theft and ensure it doesn’t happen again.”

The stolen material was taken from the University of Cincinnati’s
Research Triangle Park nuclear research site, according to a news
release from Department officials.

The Nuclear Regulatory Commission did not immediately release any
information.

According to the release, the U.S. Department of Energy’s Office
of Nuclear Material Safety and Security is leading that team’s
investigation.

“The safety of people, the environment and the nation’s nuclear
stockpile is our highest priority,” Hicks said. “We will get to
the bottom of this and make no excuses.
```

Attention is All You Need, Vaswani *et al.*, 2017

Improving Language Understanding by Generative Pre-Training, Radford *et al.*, 2018

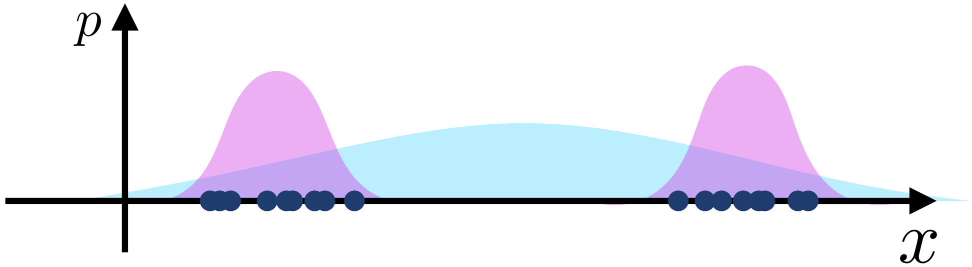
Language Models as Unsupervised Multi-task Learners, Radford *et al.*, 2019



explicit

latent variable models

latent variables result in mixtures of distributions



approach 1

directly fit a distribution to the data

$$p_\theta(x) = \mathcal{N}(x; \mu, \sigma^2)$$

approach 2

use a latent variable to model the data

$$p_\theta(x, z) = p_\theta(x|z)p_\theta(z) = \mathcal{N}(x; \mu_x(z), \sigma_x^2(z))\mathcal{B}(z; \mu_z)$$

$$p_\theta(x) = \sum_z p_\theta(x, z)$$

$$= \underbrace{\mu_z \cdot \mathcal{N}(x; \mu_x(1), \sigma_x^2(1))}_{\text{mixture component}} + \underbrace{(1 - \mu_z) \cdot \mathcal{N}(x; \mu_x(0), \sigma_x^2(0))}_{\text{mixture component}}$$

probabilistic graphical models provide a framework for modeling relationships between random variables

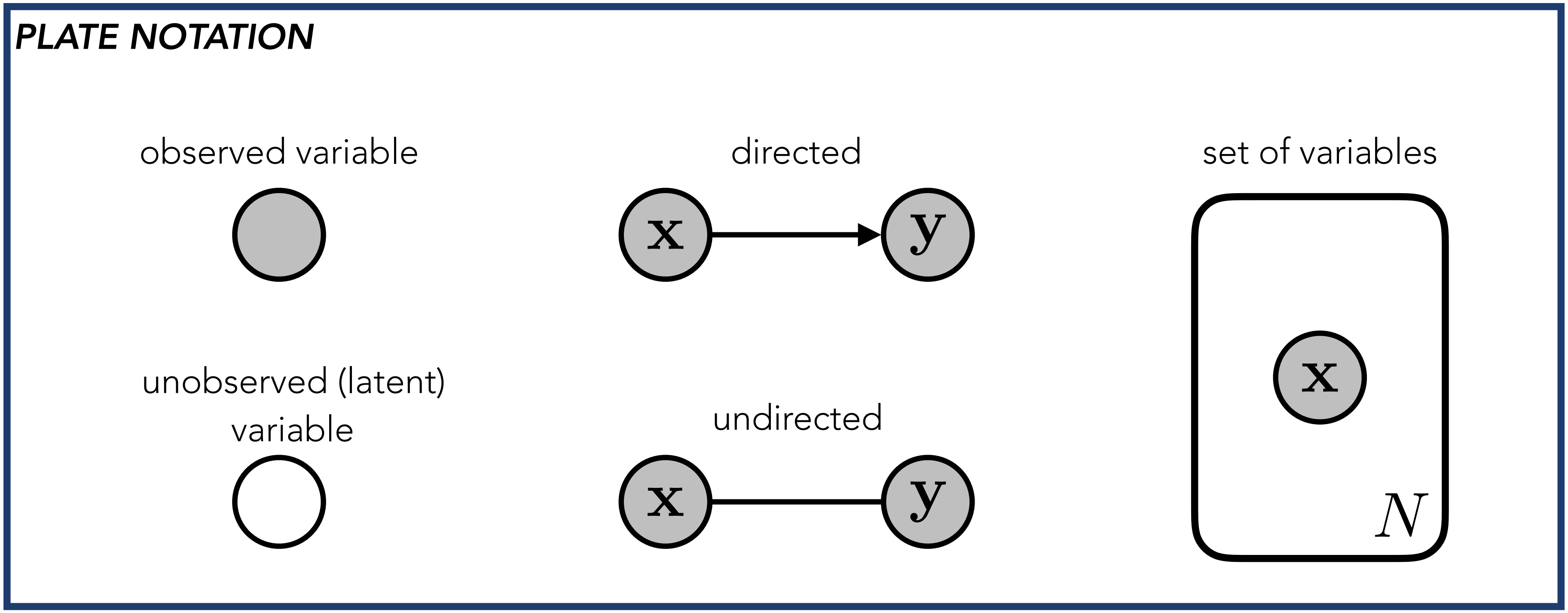


PLATE NOTATION

observed variable

unobserved (latent) variable

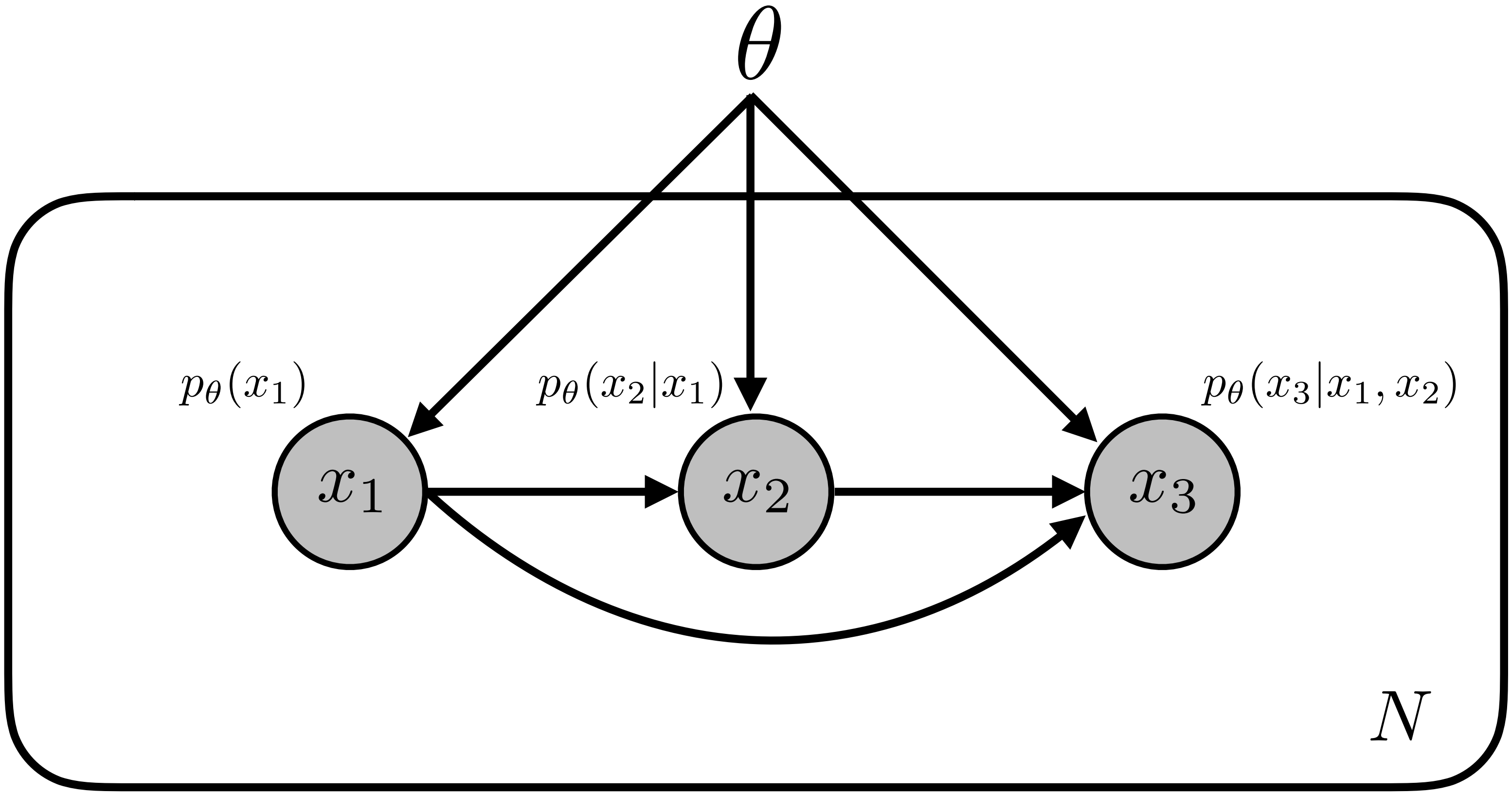
directed

undirected

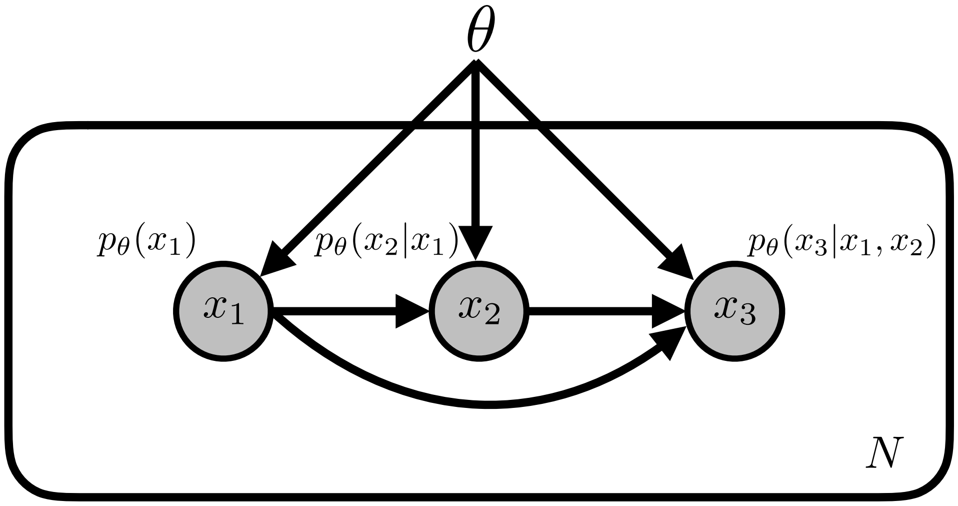
set of variables

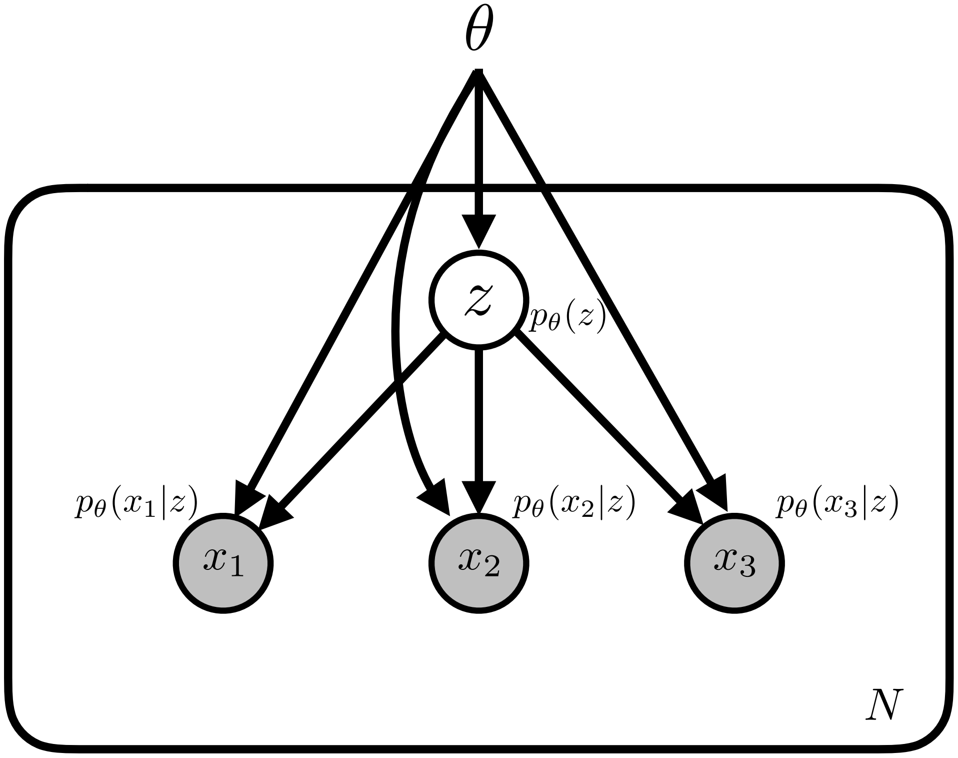
question

represent an auto-regressive model of 3 random variables with plate notation



comparing *auto-regressive models* and *latent variable models*

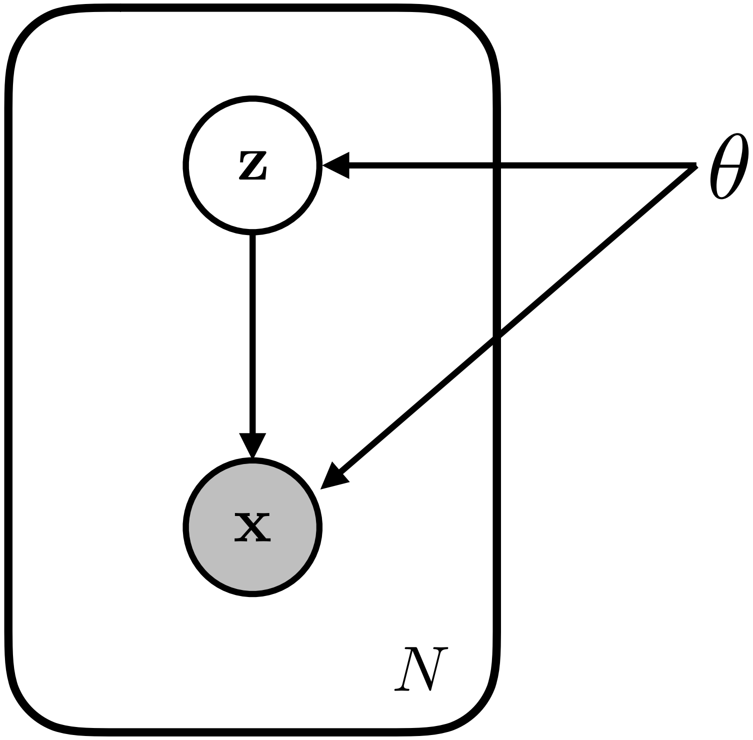




auto-regressive model

latent variable model

directed latent variable model



Generation

GENERATIVE MODEL

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$$

joint

conditional likelihood

prior

1. sample $\mathbf{z}$ from $p(\mathbf{z})$
2. use $\mathbf{z}$ samples to sample $\mathbf{x}$ from $p(\mathbf{x}|\mathbf{z})$

intuitive example: graphics engine

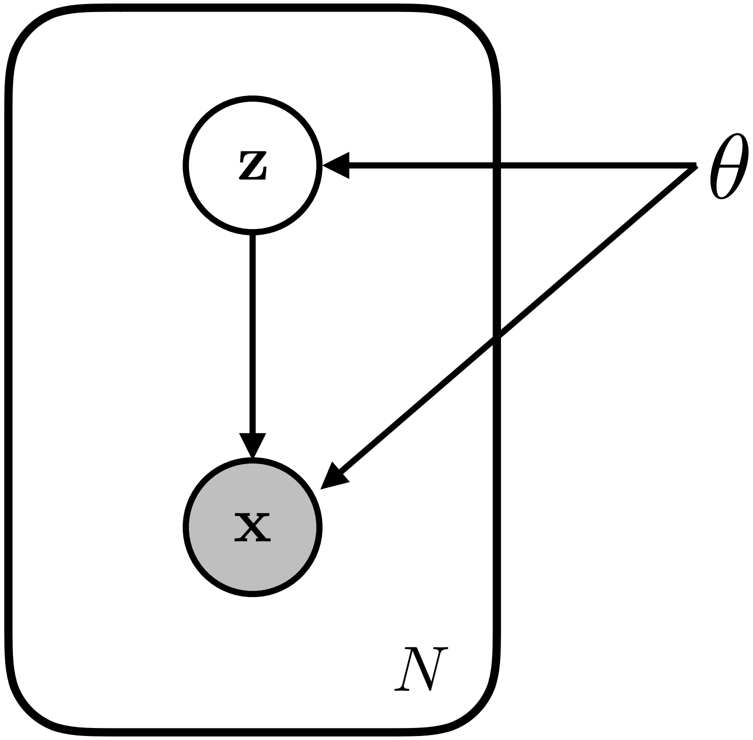
object ~ p(objects)

lighting ~ p(lighting)

background ~ p(bg)

RENDER

directed latent variable model



Posterior Inference

INFERENCE

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{x})}$$

joint

posterior

marginal likelihood

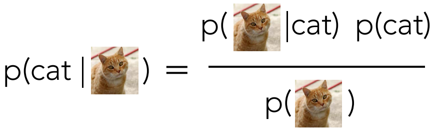
use Bayes' rule

provides conditional distribution over latent variables

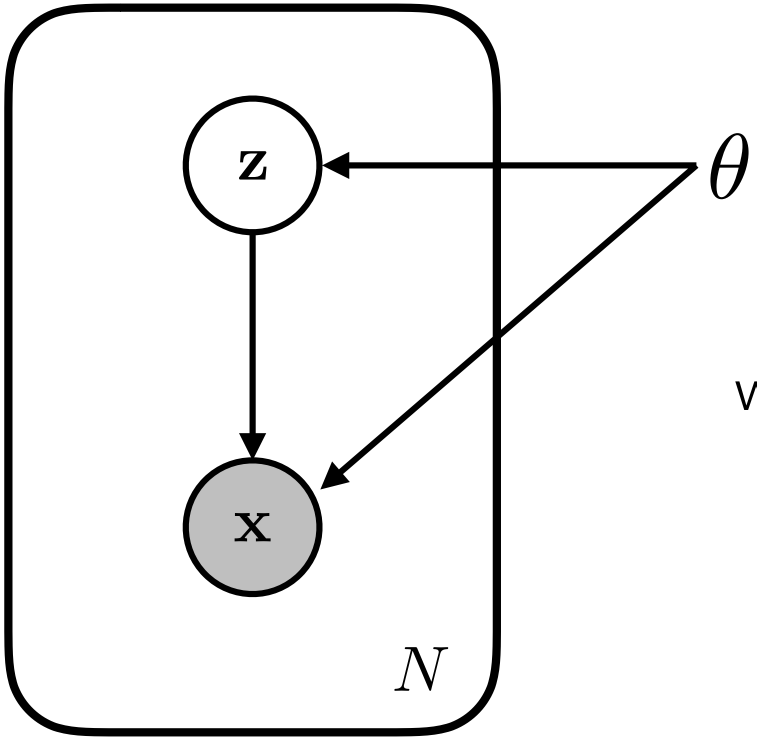
intuitive example

observation

what is the probability that I am observing a cat given these pixel observations?



<u>directed</u> latent variable model



Model Evaluation

MARGINALIZATION

marginal likelihood

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

joint

to evaluate the likelihood of an observation, we need to *marginalize* over all latent variables

i.e. consider all possible underlying states

intuitive example

observation

how likely is this observation under my model?
(what is the probability of observing this?)

for all objects, lighting, backgrounds, etc.:

how plausible is this example?

maximum likelihood estimation

maximize the log-likelihood (under the model) of the true data examples

$$\theta^* = \arg\max_\theta \; \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[\log p_\theta(\mathbf{x})\right] \approx \frac{1}{N} \sum_{i=1}^{N} \log p_\theta(\mathbf{x}^{(i)})$$

for latent variable models:

discrete

$$\log p_\theta(\mathbf{x}) = \log \sum_z p_\theta(\mathbf{x}, \mathbf{z})$$

or

continuous

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

marginalizing is often intractable in practice

variational inference

lower bound the log-likelihood by introducing an approximate posterior

introduce an ***approximate posterior*** $q(\mathbf{z}|\mathbf{x})$

$$\log p_\theta(\mathbf{x}) = \mathcal{L}(\mathbf{x}) + D_{KL}(q(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$$

where $\mathcal{L}(\mathbf{x}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}|\mathbf{x})\right]$

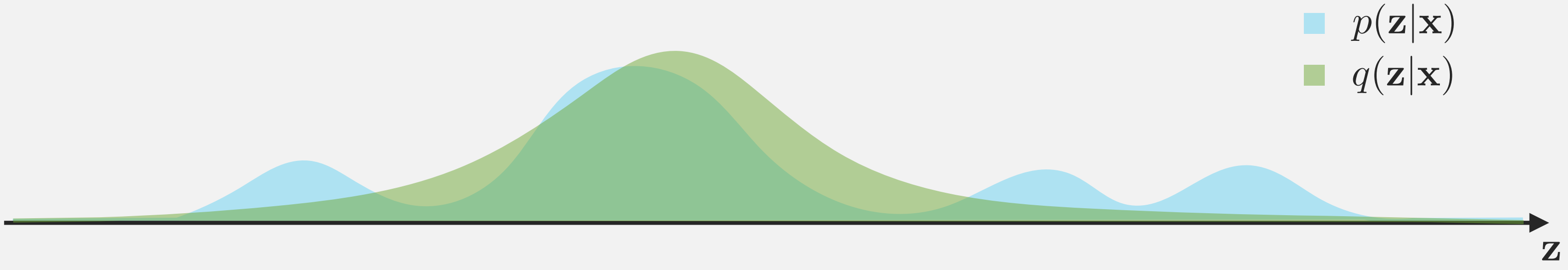
$$D_{KL} \geq 0 \longrightarrow \mathcal{L}(\mathbf{x}) \leq \log p_\theta(\mathbf{x})$$ (lower bound)

variational expectation maximization (EM)

E-Step: optimize $\mathcal{L}(\mathbf{x})$ *w.r.t.* $q(\mathbf{z}|\mathbf{x})$

M-Step: optimize $\mathcal{L}(\mathbf{x})$ *w.r.t.* θ

the E-Step indirectly minimizes $D_{KL}(q(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$



interpreting the lower bound

we can write the lower bound as

$$
\begin{aligned}
\mathcal{L} &\equiv \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x})\right] \\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x})\right] \\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x})\right] \\
&= \underbrace{\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z})\right]}_{\text{reconstruction}} - \underbrace{D_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))}_{\text{regularization}}
\end{aligned}
$$

$q(\mathbf{z}|\mathbf{x})$ is optimized to represent the data while staying close to the prior

connections to *compression, information theory*

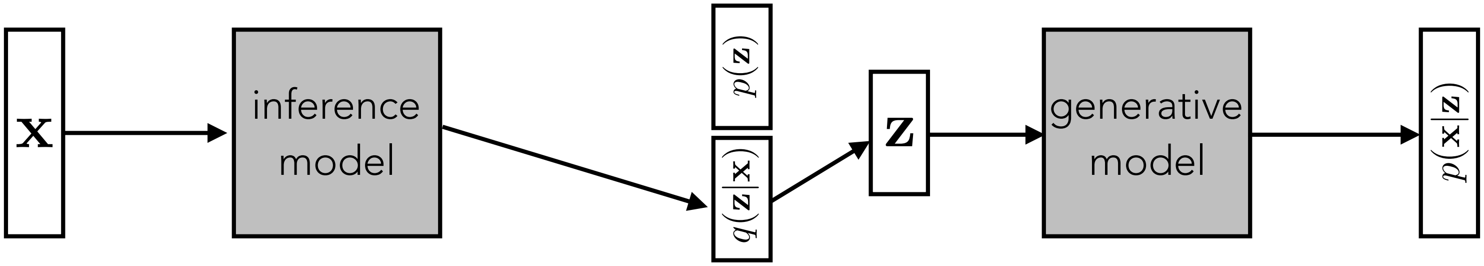
variational autoencoder (VAE)

variational expectation maximization (EM)

E-Step: optimize $\mathcal{L}(\mathbf{x})$ *w.r.t.* $q(\mathbf{z}|\mathbf{x})$

M-Step: optimize $\mathcal{L}(\mathbf{x})$ *w.r.t.* θ

use a separate ***inference model*** to directly output approximate posterior estimates



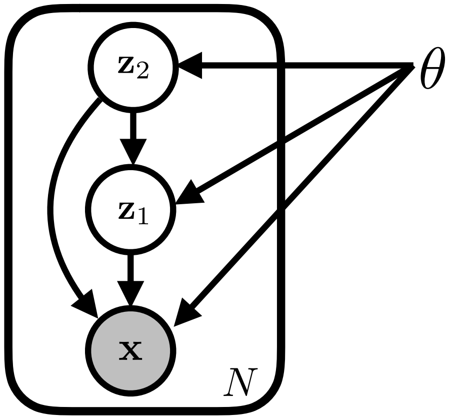
learn both models jointly using *stochastic backpropagation*

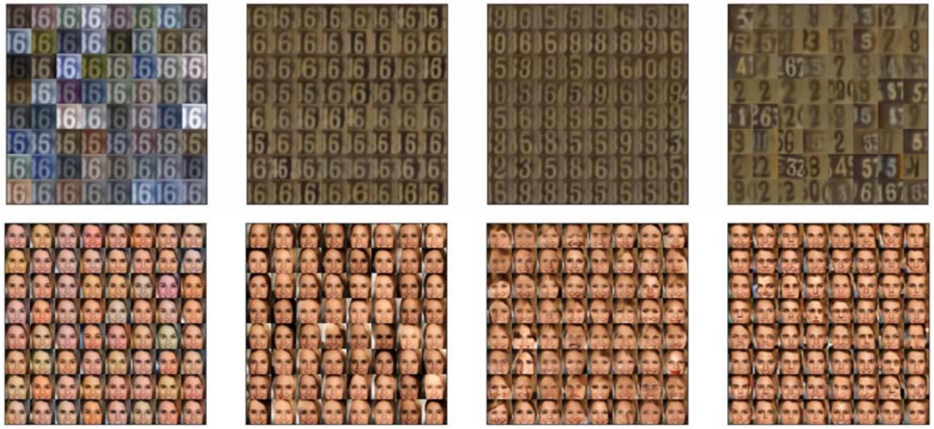
reparametrization trick: $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Autoencoding Variational Bayes, Kingma & Welling, 2014

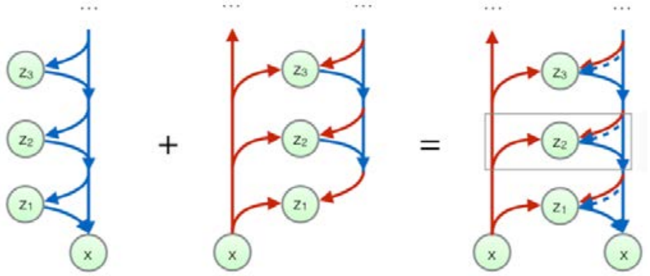
Stochastic Backpropagation, Rezende *et al.*, 2014

<u>hierarchical</u> latent variable models



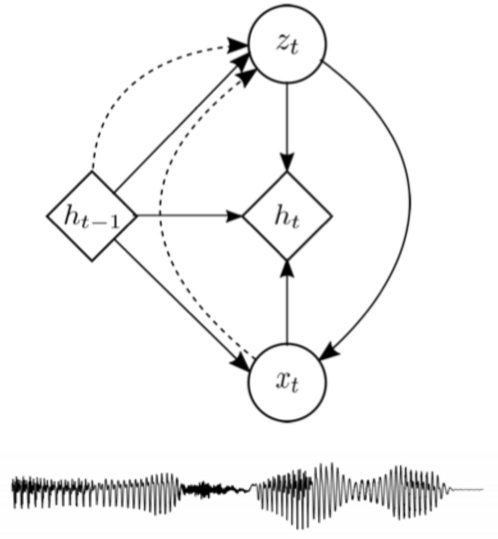


Learning Hierarchical Features from Generative Models, *Zhao et al.*, 2017

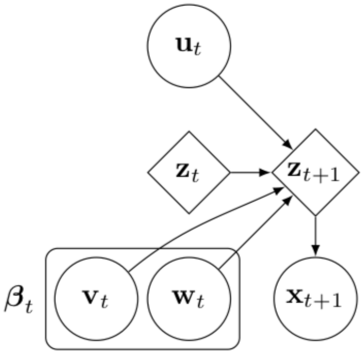


Improving Variational Inference with Inverse Auto-regressive Flow, *Kingma et al.*, 2016

<u>sequential</u> latent variable models

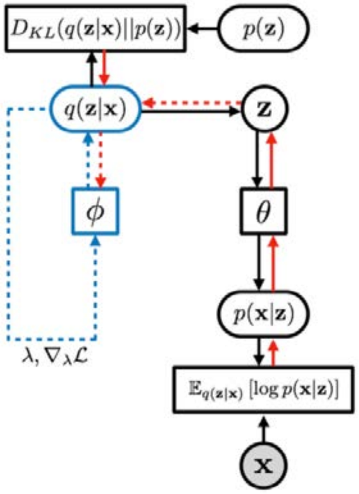


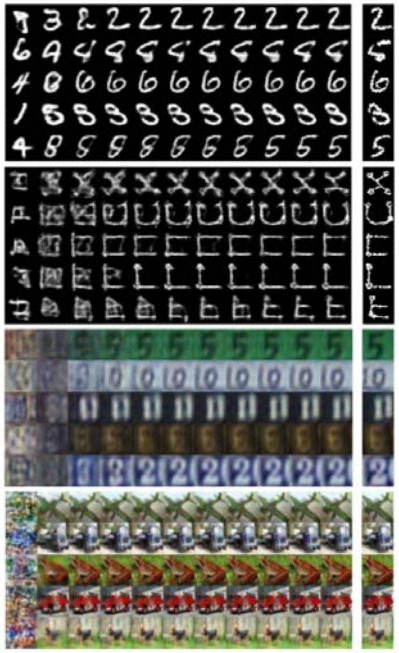
A Recurrent Latent Variable Model for Sequential Data, *Chung et al.*, 2015



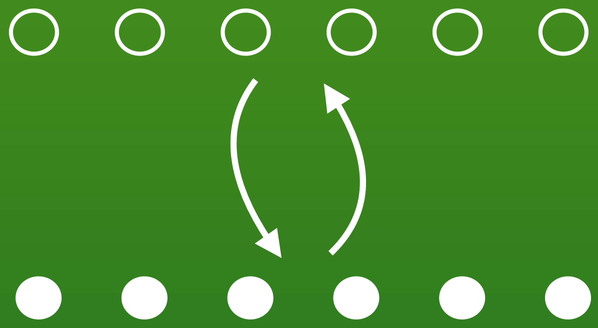
Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data, *Karl et al.*, 2016

iterative inference models


$D_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$
$p(\mathbf{z})$
$q(\mathbf{z}|\mathbf{x})$
$\mathbf{z}$
ϕ
θ
$\lambda, \nabla_\lambda \mathcal{L}$
$p(\mathbf{x}|\mathbf{z})$
$\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})]$
$\mathbf{x}$



Iterative Amortized Inference, *Marino et al.*, 2018

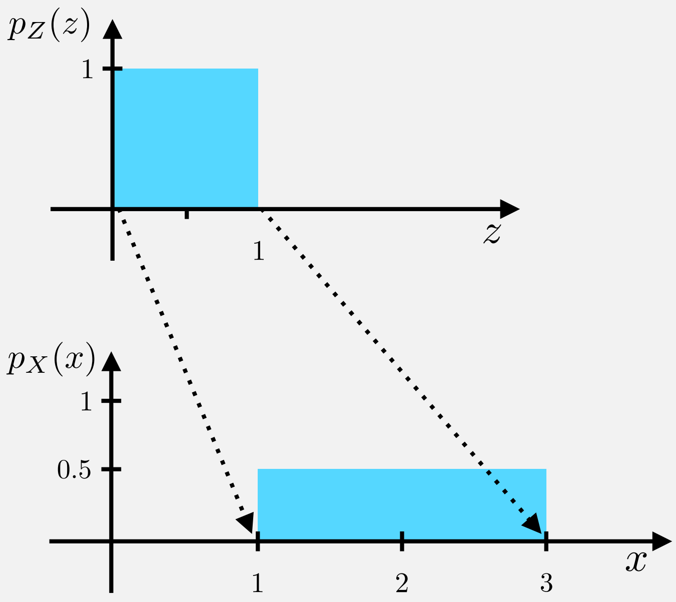


invertible explicit latent variable models

change of variables

use an invertible mapping to directly evaluate the log likelihood

simple example



sample z from a *base distribution*

$$z \sim p_Z(z) = \mathrm{Uniform}(0, 1)$$

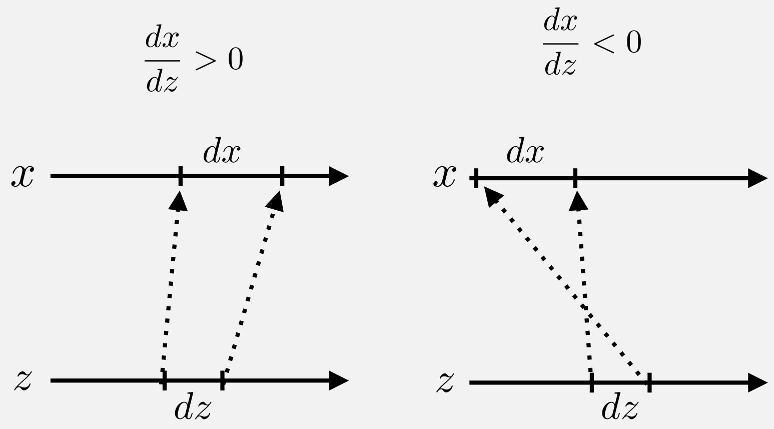
apply a transform to z to get a *transformed distribution*

$$x = f(z) = 2z + 1$$

$$p_X(x)dx = p_Z(z)dz$$

$$p_X(x) = p_Z(z)\left|\frac{dz}{dx}\right|$$

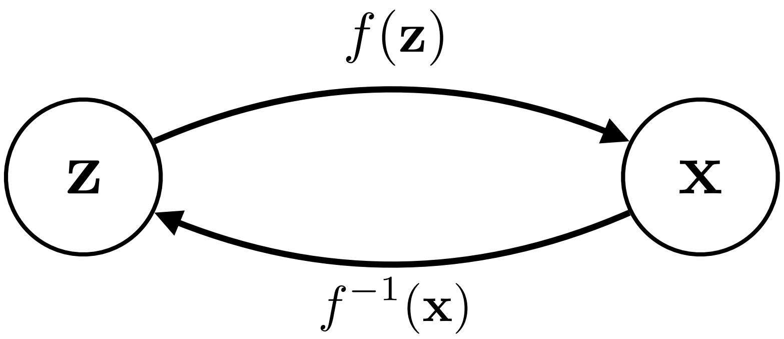
conservation of probability mass



change of variables

$p_Z(\mathbf{z})$

base distribution



$p_X(\mathbf{x})$

transformed distribution

change of variables formula

$$p_X(\mathbf{x}) = p_Z(\mathbf{z}) \left|\det \mathbf{J}(f^{-1}(\mathbf{x}))\right|$$

or

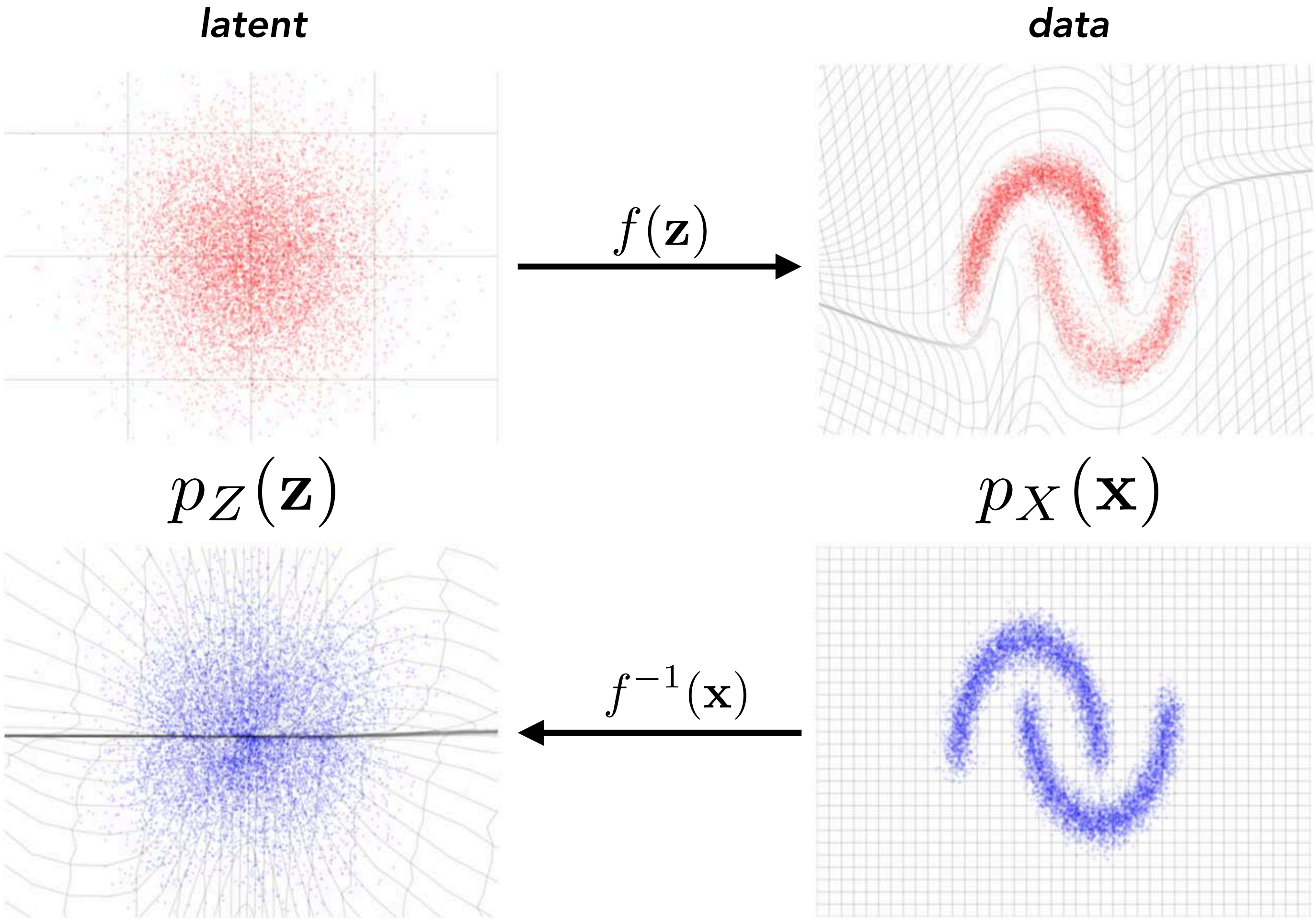
$$\log p_X(\mathbf{x}) = \log p_Z(\mathbf{z}) + \log \left|\det \mathbf{J}(f^{-1}(\mathbf{x}))\right|$$

$\mathbf{J}(f^{-1}(\mathbf{x}))$ is the *Jacobian* matrix of the inverse transform

$\det \mathbf{J}(f^{-1}(\mathbf{x}))$ *is the local distortion in volume from the transform*

change of variables

transform the data into a space that is easier to model



latent

data

$f(\mathbf{z})$

$p_Z(\mathbf{z})$

$p_X(\mathbf{x})$

$f^{-1}(\mathbf{x})$

Density Estimation Using Real NVP, Dinh *et al.*, 2016

maximum likelihood estimation

maximize the log-likelihood (under the model) of the true data examples

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] \approx \frac{1}{N}\sum_{i=1}^{N} \log p_\theta(\mathbf{x}^{(i)})$$

for invertible latent variable models:

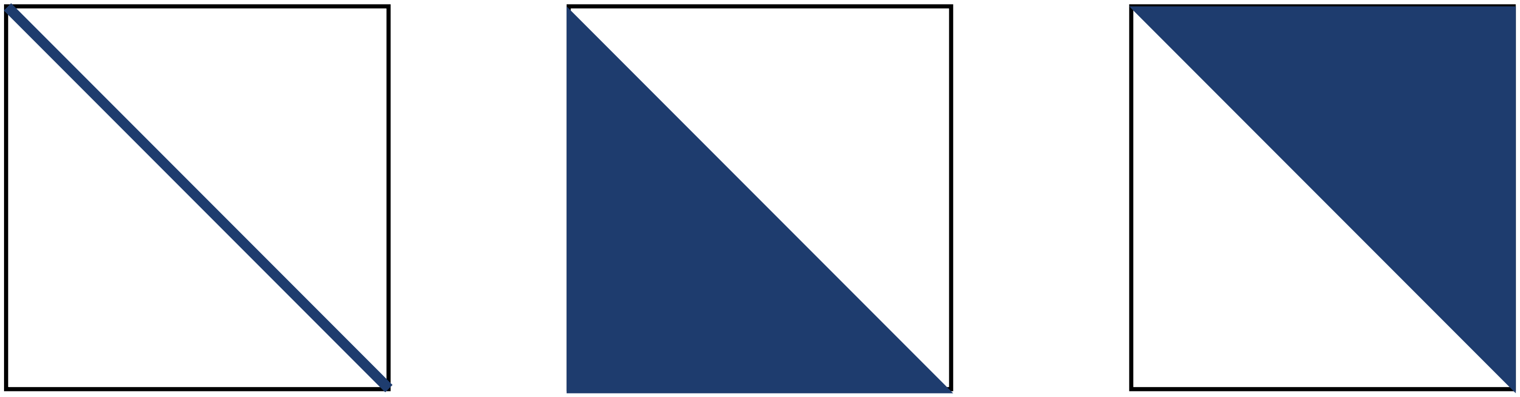
$$\log p_\theta(\mathbf{x}) = \log p_\theta(\mathbf{z}) + \log \left|\det \mathbf{J}(f_\theta^{-1}(\mathbf{x}))\right|$$

$$\theta^* = \arg\max_{\theta} \frac{1}{N}\sum_{i=1}^{N} \left[\log p_\theta(\mathbf{z}^{(i)}) + \log \left|\det \mathbf{J}(f_\theta^{-1}(\mathbf{x}^{(i)}))\right|\right]$$

change of variables

to use the change of variables formula, we need to evaluate $\det \mathbf{J}(f^{-1}(\mathbf{x}))$

for an arbitrary $N \times N$ Jacobian matrix, this is worst case $O(N^3)$



restrict the transforms to those with diagonal or triangular inverse Jacobians

allows us to compute $\det \mathbf{J}(f^{-1}(\mathbf{x}))$ in $O(N)$

→ *product of diagonal entries*

masked autoregressive flow (MAF)

autoregressive sampling can be interpreted as a transformed distribution

$$x_i \sim \mathcal{N}(x_i; \mu_i(\mathbf{x}_{1:i-1}), \sigma_i^2(\mathbf{x}_{1:i-1})) \longrightarrow x_i = \mu_i(\mathbf{x}_{1:i-1}) + \sigma_i(\mathbf{x}_{1:i-1}) \cdot z_i$$

where $z_i \sim \mathcal{N}(z_i; 0, 1)$

must generate each x_i *sequentially*

however, we can parallelize the inverse transform:

$$z_i = \frac{x_i - \mu_i(\mathbf{x}_{1:i-1})}{\sigma_i(\mathbf{x}_{1:i-1})}$$

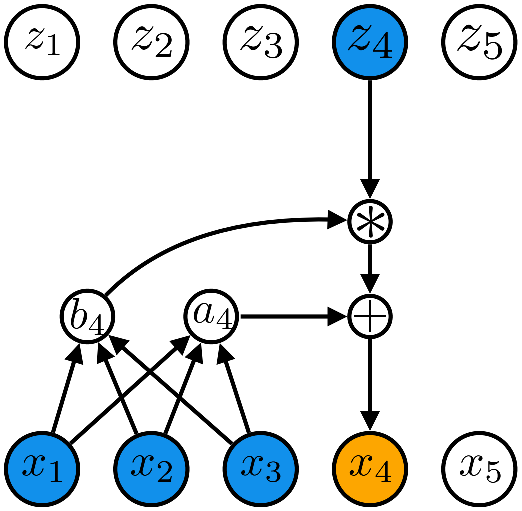
Masked Autoregressive Flow, Papamakarios *et al.*, 2017

see also **Inverse Autoregressive Flow**, Kingma *et al.*, 2016

masked autoregressive flow (MAF)

TRANSFORM

base distribution

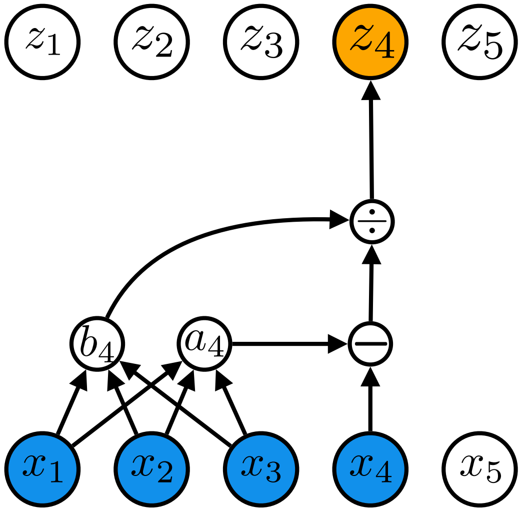


transformed distribution

$$x_4 = a_4(\mathbf{x}_{1:3}) + b_4(\mathbf{x}_{1:3}) \cdot z_4$$

INVERSE TRANSFORM

base distribution



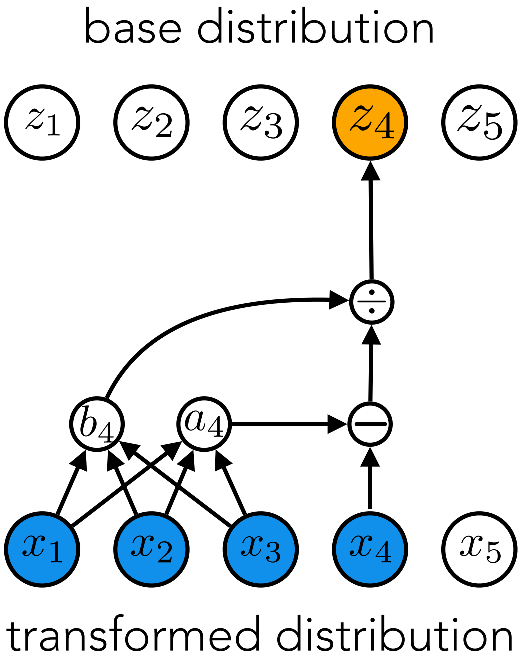
transformed distribution

$$z_4 = \frac{x_4 - a_4(\mathbf{x}_{1:3})}{b_4(\mathbf{x}_{1:3})}$$

Masked Autoregressive Flow, Papamakarios *et al.*, 2017

question

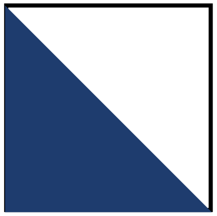
INVERSE TRANSFORM



$$z_4 = \frac{x_4 - a_4(\mathbf{x}_{1:3})}{b_4(\mathbf{x}_{1:3})}$$

What is the form of $\mathbf{J}(f^{-1}(\mathbf{x}))$?

lower triangular



each z_i only depends on $\mathbf{x}_{1:i}$

What is $\det \mathbf{J}(f^{-1}(\mathbf{x}))$?

product of diagonal elements of $\mathbf{J}(f^{-1}(\mathbf{x}))$

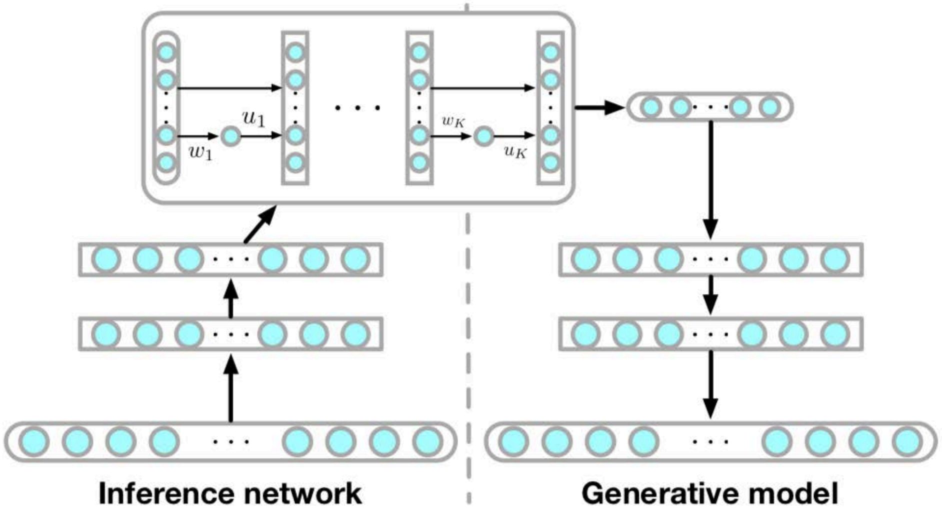
$$\det \mathbf{J}(f^{-1}(\mathbf{x})) = \prod_i \frac{1}{b_i(\mathbf{x}_{1:i})}$$

Masked Autoregressive Flow, Papamakarios *et al.*, 2017

normalizing flows (NF)

can also use the change of variables formula for variational inference

parameterize $q(\mathbf{z}|\mathbf{x})$ as a transformed distribution

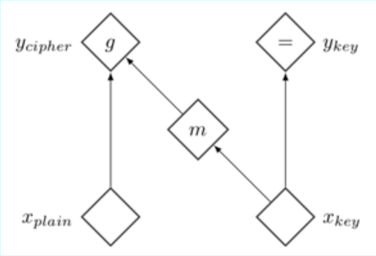


use more complex approximate posterior, but evaluate a simpler distribution

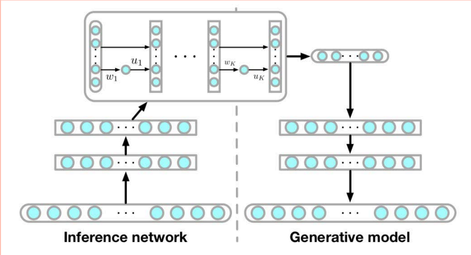
Normalizing Flows, Rezende & Mohamed, 2015

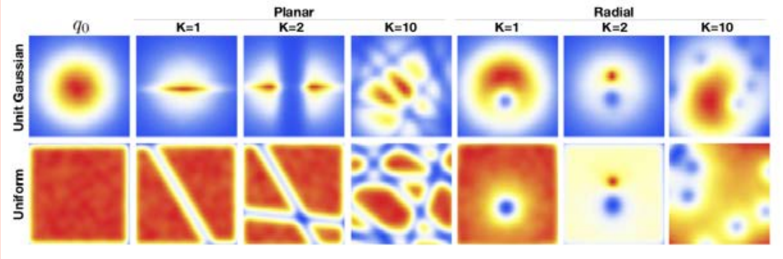
some recent work

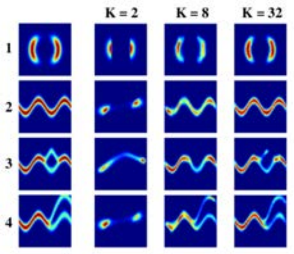
NICE: Non-linear Independent Components Estimation, Dinh *et al.*, 2014



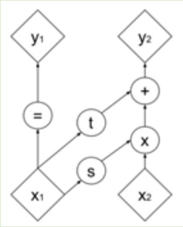
Variational Inference with Normalizing Flows, Rezende & Mohamed, 2015



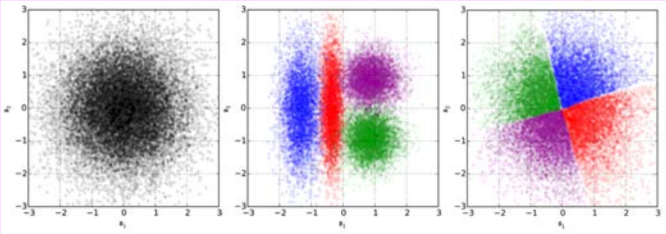


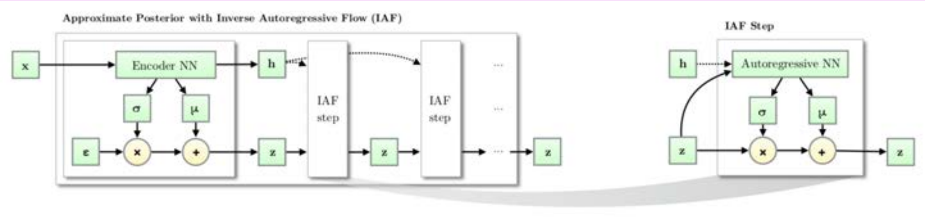


Density Estimation Using Real NVP, Dinh *et al.*, 2016



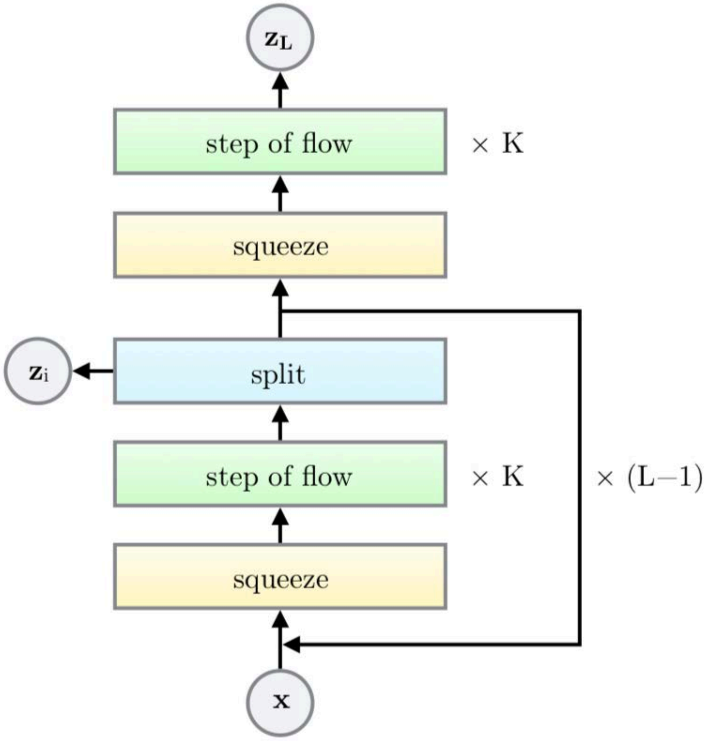
Improving Variational Inference with Inverse Autoregressive Flow, Kingma *et al.*, 2016





Glow

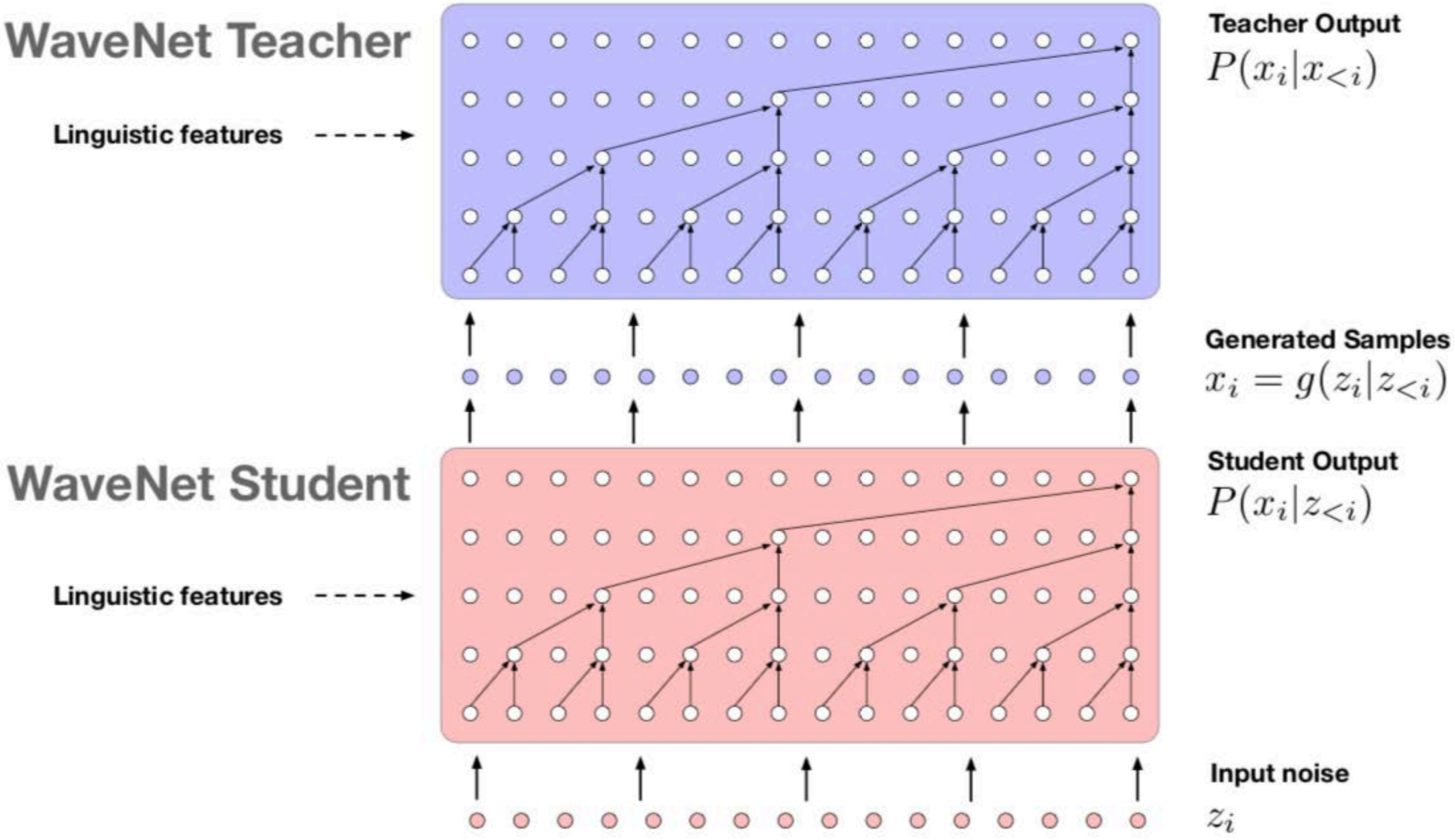
use 1 x 1 convolutions to perform transform



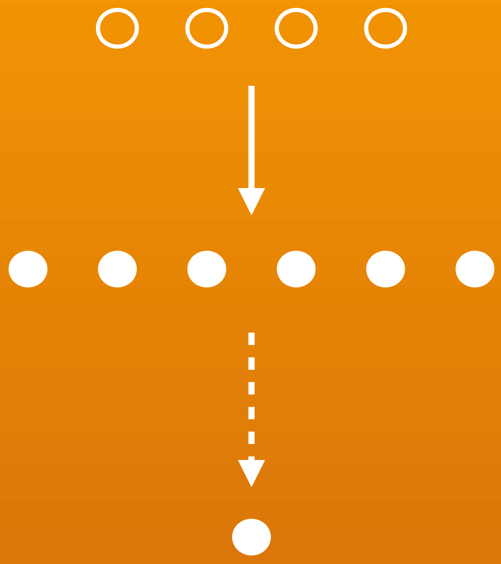
Glow, Kingma & Dhariwal, 2018

Parallel WaveNet

distill an autoregressive distribution into a parallel transform



Parallel WaveNet, van den Oord *et al.*, 2018

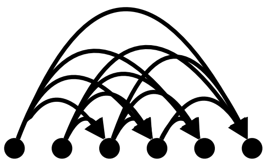


implicit

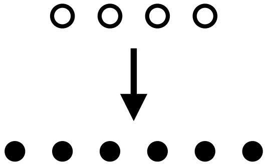
latent variable models

many generative models are defined in terms of an *explicit* likelihood

in which $p_\theta(\mathbf{x})$ has a parametric form



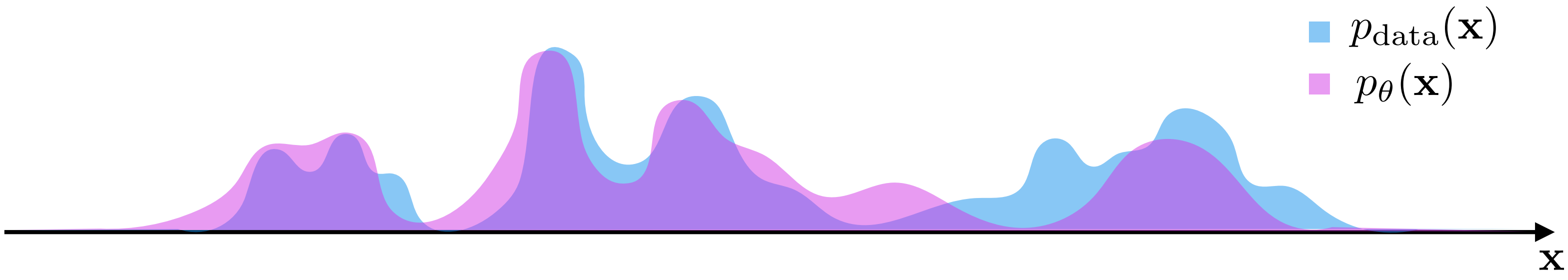
$p_\theta(x_i|\mathbf{x}_{<i})$



$p_\theta(\mathbf{x}|\mathbf{z})$

this may limit the types of distributions that can be learned

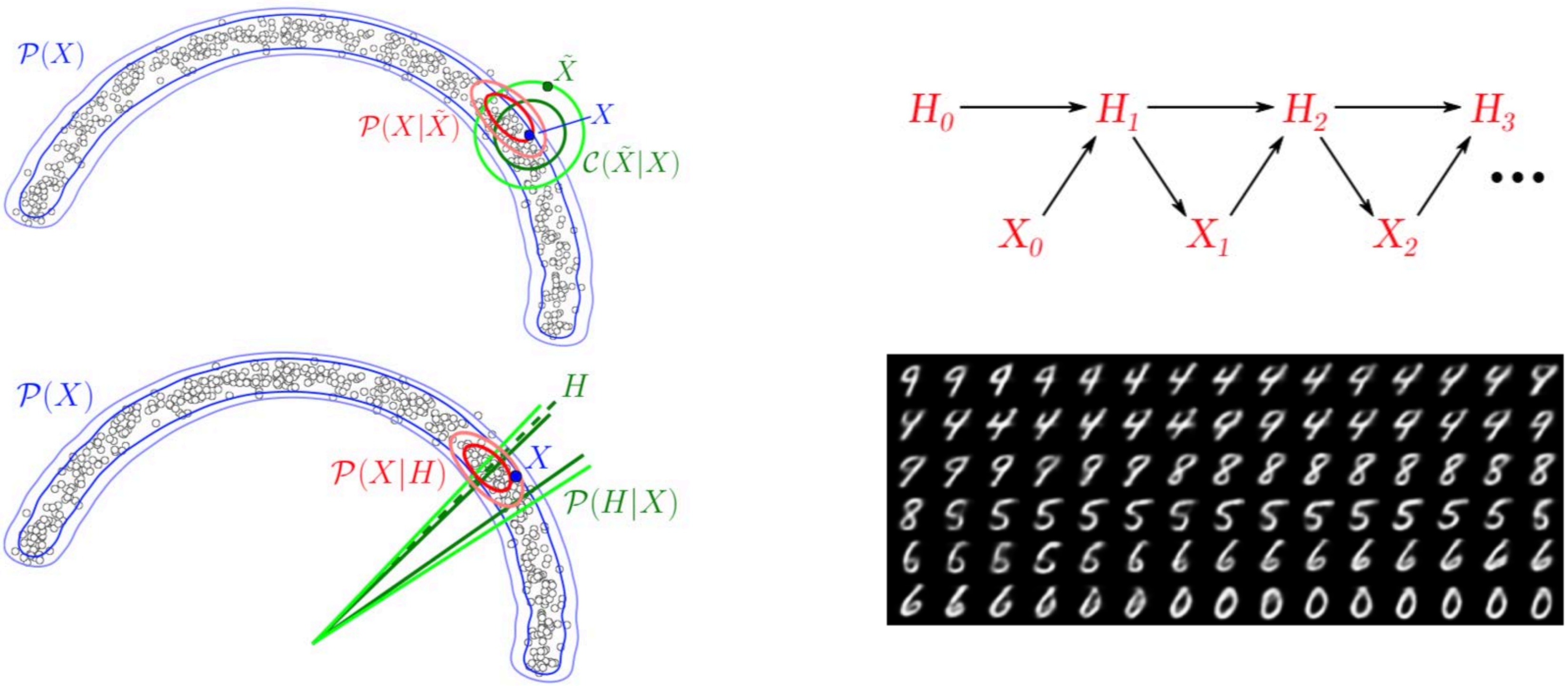
instead of using an *explicit* probability density,
learn a model that defines an *implicit density*



specify a stochastic procedure for generating the data
that does not require an explicit likelihood evaluation

Learning in Implicit Generative Models, Mohamed & Lakshminarayanan, 2016

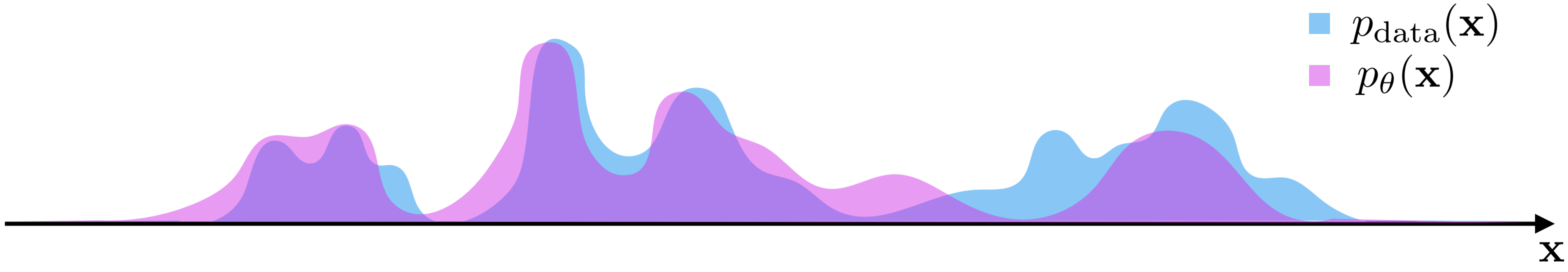
Generative Stochastic Networks (GSNs)



Deep Generative Stochastic Networks Trainable by Backprop, Bengio *et al.*, 2013

train an auto-encoder to learn Monte Carlo sampling transitions

the generative distribution is *implicitly* defined by this transition



estimate density ratio through *hypothesis testing*

data distribution $p_{\text{data}}(\mathbf{x})$ generated distribution $p_\theta(\mathbf{x})$

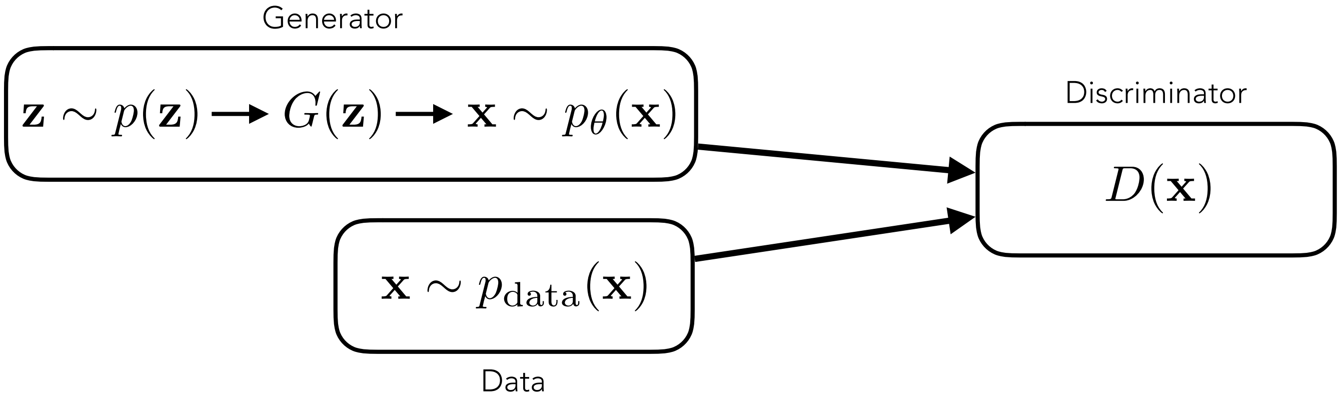
$$\frac{p_{\text{data}}(\mathbf{x})}{p_\theta(\mathbf{x})} = \frac{p(\mathbf{x}|y = \text{data})}{p(\mathbf{x}|y = \text{model})}$$

$$\frac{p_{\text{data}}(\mathbf{x})}{p_\theta(\mathbf{x})} = \frac{p(y = \text{data}|\mathbf{x})p(\mathbf{x})/p(y = \text{data})}{p(y = \text{model}|\mathbf{x})p(\mathbf{x})/p(y = \text{model})}$$ (Bayes' rule)

$$\frac{p_{\text{data}}(\mathbf{x})}{p_\theta(\mathbf{x})} = \frac{p(y = \text{data}|\mathbf{x})}{p(y = \text{model}|\mathbf{x})}$$ (assuming equal dist. prob.)

density estimation becomes a sample discrimination task

Generative Adversarial Networks (GANs)


Generator
$\mathbf{z} \sim p(\mathbf{z}) \rightarrow G(\mathbf{z}) \rightarrow \mathbf{x} \sim p_\theta(\mathbf{x})$
Discriminator
$D(\mathbf{x})$
$\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$
Data

Generator: $G(\mathbf{z})$

Discriminator: $D(\mathbf{x}) = \hat{p}(y = \text{data}|\mathbf{x}) = 1 - \hat{p}(y = \text{model}|\mathbf{x})$

Log-Likelihood:
$$\begin{aligned}
&\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\log \hat{p}(y = \text{data}|\mathbf{x})\right] + \mathbb{E}_{p_\theta(\mathbf{x})}\left[\log \hat{p}(y = \text{model}|\mathbf{x})\right] \\
&= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\log D(\mathbf{x})\right] + \mathbb{E}_{p_\theta(\mathbf{x})}\left[\log(1 - D(\mathbf{x}))\right] \\
&= \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\log D(\mathbf{x})\right] + \mathbb{E}_{p(\mathbf{z})}\left[\log(1 - D(G(\mathbf{z})))\right]
\end{aligned}$$

Minimax:
$$\min_G \max_D \mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\log D(\mathbf{x})\right] + \mathbb{E}_{p(\mathbf{z})}\left[\log(1 - D(G(\mathbf{z})))\right]$$

Ian *Goodfellow*, 2016
Shakir *Mohamed*, 2016

Generative Adversarial Networks (GANs)

Minimax: $\min_G \max_D \mathbb{E}_{p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{p(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]$

GANs minimize the *Jensen-Shannon Divergence*:

For a fixed $G(\mathbf{z})$ the optimal discriminator is $D^*(\mathbf{x}) = \dfrac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_\theta(\mathbf{x})}$

Plugging this into the objective

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} [\log D^*(\mathbf{x})] + \mathbb{E}_{p_\theta(\mathbf{x})} [\log(1 - D^*(\mathbf{x}))]$$

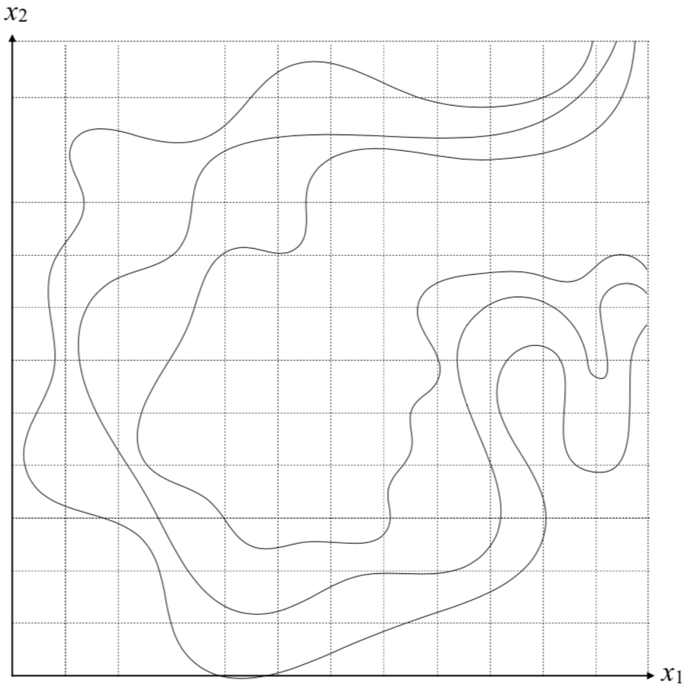
$$= \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[\log \left(\frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_\theta(\mathbf{x})}\right)\right] + \mathbb{E}_{p_\theta(\mathbf{x})} \left[\log \left(\frac{p_\theta(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_\theta(\mathbf{x})}\right)\right]$$

$$= \log\left(\frac{1}{4}\right) + D_{KL}\left(p_{\text{data}}(\mathbf{x}) \middle|\middle| \frac{p_{\text{data}}(\mathbf{x}) + p_\theta(\mathbf{x})}{2}\right) + D_{KL}\left(p_\theta(\mathbf{x}) \middle|\middle| \frac{p_{\text{data}}(\mathbf{x}) + p_\theta(\mathbf{x})}{2}\right)$$

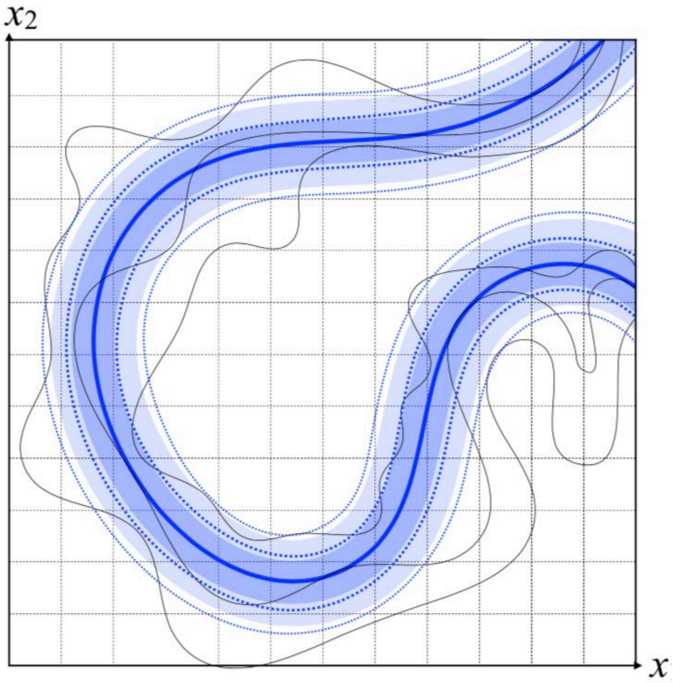
$$= \log\left(\frac{1}{4}\right) + 2 \cdot D_{JS}(p_{\text{data}}(\mathbf{x}) || p_\theta(\mathbf{x}))$$

Generative Adversarial Networks, *Goodfellow et al.*, 2014

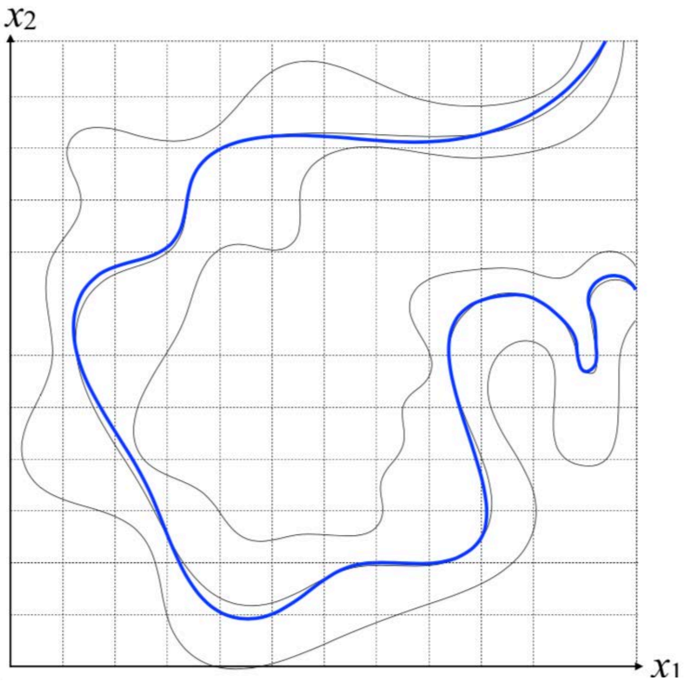
interpretation



data manifold



explicit model



implicit model

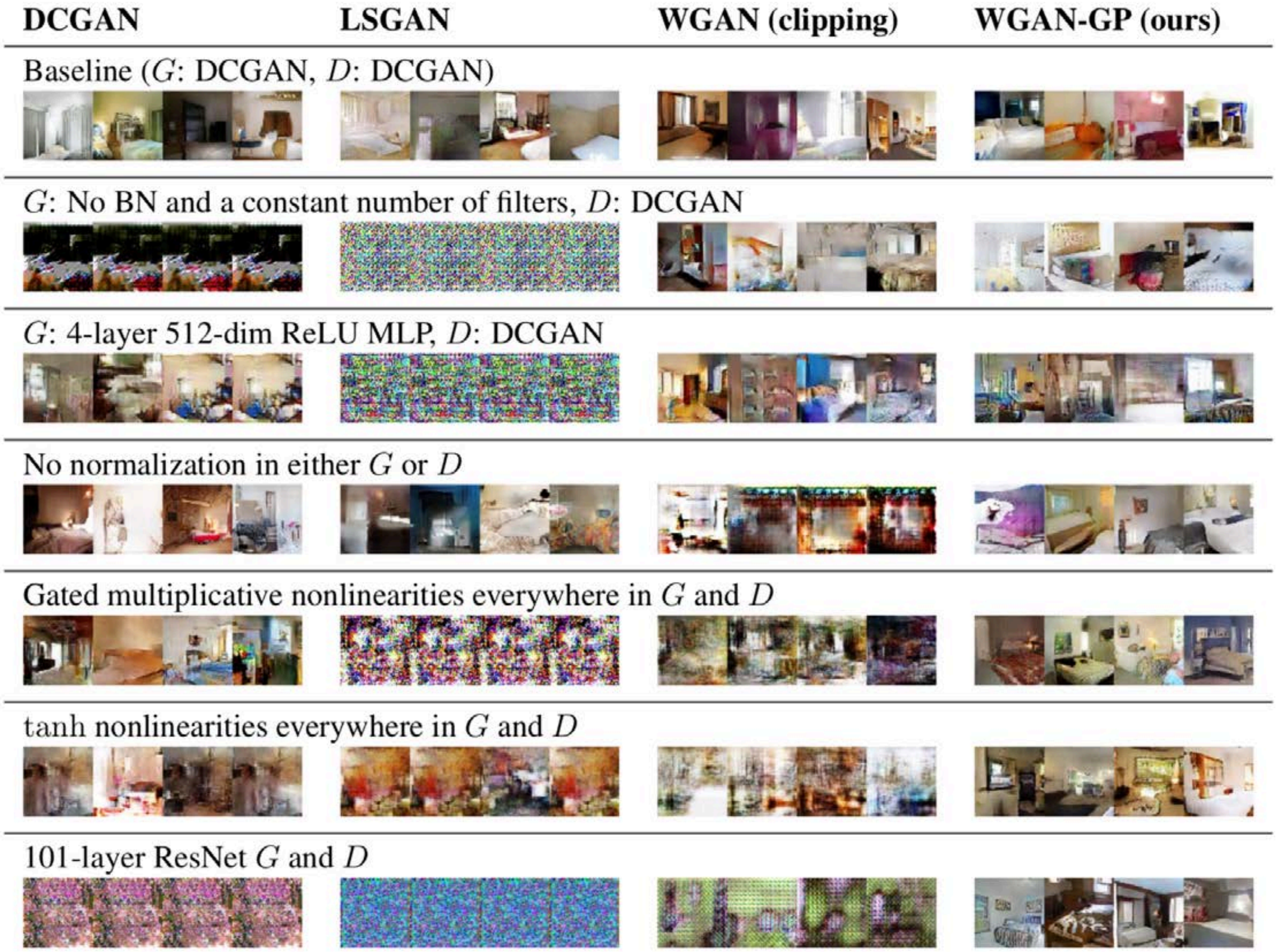
explicit models tend to cover the entire data manifold, but are constrained

implicit models tend to capture part of the data manifold, but can neglect other parts

→ *"mode collapse"*

Generative Adversarial Networks (GANs)

GANs can be difficult to optimize

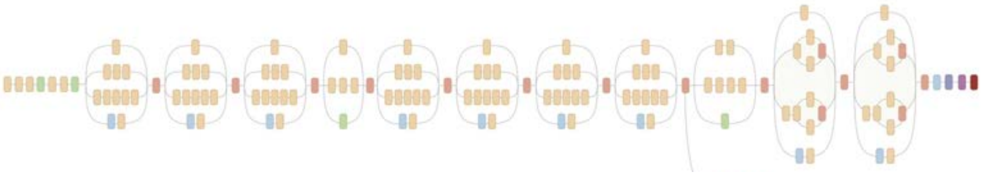

DCGAN
LSGAN
WGAN (clipping)
WGAN-GP (ours)
Baseline (G: DCGAN, D: DCGAN)
G: No BN and a constant number of filters, D: DCGAN
G: 4-layer 512-dim ReLU MLP, D: DCGAN
No normalization in either G or D
Gated multiplicative nonlinearities everywhere in G and D
tanh nonlinearities everywhere in G and D
101-layer ResNet G and D

Improved Training of Wasserstein GANs, Gulrajani et al., 2017

evaluation

without an explicit likelihood, it is difficult to quantify the performance

inception score



use a pre-trained Inception v3 model to quantify class and distribution entropy

$$\mathrm{IS}(G) = \exp\left(\mathbb{E}_{p(\tilde{\mathbf{x}})} D_{KL}(p(y|\tilde{\mathbf{x}})||p(y))\right)$$

$p(y|\tilde{\mathbf{x}})$ is the class distribution for a given image

⟶ should be highly peaked (low entropy)

$p(y) = \int p(y|\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$ is the marginal class distribution

⟶ want this to be uniform (high entropy)

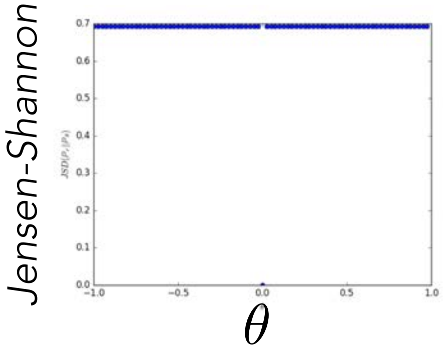
Improved Techniques for Training GANs, *Salimans et al.*, 2016

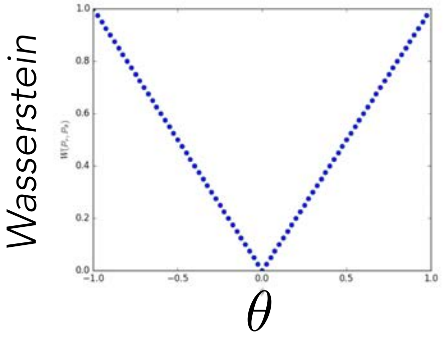
A Note on the Inception Score, *Barratt & Sharma*, 2018

Wasserstein GAN (W-GAN)

the Jenson-Shannon divergence can be **discontinuous**, making it difficult to train

θ is a gen. model parameter





instead use the Wasserstein distance, continuous and diff. almost everywhere:

$$W(p_{\text{data}}(\mathbf{x}), p_\theta(\mathbf{x})) = \inf_{\gamma \in \prod(p_{\text{data}}(\mathbf{x}), p_\theta(\mathbf{x}))} \mathbb{E}_{(\hat{\mathbf{x}}, \tilde{\mathbf{x}}) \sim \gamma} \left[|| \hat{\mathbf{x}} - \tilde{\mathbf{x}} || \right]$$

"minimum cost of transporting points between two distributions"

intractable to evaluate, but can instead constrain the discriminator

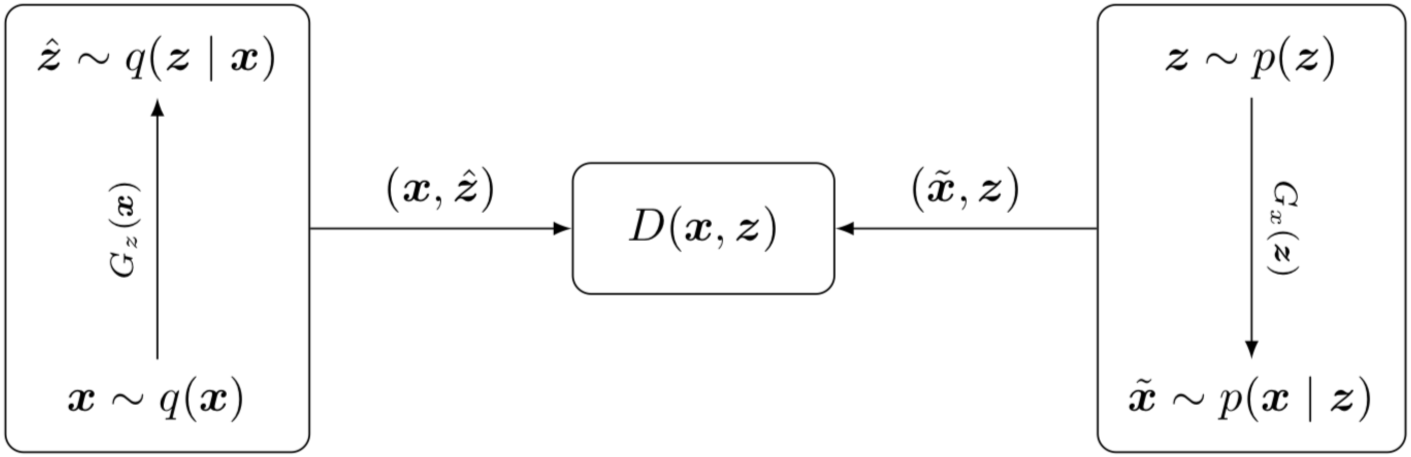
$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[D(\mathbf{x}) \right] - \mathbb{E}_{p_\theta(\mathbf{x})} \left[D(\mathbf{x}) \right]$$

$\mathcal{D}$ is the set of Lipschitz functions (bounded derivative),
enforced through weight clipping, gradient penalty, spectral normalization

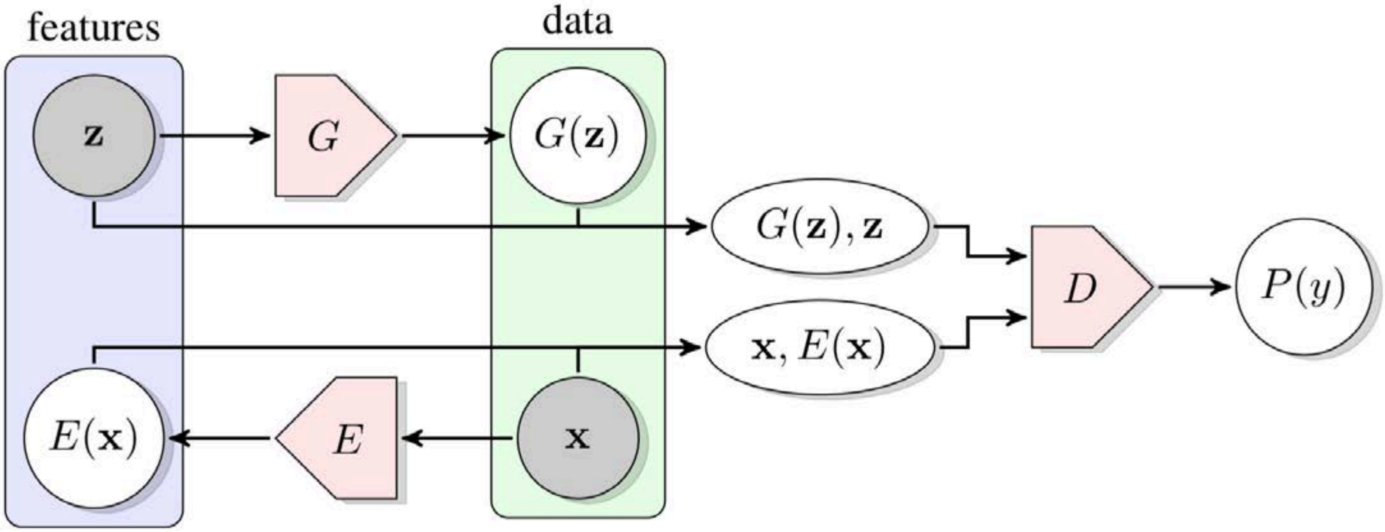
Wasserstein GANs, *Arjovsky et al.*, 2017
Improved Training of Wasserstein GANs, Gulrajani *et al.*, 2017
Spectral Normalization for GANs, Miyato *et al.*, 2018

extensions: inference

can we also learn to ***infer a latent representation****?*


$\hat{\boldsymbol{z}} \sim q(\boldsymbol{z} \mid \boldsymbol{x})$
$G_z(\boldsymbol{x})$
$\boldsymbol{x} \sim q(\boldsymbol{x})$
$(\boldsymbol{x}, \hat{\boldsymbol{z}})$
$D(\boldsymbol{x}, \boldsymbol{z})$
$(\tilde{\boldsymbol{x}}, \boldsymbol{z})$
$\boldsymbol{z} \sim p(\boldsymbol{z})$
$G_x(\boldsymbol{z})$
$\tilde{\boldsymbol{x}} \sim p(\boldsymbol{x} \mid \boldsymbol{z})$

Adversarially Learned Inference, *Dumoulin et al.*, 2017



Adversarial Feature Learning, *Donahue et al.*, 2017

applications

image to image translation

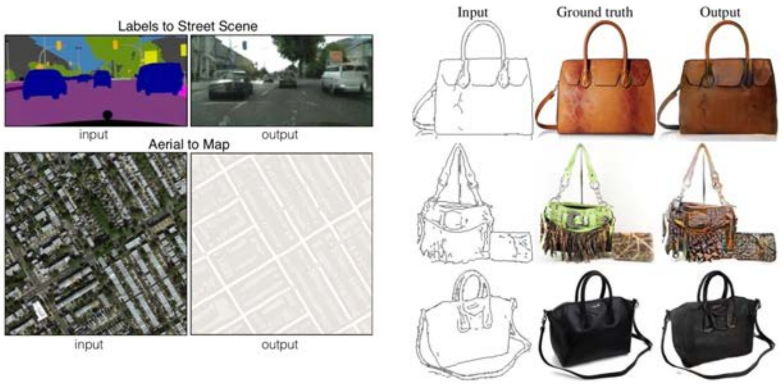
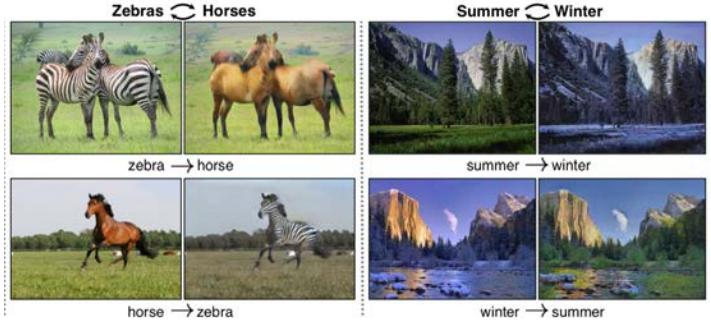
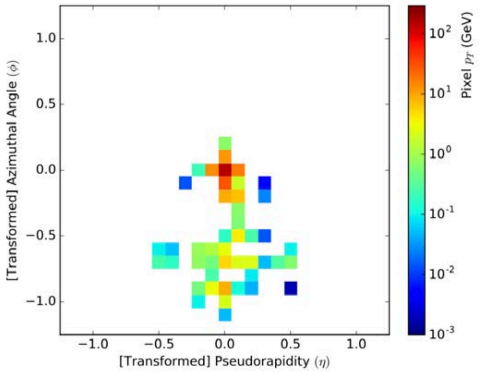


Image-to-Image Translation with Conditional Adversarial Networks, *Isola et al.*, 2016



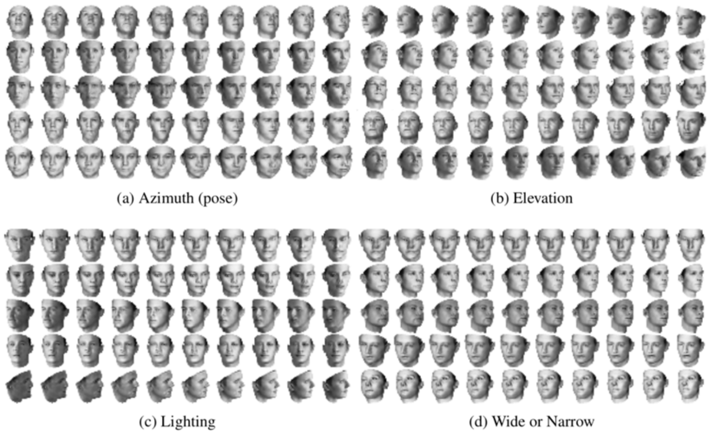
Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, *Zhu et al.*, 2017

experimental simulation



Learning Particle Physics by Example, *de Oliveira et al.*, 2017

interpretable representations



(a) Azimuth (pose)

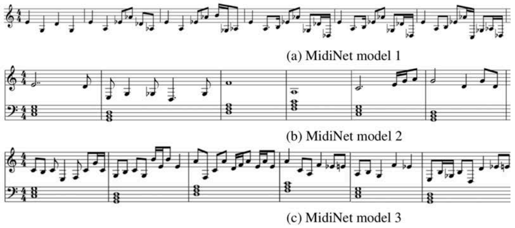
(b) Elevation

(c) Lighting

(d) Wide or Narrow

InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets, *Chen et al.*, 2016

music synthesis



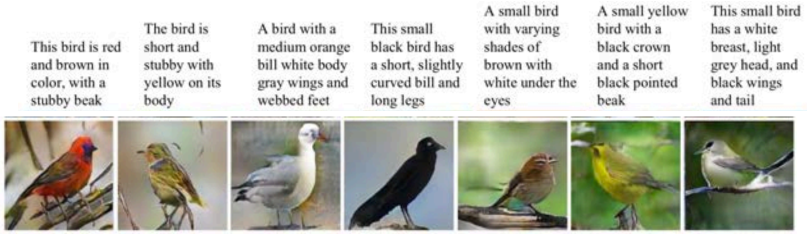
(a) MidiNet model 1

(b) MidiNet model 2

(c) MidiNet model 3

MIDINET: A CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORK FOR SYMBOLIC-DOMAIN MUSIC GENERATION, *Yang et al.*, 2017

text to image synthesis



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

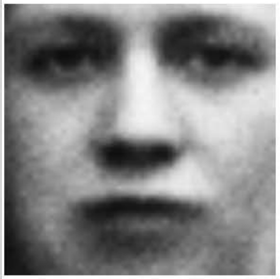
This small black bird has a short, slightly curved bill and long legs

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

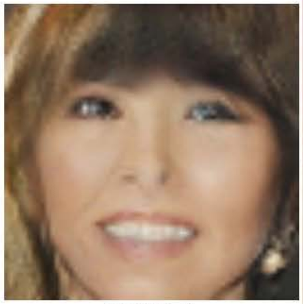
A small yellow bird with a black crown and a short black pointed beak

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

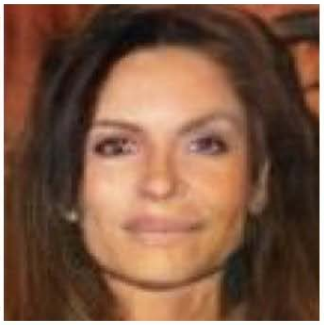
StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks, *Zhang et al.*, 2016



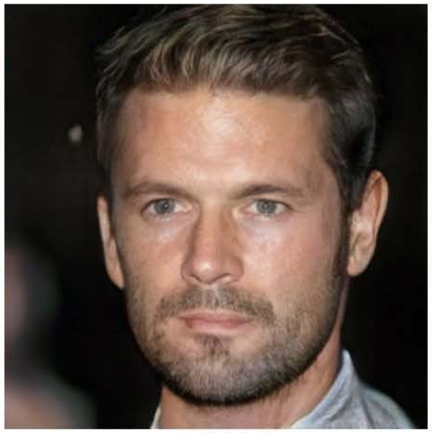
2014



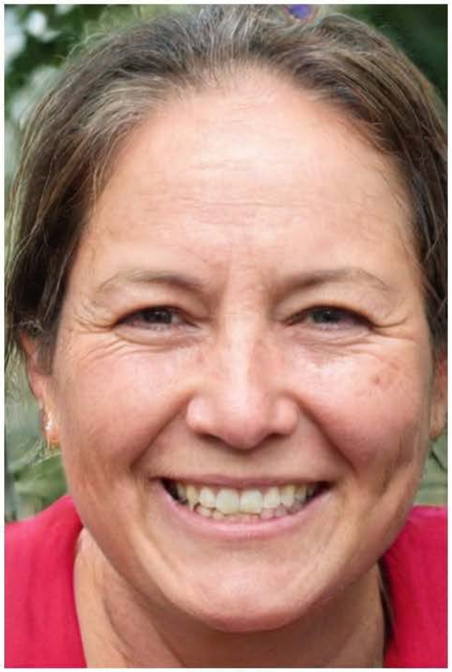
2015



2016



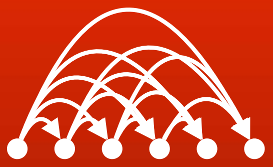
2017



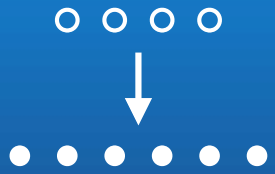
2018

arxiv.org/abs/1406.2661
arxiv.org/abs/1511.06434
arxiv.org/abs/1606.07536
arxiv.org/abs/1710.10196
arxiv.org/abs/1812.04948

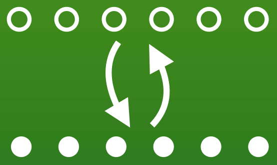
DISCUSSION



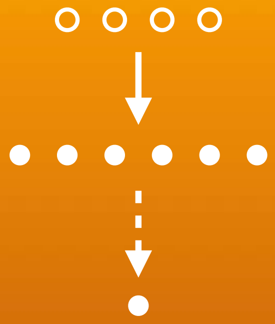
autoregressive models



explicit latent variable models



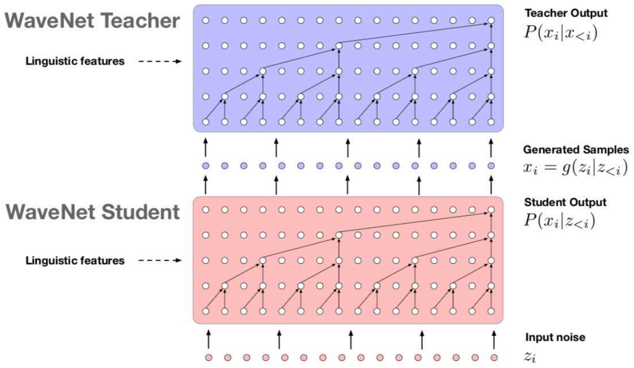
invertible explicit latent variable models



implicit latent variable models

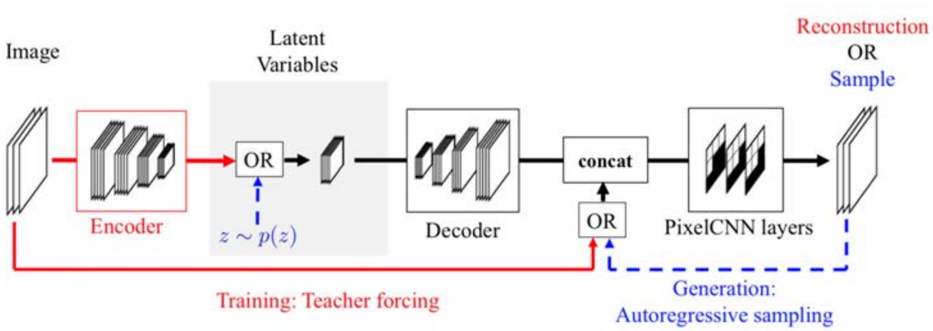
combining models

autoregressive + invertible model



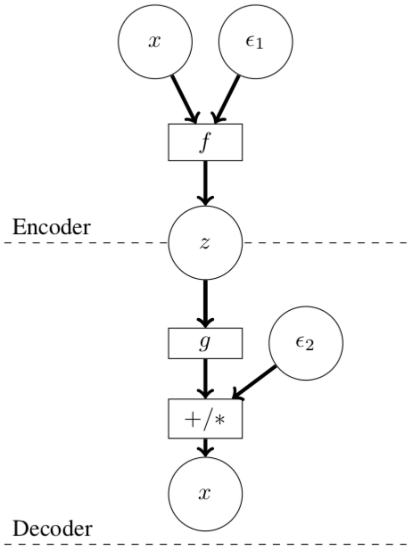
Parallel WaveNet, van den Oord *et al.*, 2018

autoregressive + explicit latent variable model



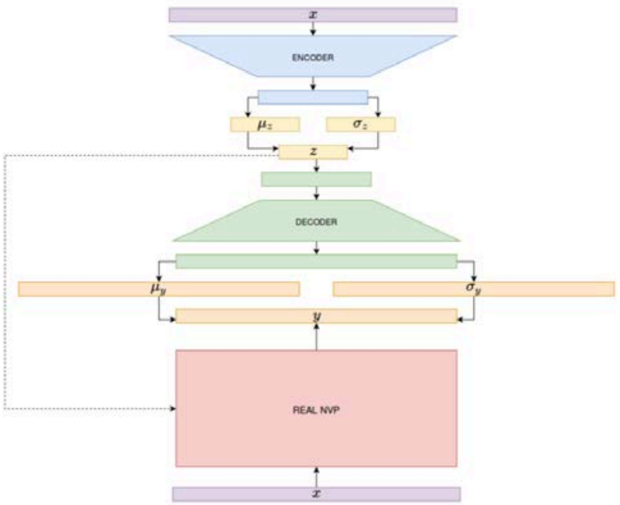
PixelVAE, Gulrajani *et al.*, 2017

explicit + implicit latent variable model



Adversarial Variational Bayes, Mescheder *et al.*, 2017

explicit + invertible latent variable model

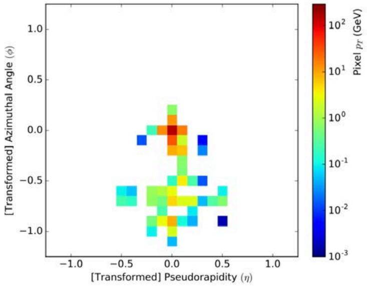


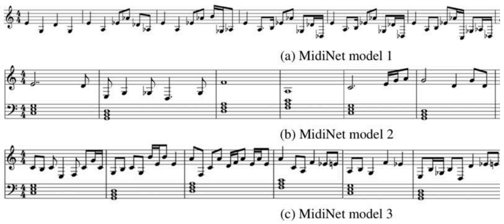
Deep Variational Inference Without Pixel-Wise Reconstruction,
Agrawal & Dukkipati, 2016

generative models: *what are they good for?*

generative models model the data distribution

1. can generate and simulate data



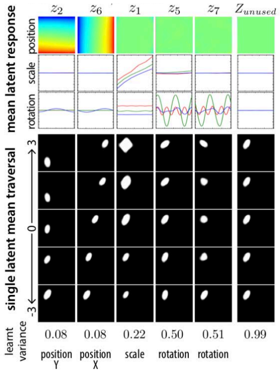


(a) MidiNet model 1

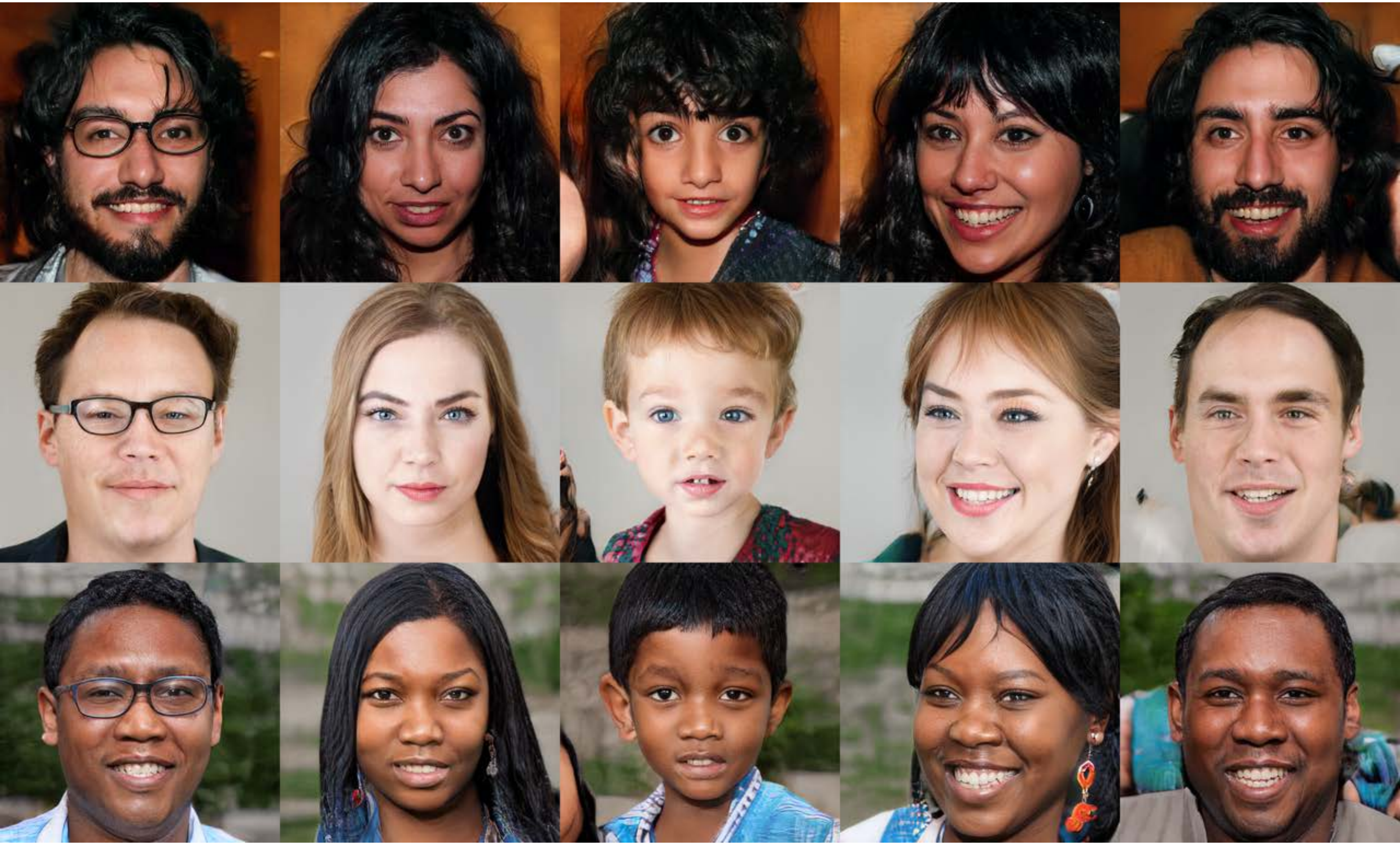
(b) MidiNet model 2

(c) MidiNet model 3

2. can extract structure from data

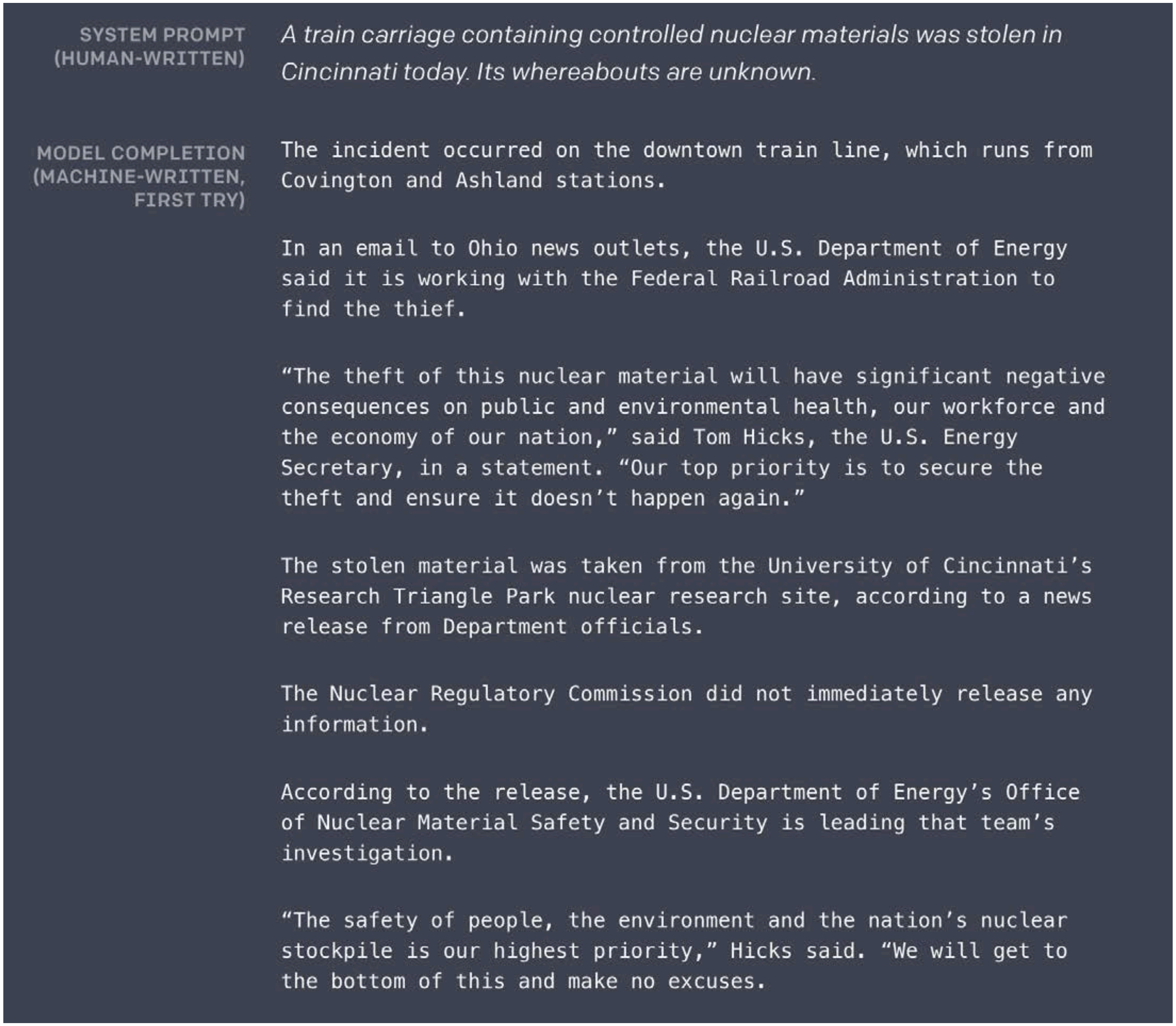


ethical concerns



StyleGAN, Karras *et al.*, 2018

ethical concerns



SYSTEM PROMPT (HUMAN-WRITTEN)

A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.

MODEL COMPLETION (MACHINE-WRITTEN, FIRST TRY)

The incident occurred on the downtown train line, which runs from Covington and Ashland stations.

In an email to Ohio news outlets, the U.S. Department of Energy said it is working with the Federal Railroad Administration to find the thief.

“The theft of this nuclear material will have significant negative consequences on public and environmental health, our workforce and the economy of our nation,” said Tom Hicks, the U.S. Energy Secretary, in a statement. “Our top priority is to secure the theft and ensure it doesn’t happen again.”

The stolen material was taken from the University of Cincinnati’s Research Triangle Park nuclear research site, according to a news release from Department officials.

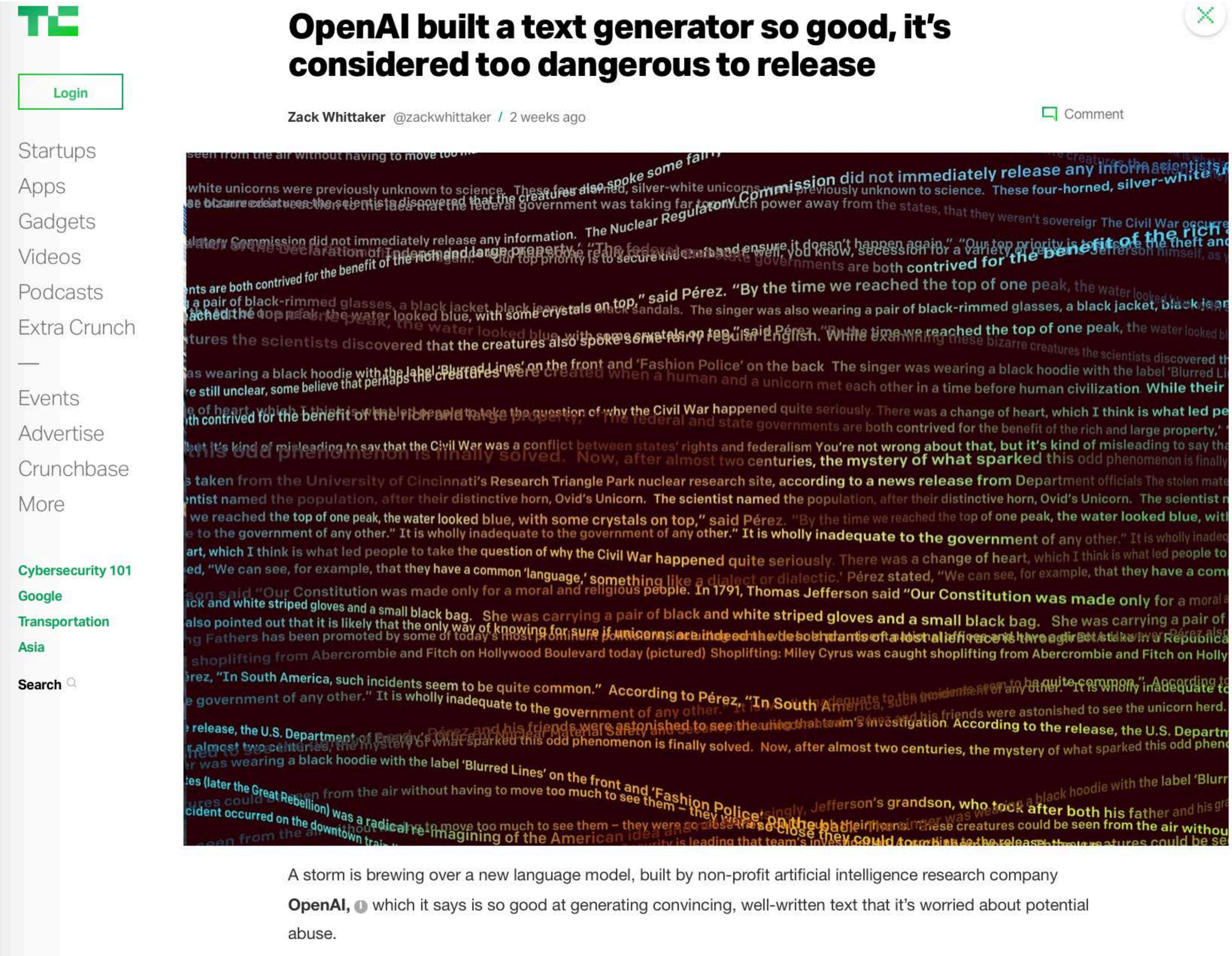
The Nuclear Regulatory Commission did not immediately release any information.

According to the release, the U.S. Department of Energy’s Office of Nuclear Material Safety and Security is leading that team’s investigation.

“The safety of people, the environment and the nation’s nuclear stockpile is our highest priority,” Hicks said. “We will get to the bottom of this and make no excuses.

Language Models as Unsupervised Multi-task Learners, Radford *et al.*, 2019

ethical concerns



Login

Startups
Apps
Gadgets
Videos
Podcasts
Extra Crunch
—
Events
Advertise
Crunchbase
More

Cybersecurity 101
Google
Transportation
Asia

Search

OpenAI built a text generator so good, it's considered too dangerous to release

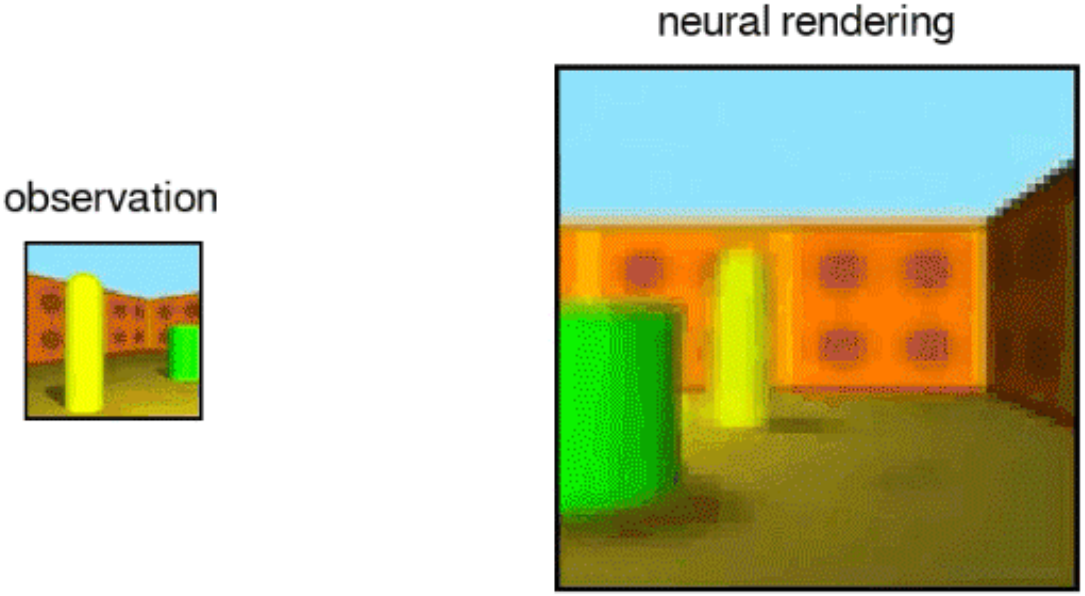
Zack Whittaker @zackwhittaker / 2 weeks ago

Comment

A storm is brewing over a new language model, built by non-profit artificial intelligence research company **OpenAI,** which it says is so good at generating convincing, well-written text that it's worried about potential abuse.

applying generative models to new forms of data

model-based RL: using a (generative) model to plan actions



GQN, Eslami *et al.*, 2018