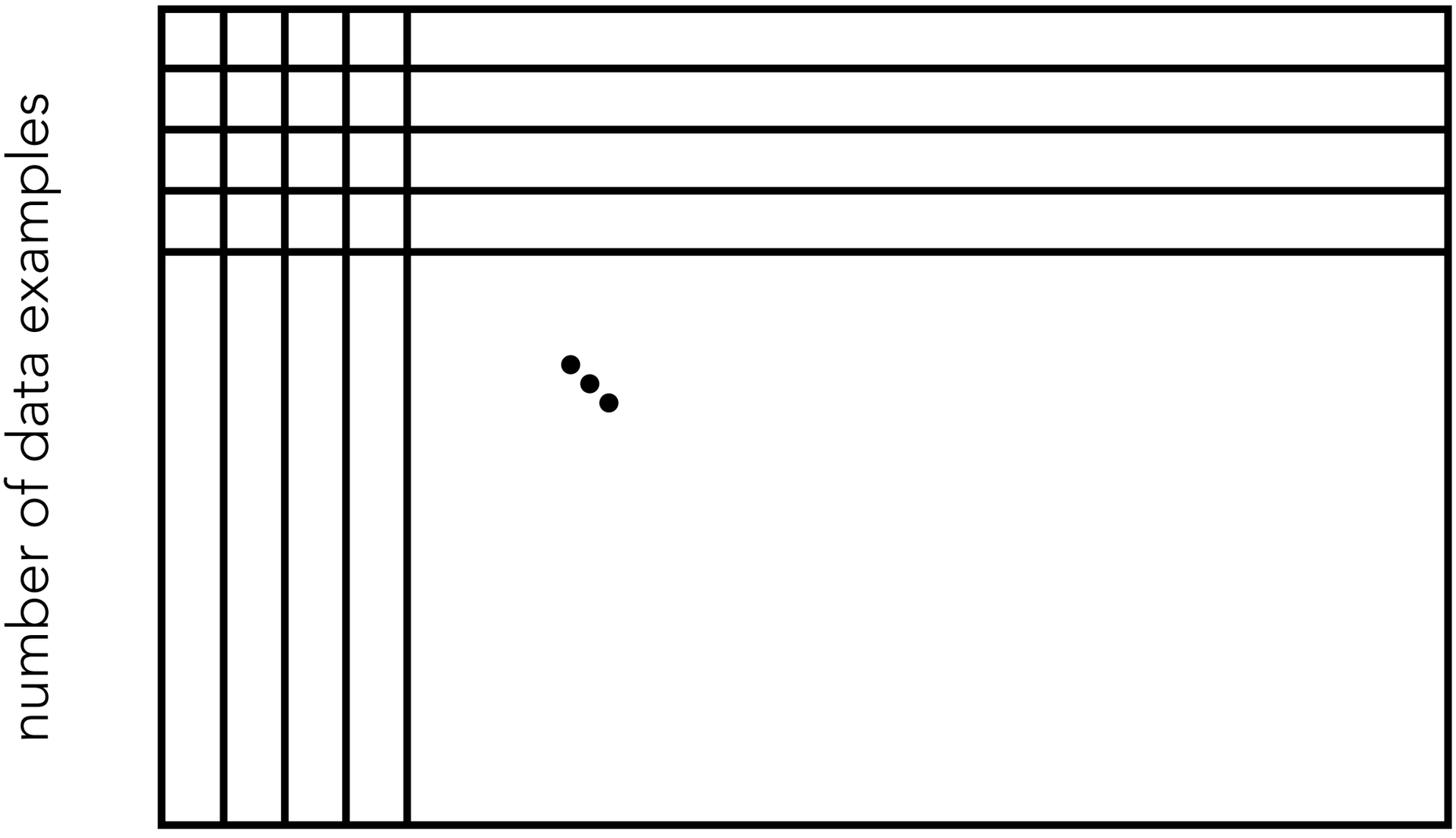
DEEP GENERATIVE MODELS

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

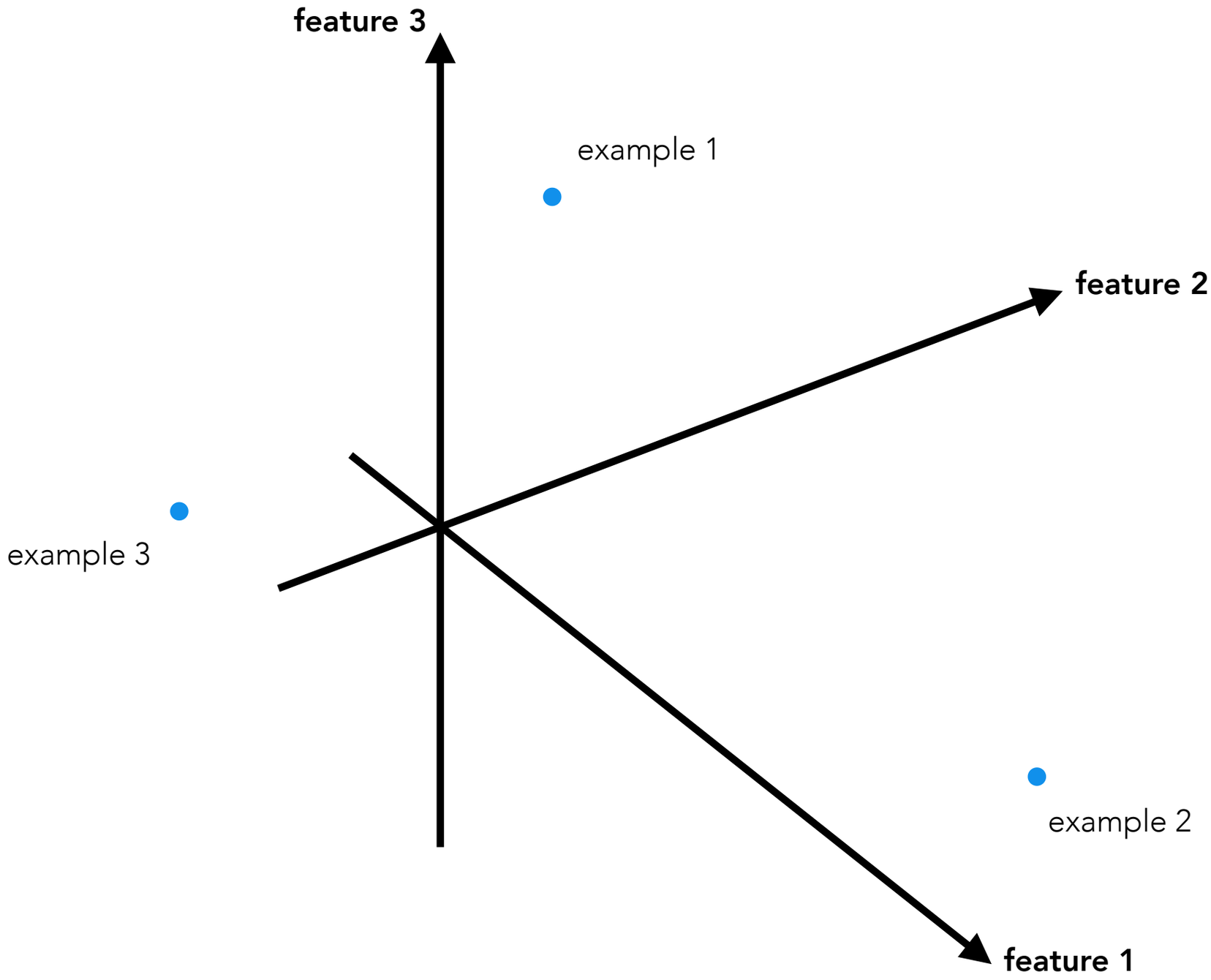
GENERATIVE MODELS

number of features

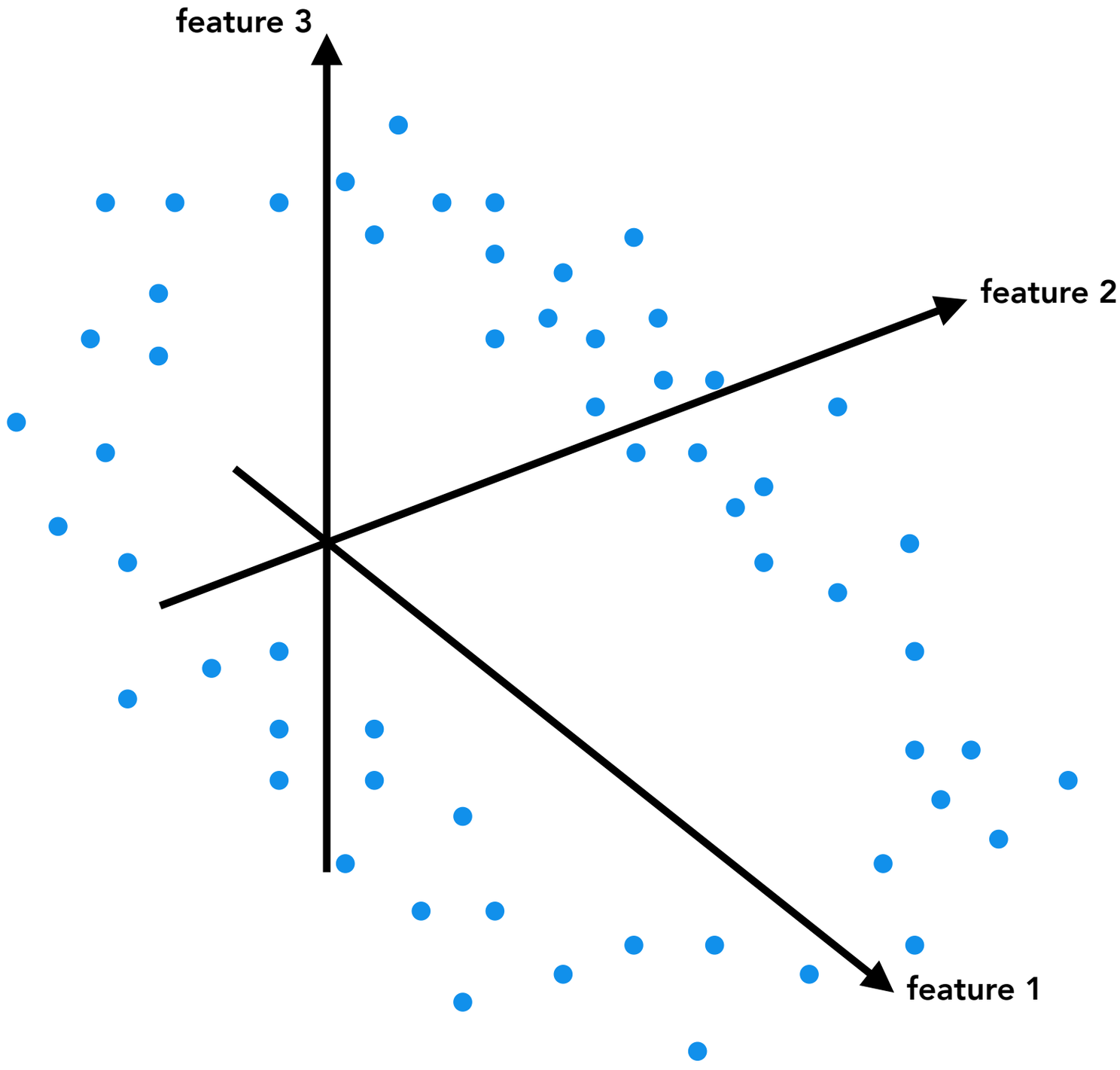


number of data examples

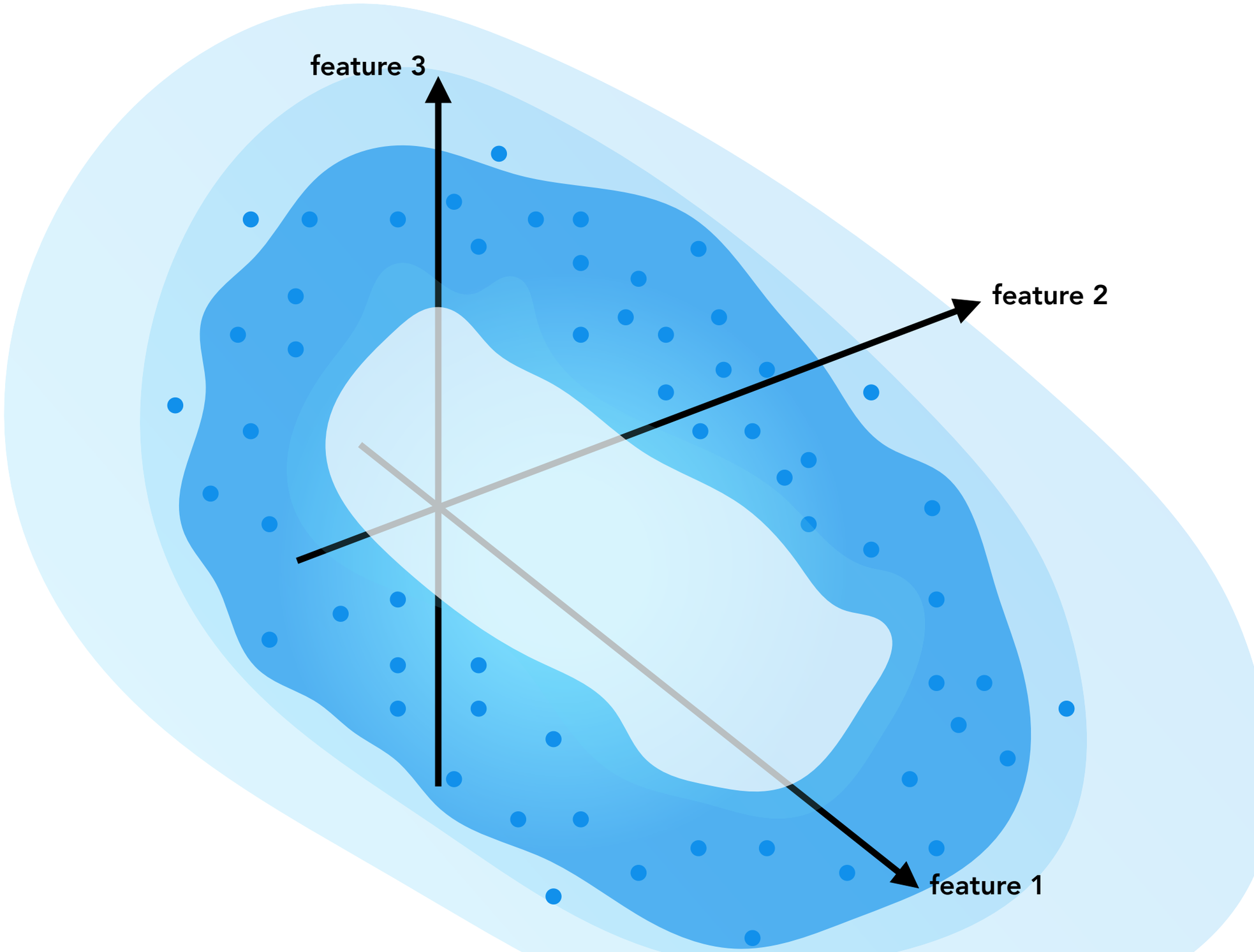
DATA



DATA DISTRIBUTION



EMPIRICAL DATA DISTRIBUTION



DENSITY ESTIMATION

estimating the density of the empirical data distribution

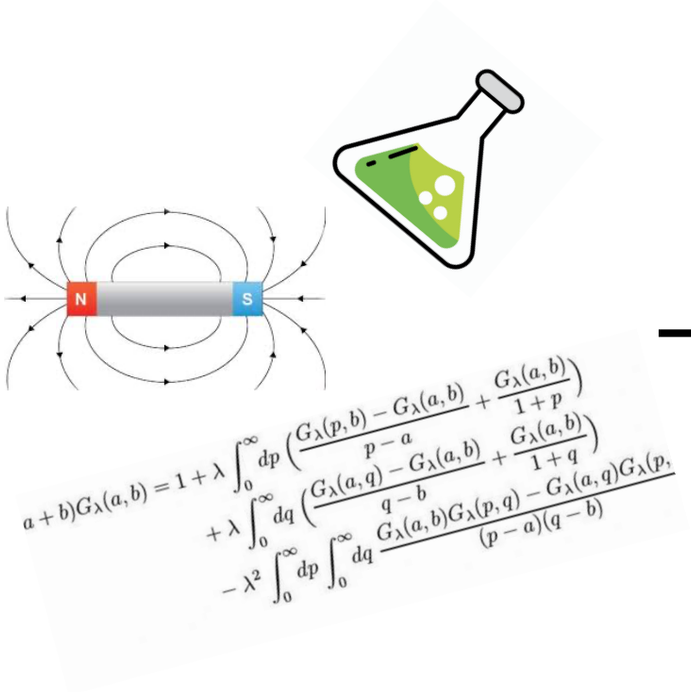
GENERATIVE MODEL

a model of the data distribution

note: *scientific theories are generative models*

data observations: measured physical phenomena

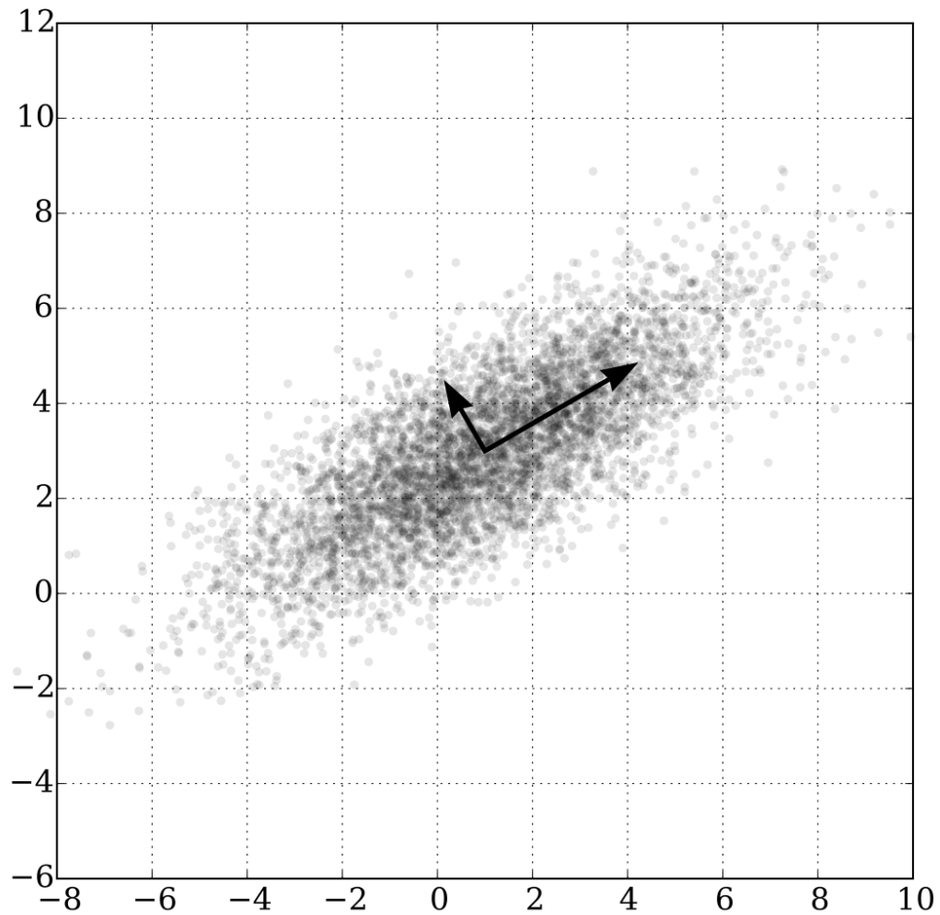
model: variables, functional relationships



observations

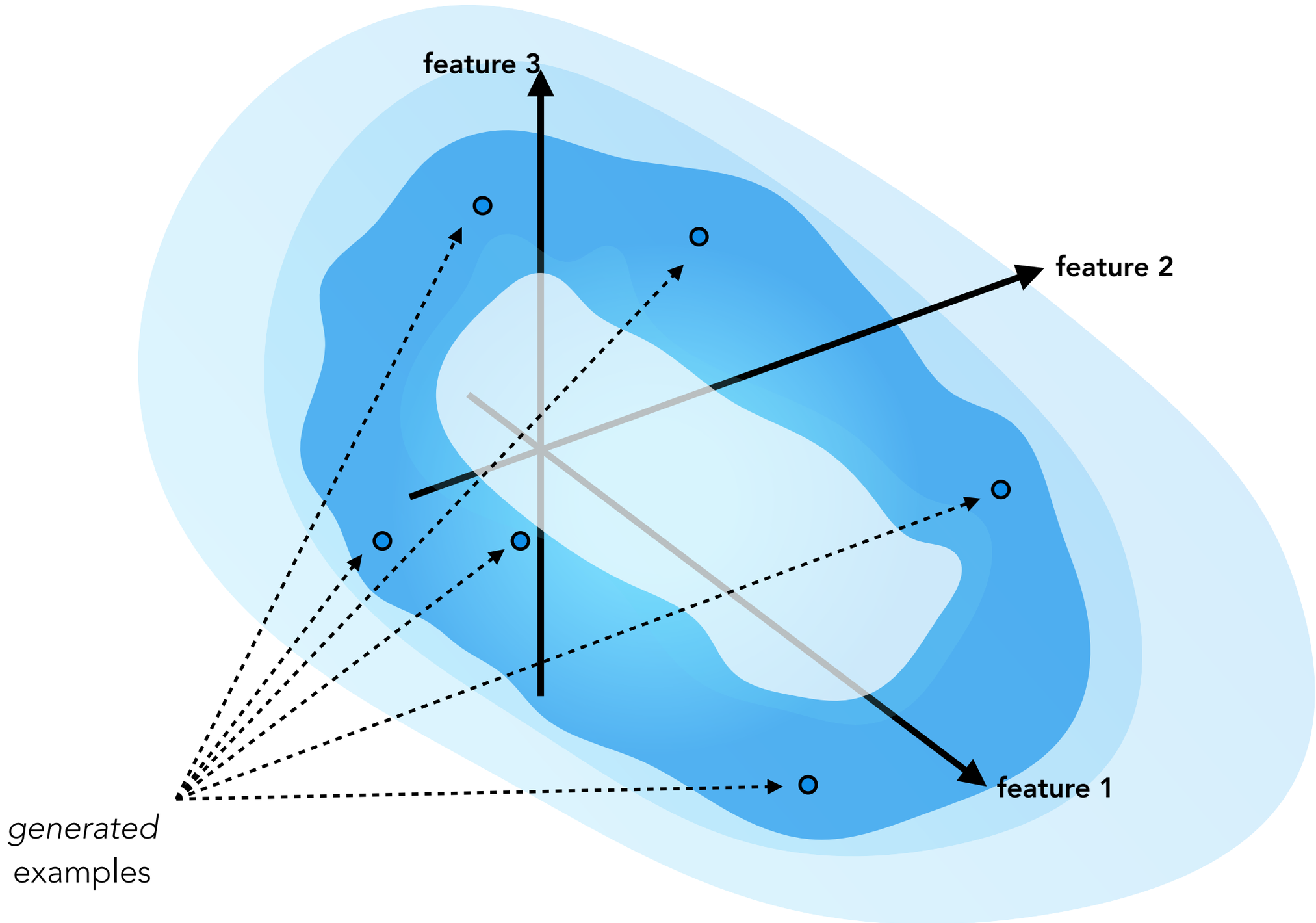
generative process

PCA: a linear-Gaussian generative model



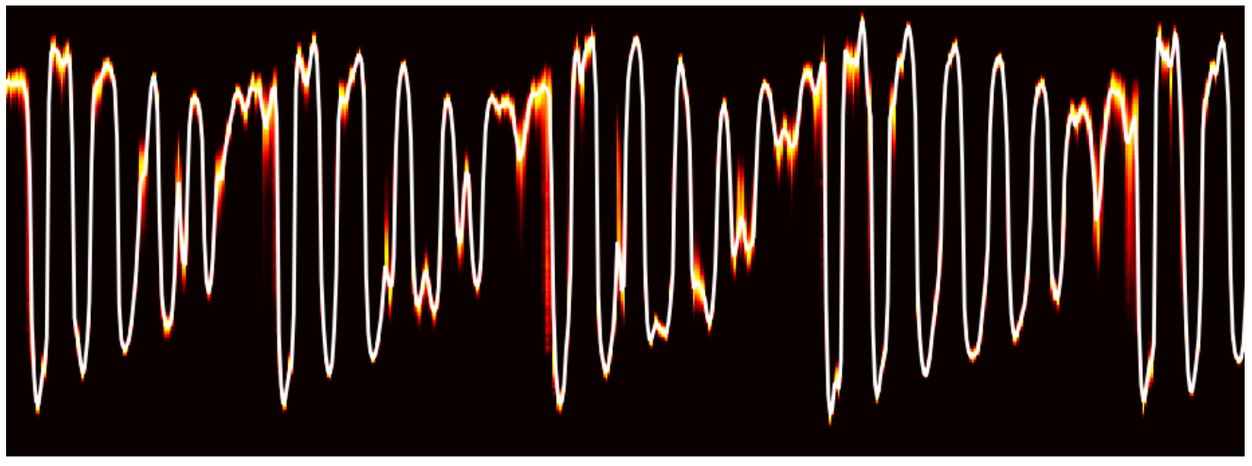
why learn a generative model?

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

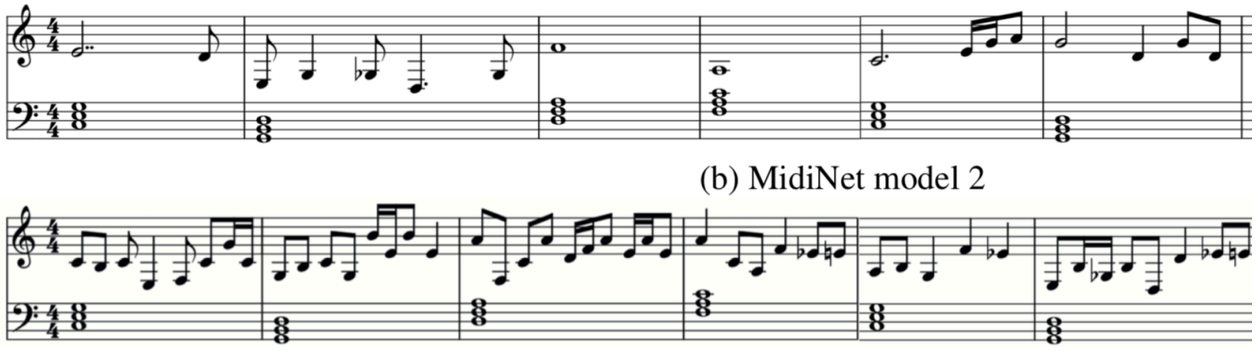


Glow, Kingma & Dhariwal, 2018

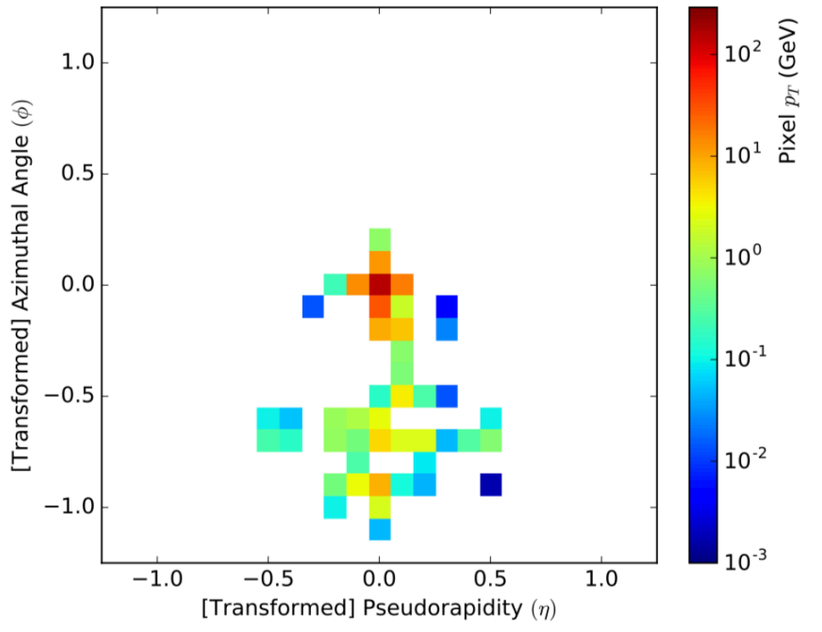
BigGan, Brock *et al.*, 2019



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

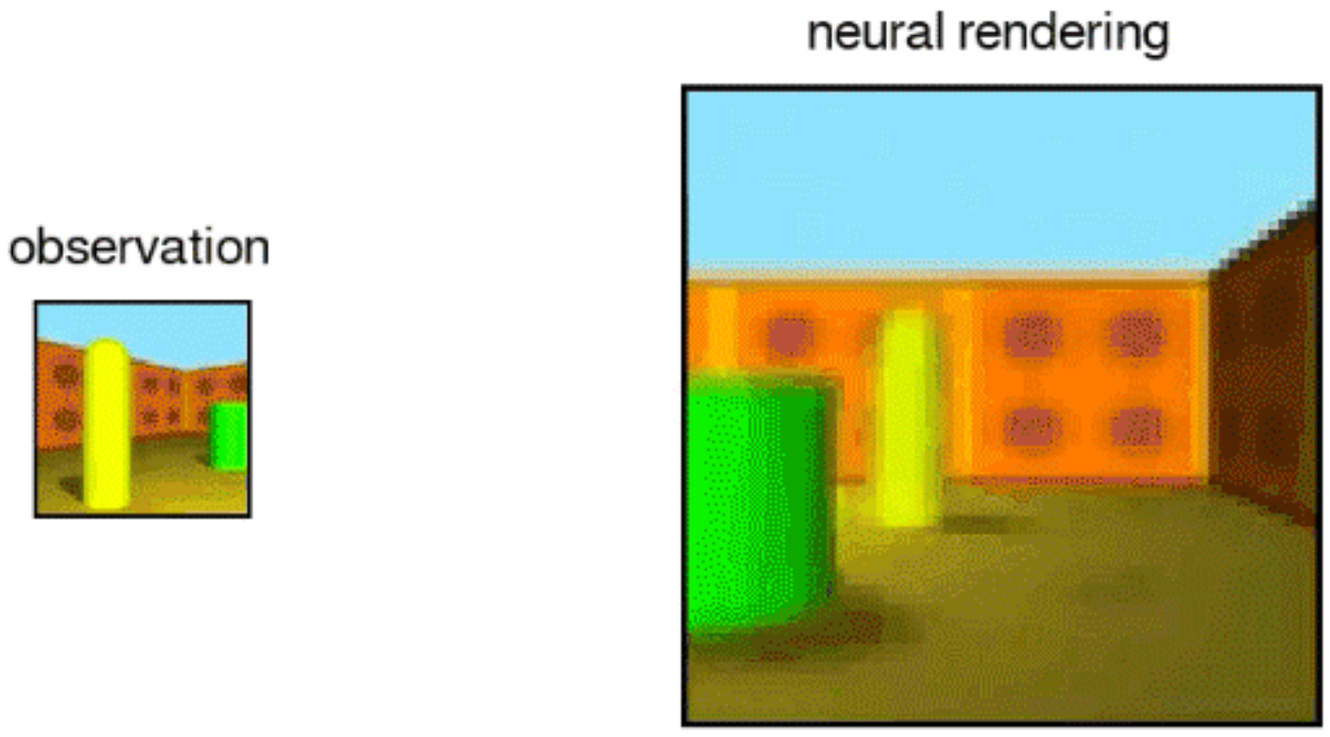


MidiNet, Yang *et al.*, 2017

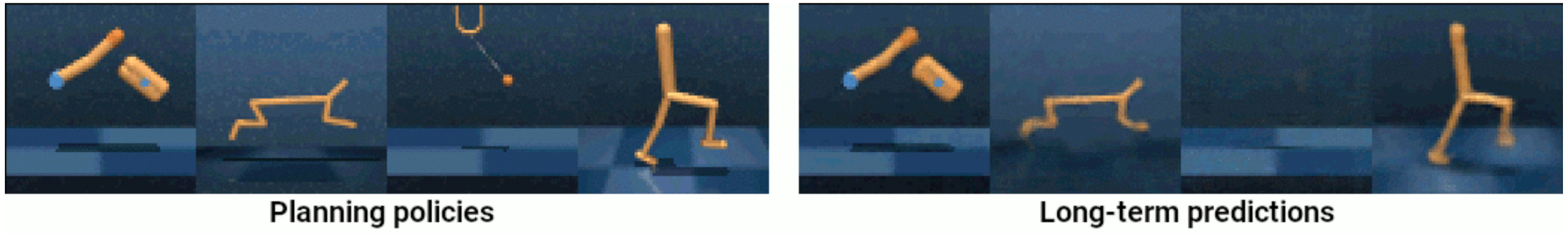


Learning Particle Physics by Example, *de Oliveira et al.*, 2017

(generative) model-based planning

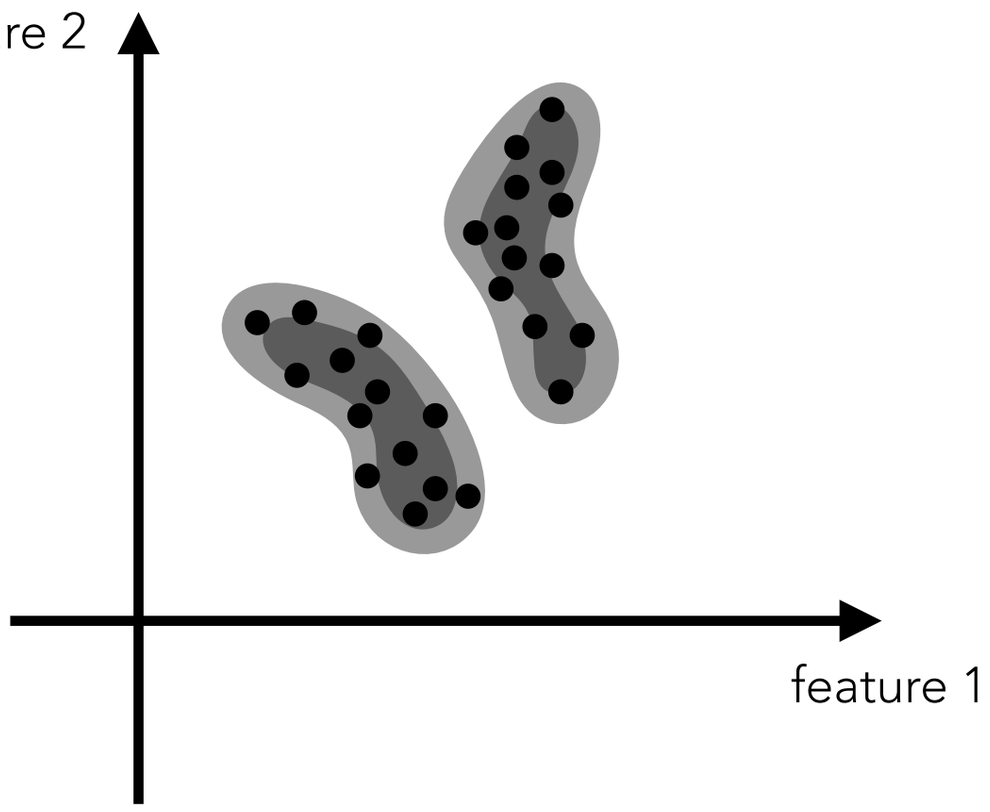


GQN, Eslami *et al.*, 2018

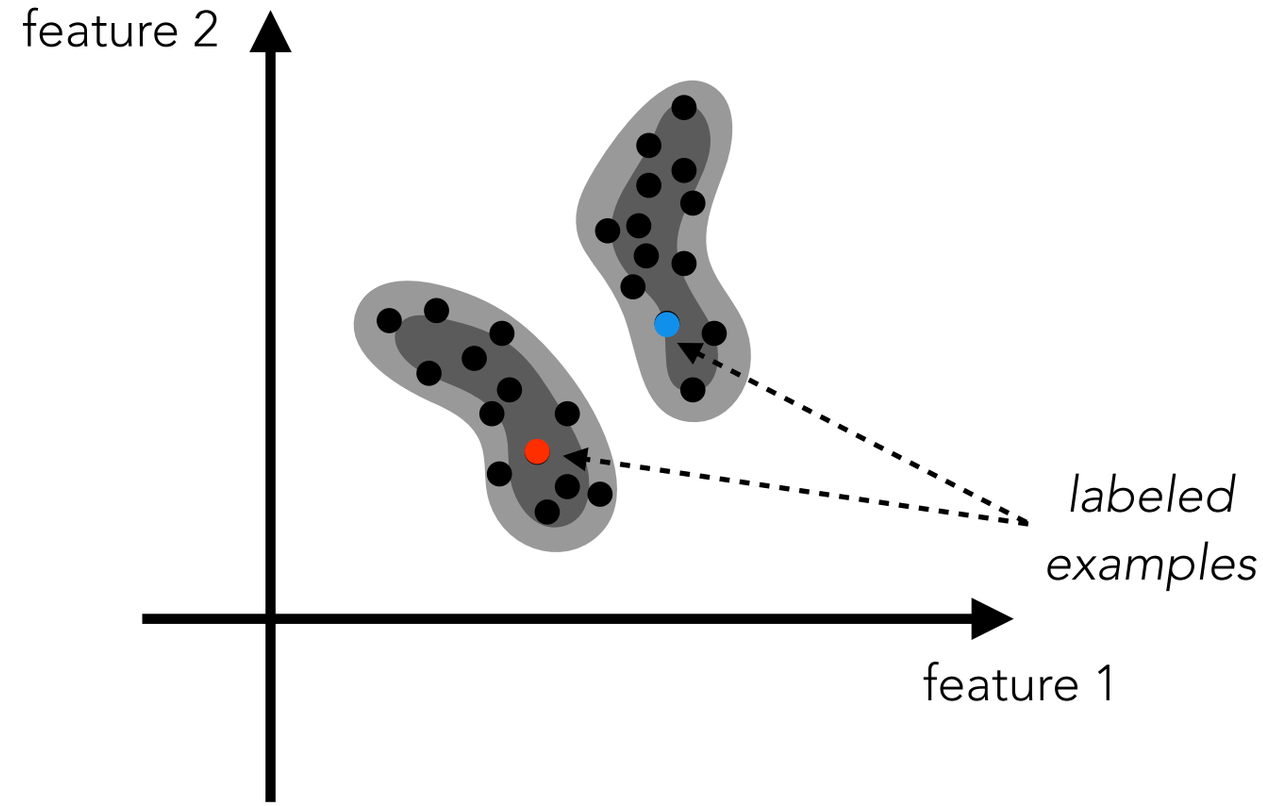


PlaNet, Hafner *et al.*, 2019

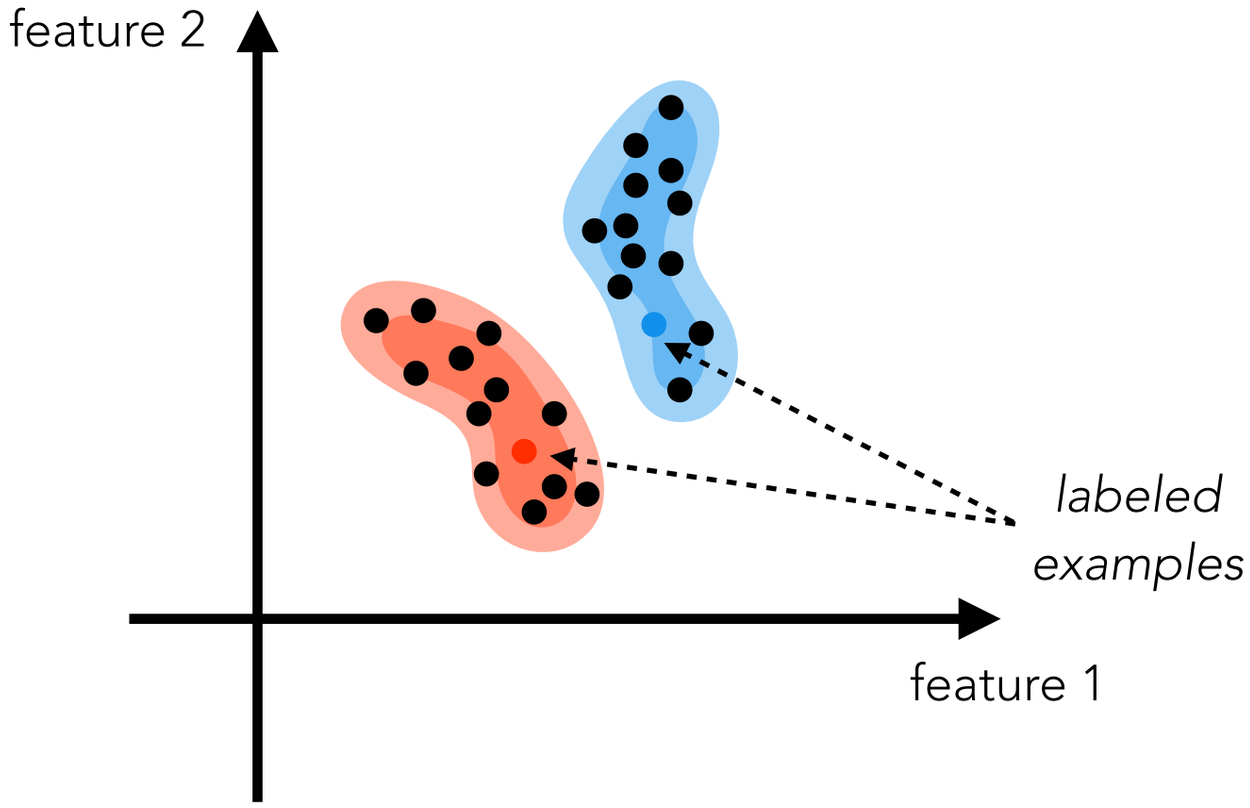
generative models can ***extract structure from data***



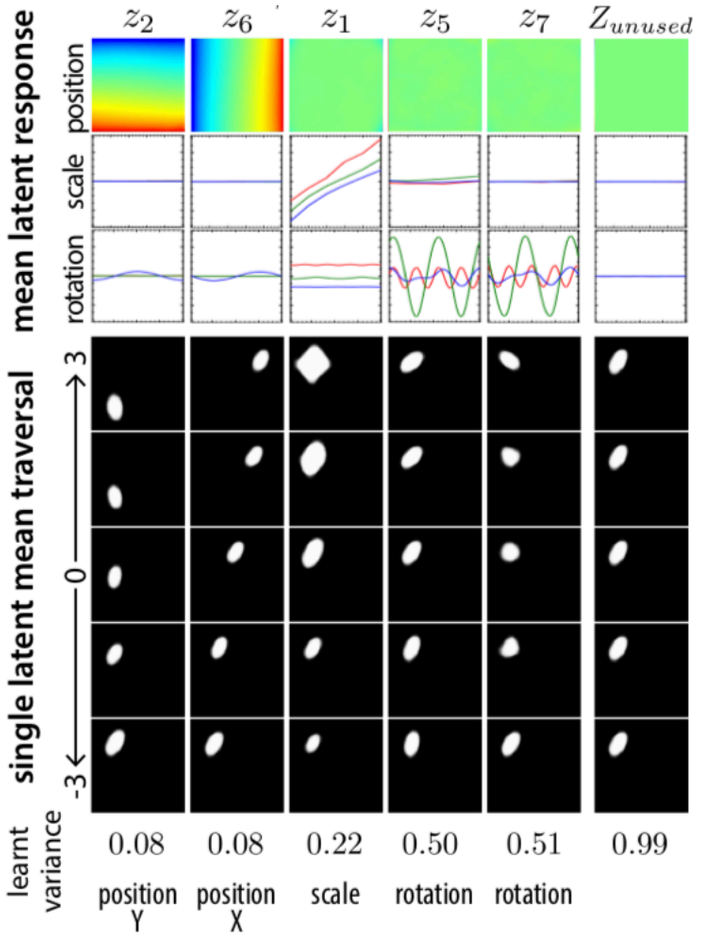
generative models can ***extract structure from data***



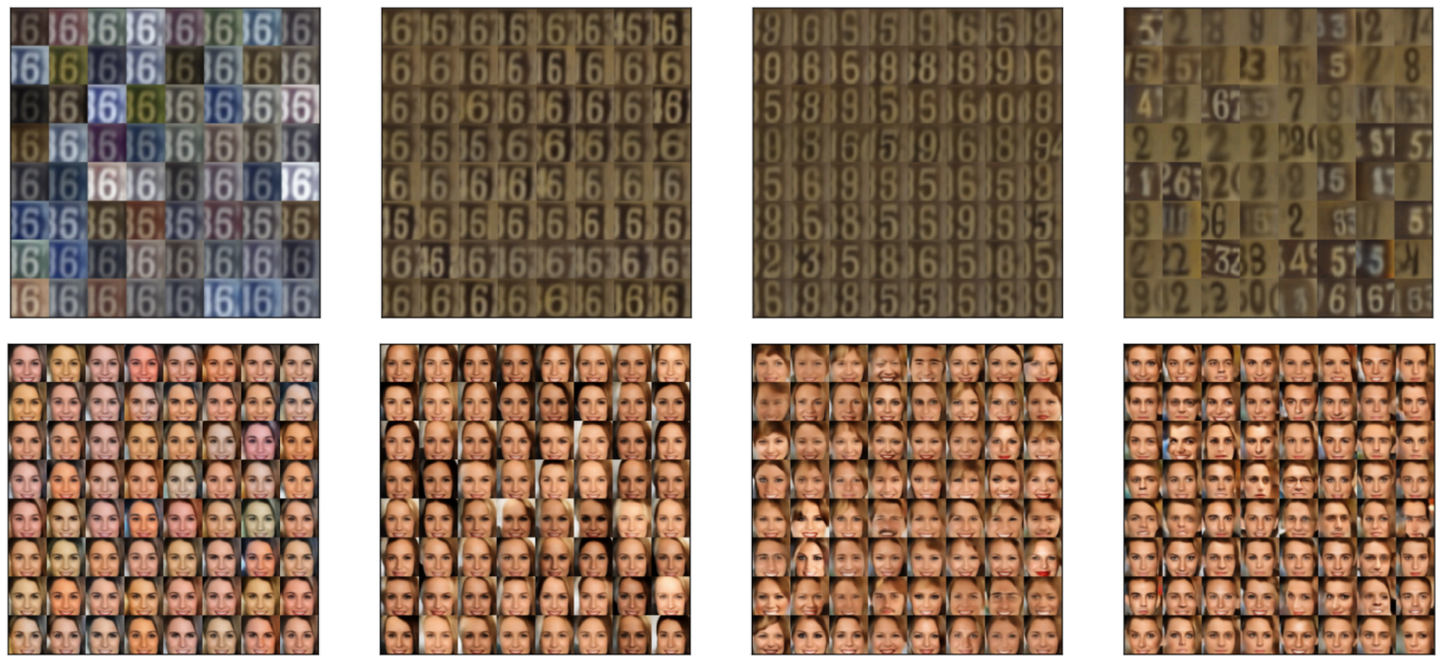
generative models can ***extract structure from data***



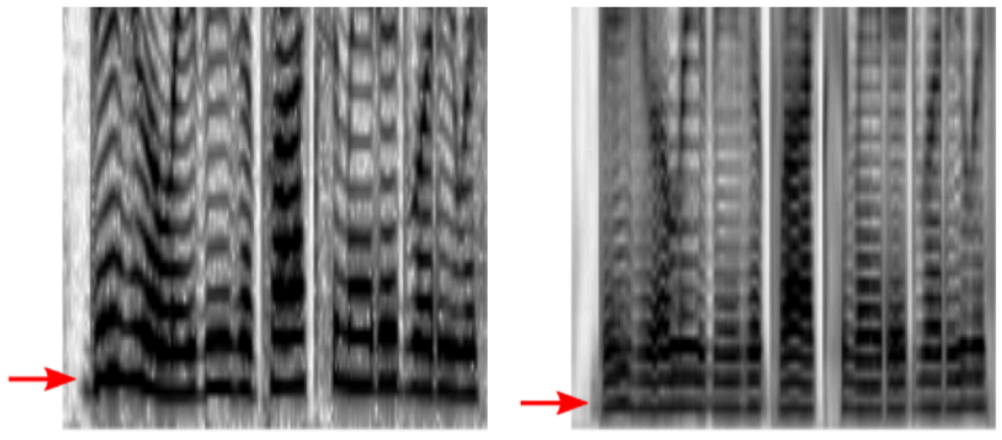
can make it easier to learn and generalize on new tasks



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

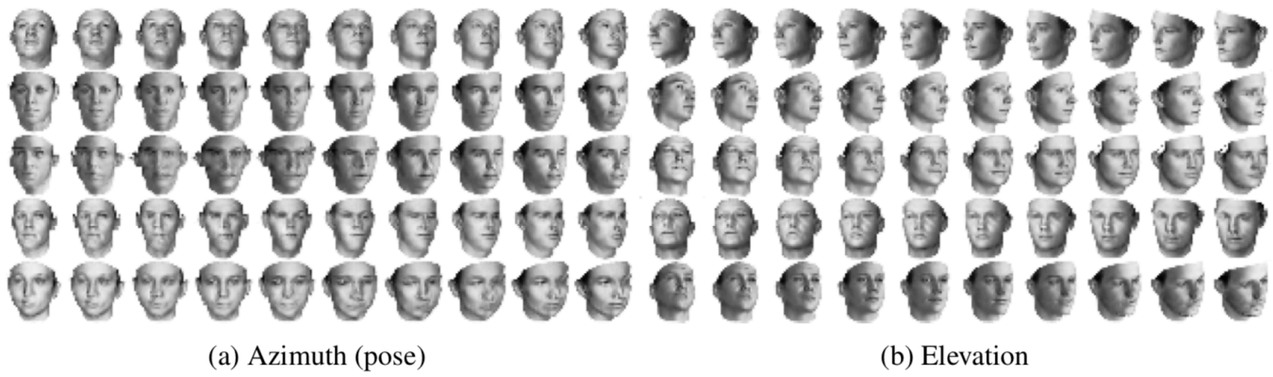


VLAE, Zhao *et al.*, 2017



(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

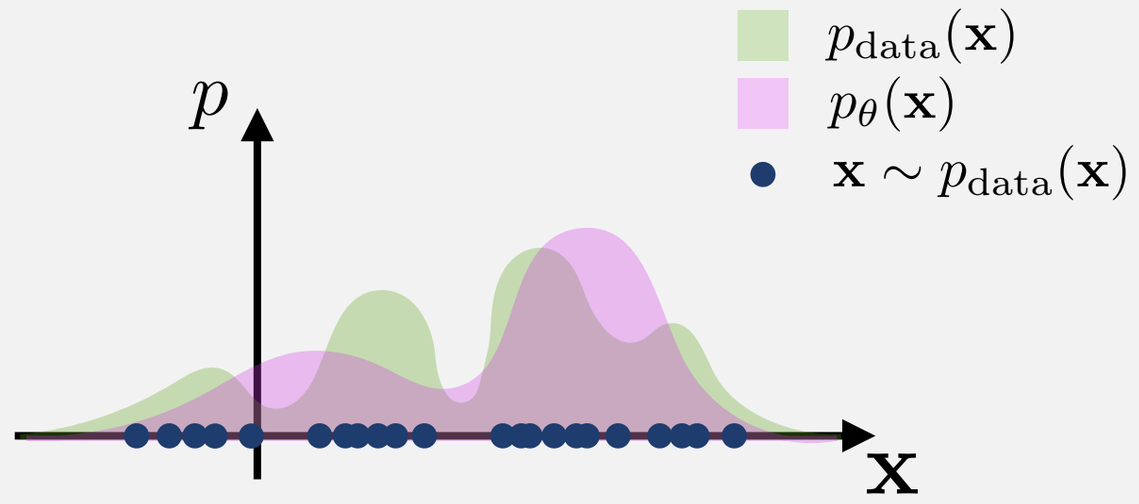
InfoGAN, Chen *et al.*, 2016

modeling the data distribution

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

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

parameters: θ

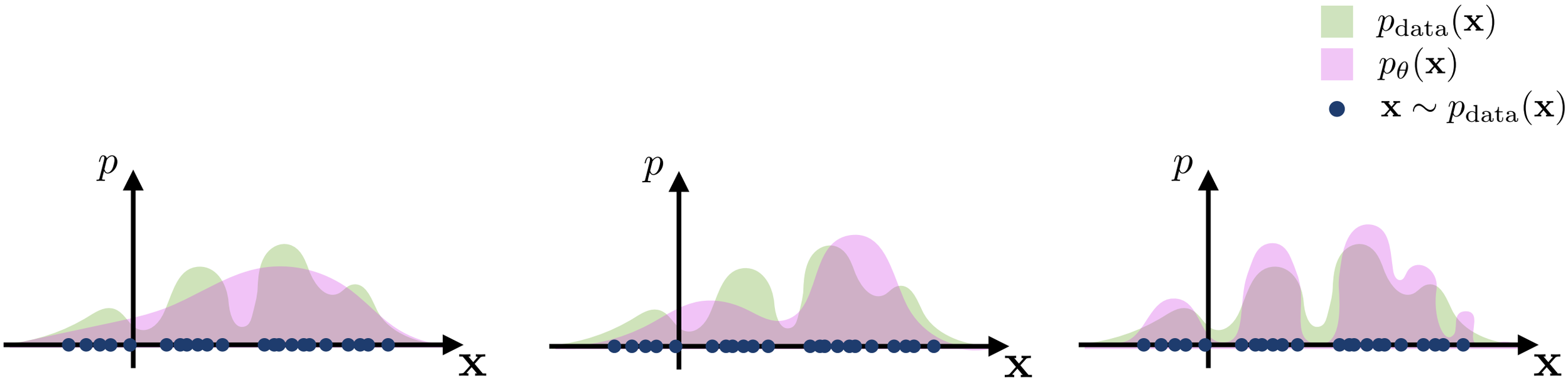


maximum likelihood estimation

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

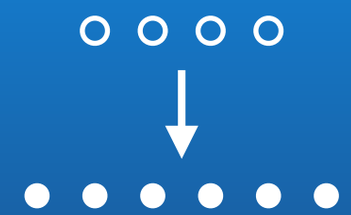
large variance

model complexity

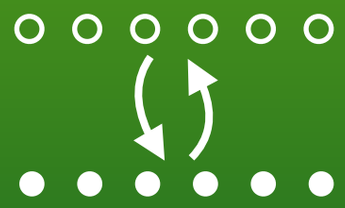
deep generative model

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

autoregressive models

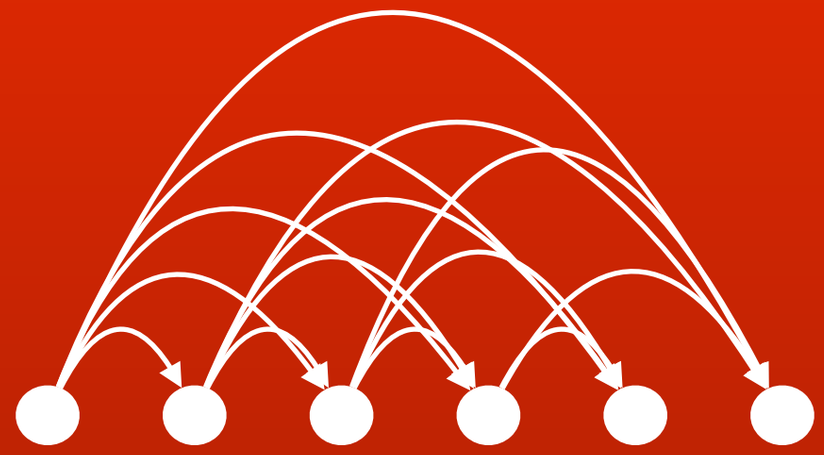


explicit latent variable models



invertible explicit latent variable models

energy-based models

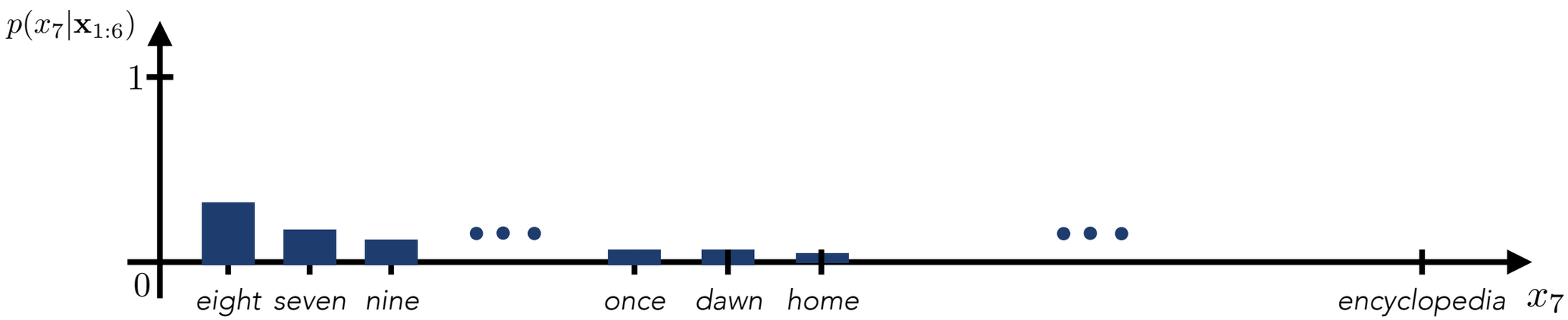


autoregressive models

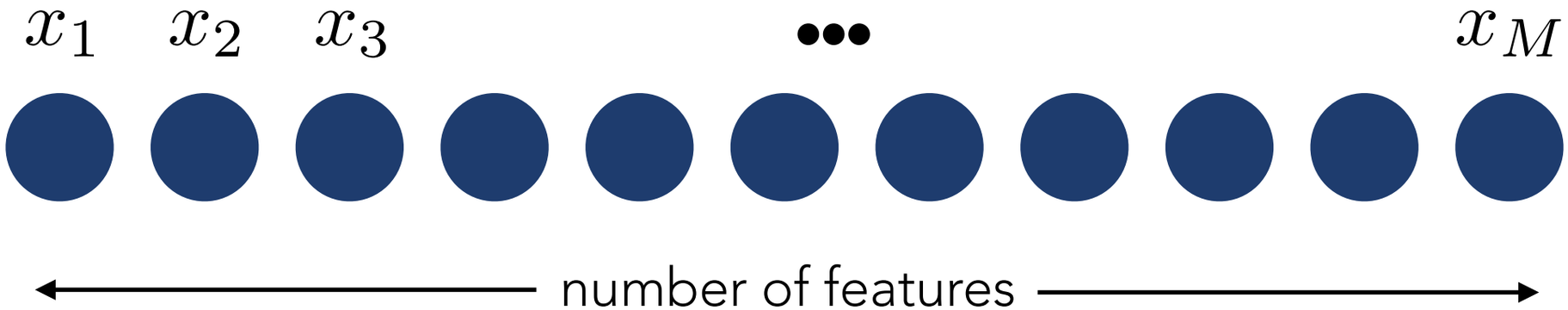
conditional probability distributions

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

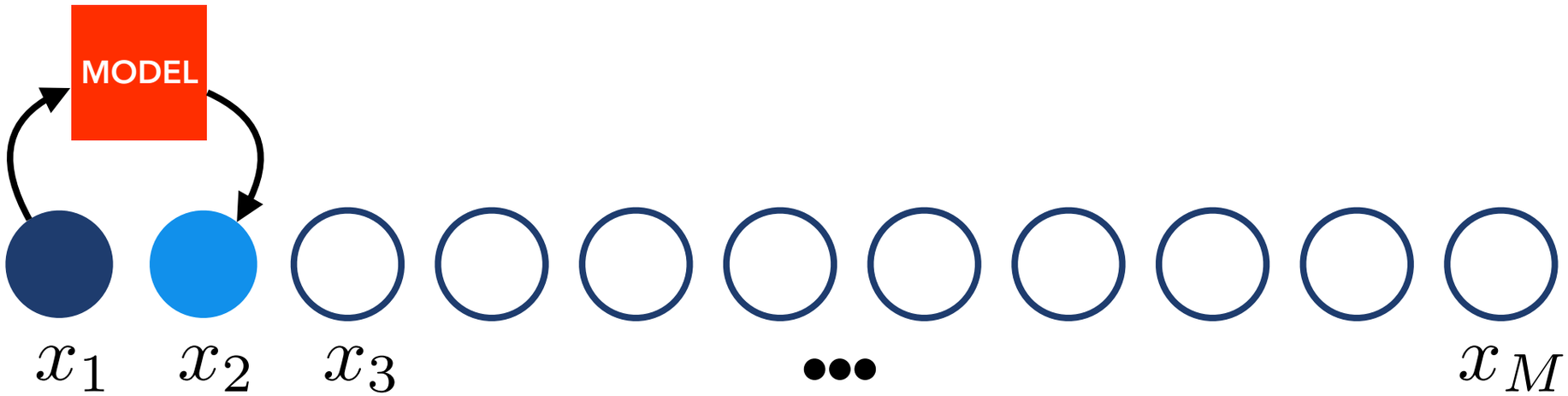
learn to ***auto-regress*** each value

$$p_\theta(x_1)$$

model the conditional distributions of the data

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

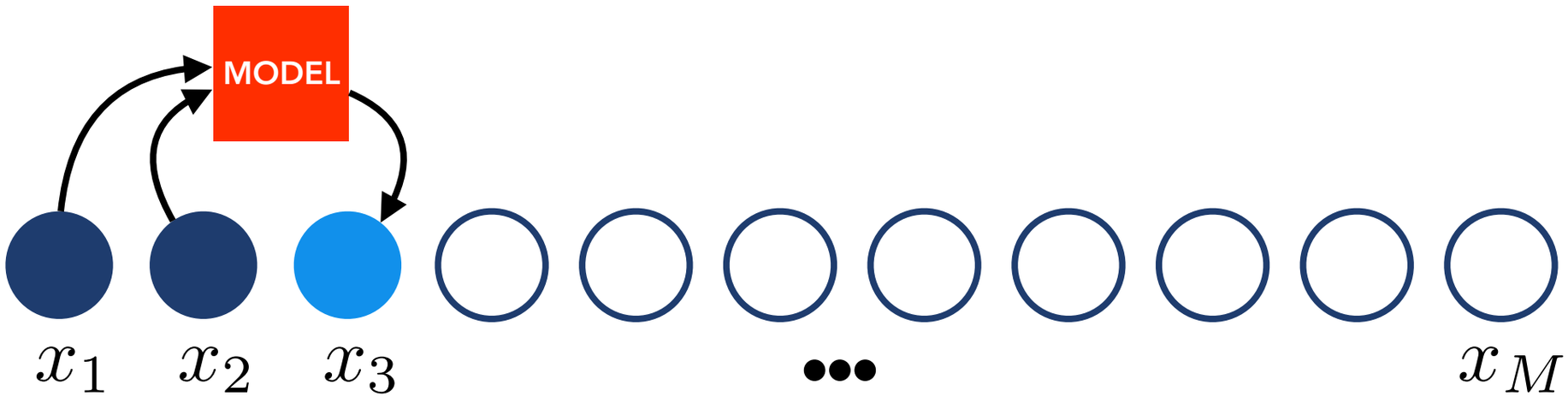
$$p_\theta(x_2|x_1)$$



model the conditional distributions of the data

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

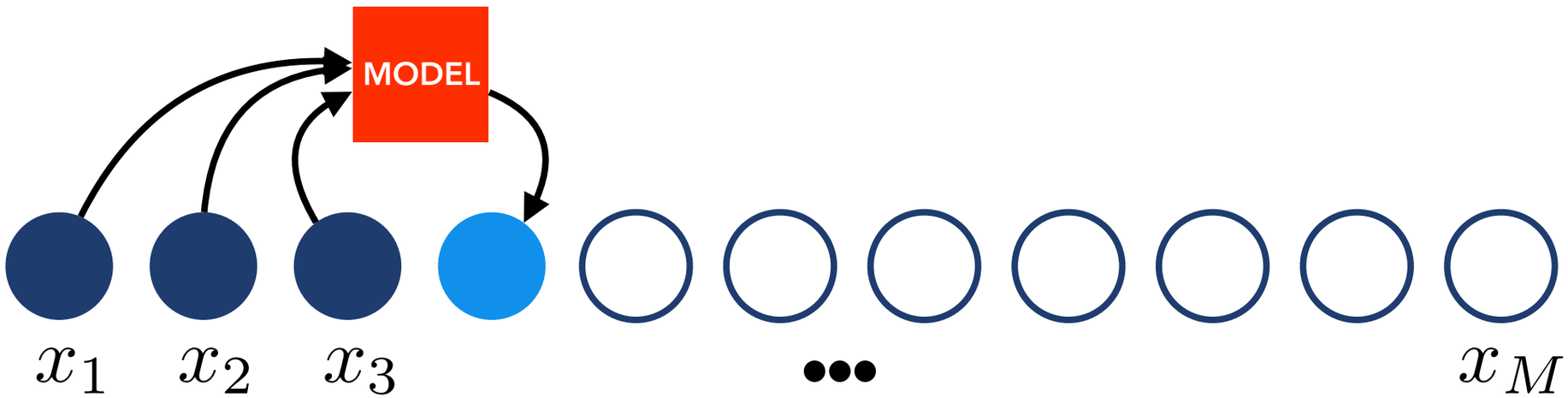
$$p_\theta(x_3|x_1, x_2)$$



model the conditional distributions of the data

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

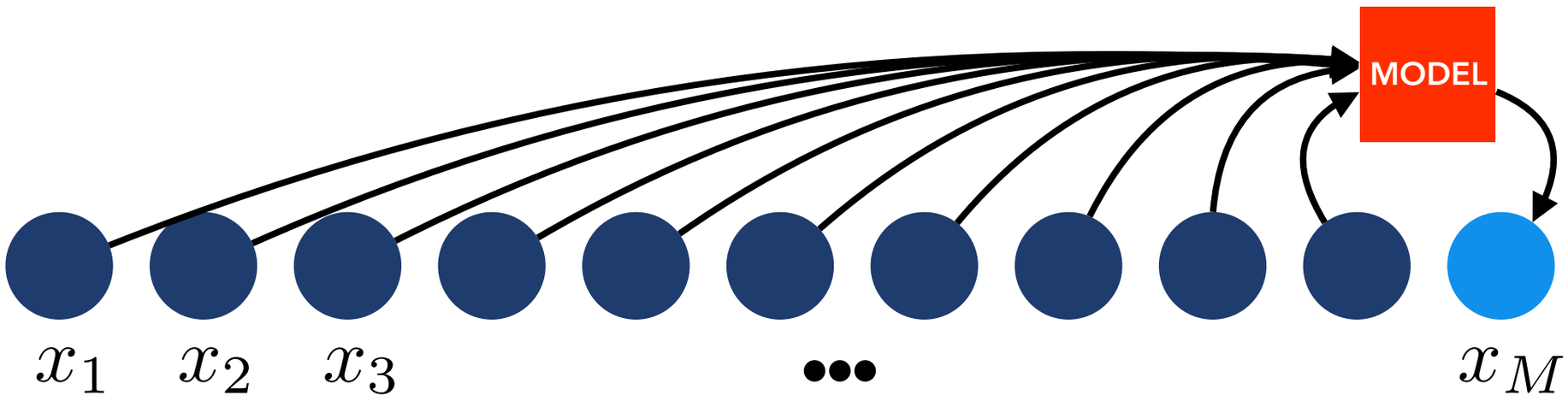
$$p_\theta(x_4|x_1, x_2, x_3)$$



model the conditional distributions of the data

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

$$p_\theta\left(x_M \middle| \mathbf{x}_{<M}\right)$$



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:

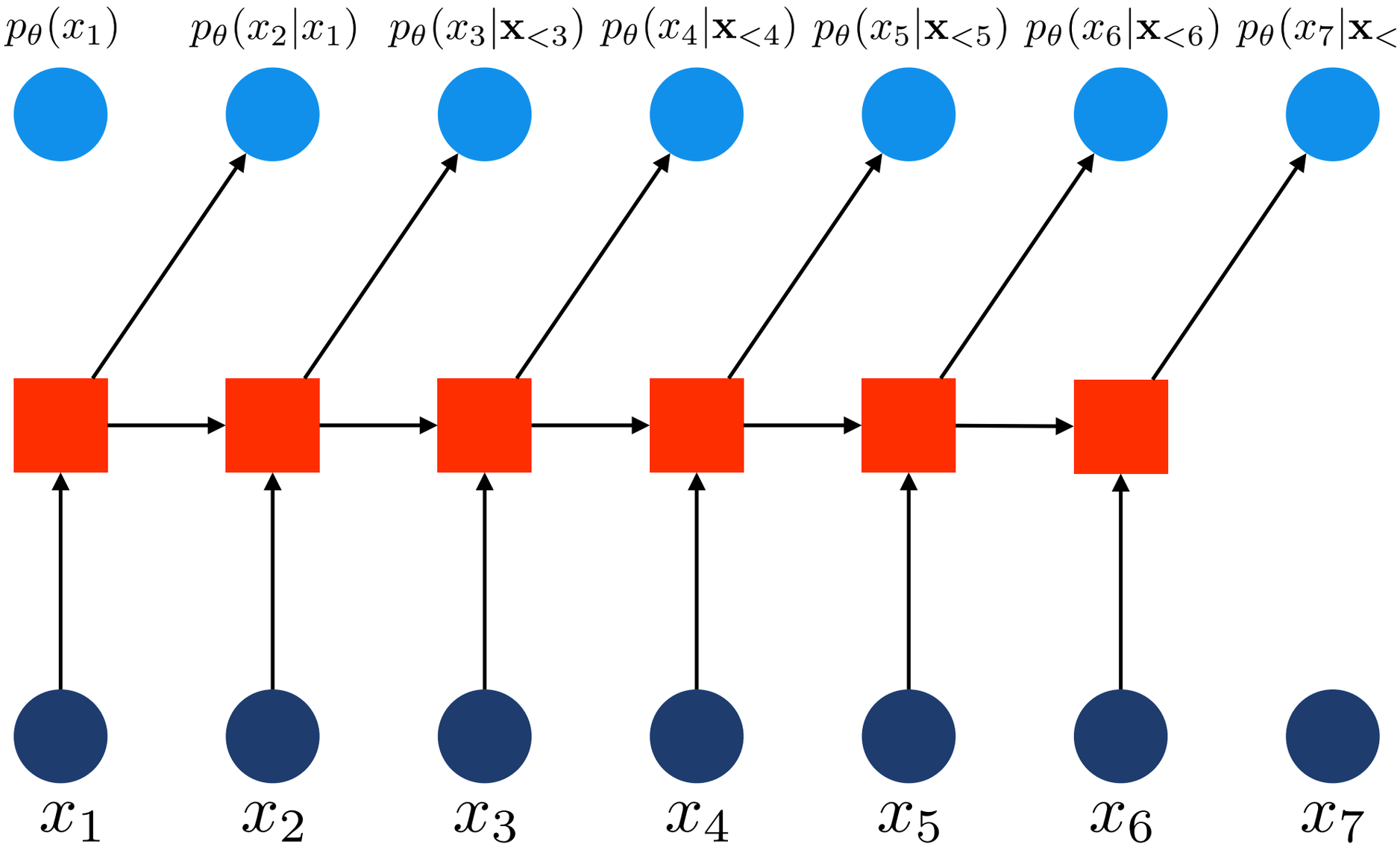
$$\log p_\theta(\mathbf{x}) = \log \left(\prod_{j=1}^{M} p_\theta(x_j | \mathbf{x}_{<j}) \right)$$

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

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

models

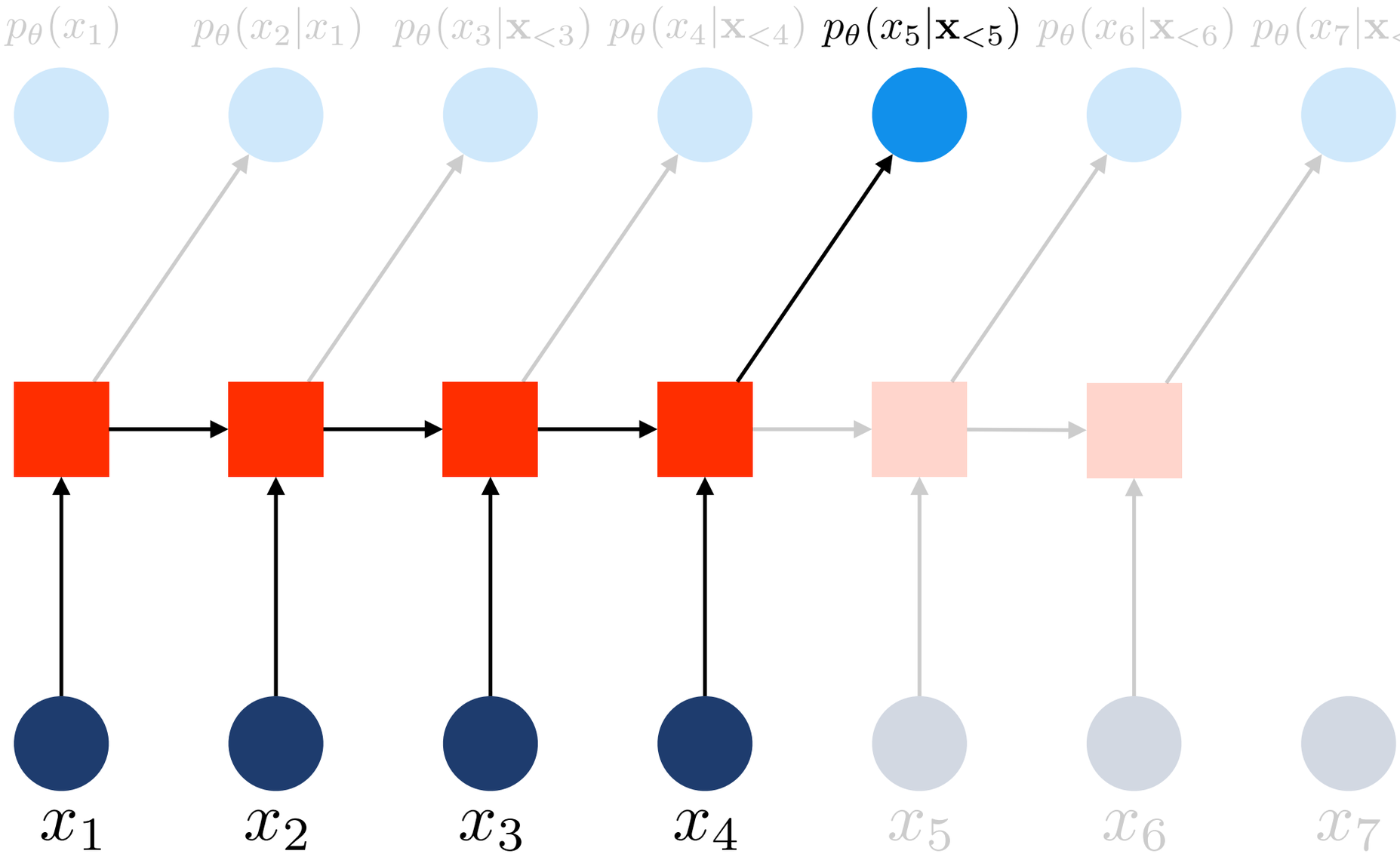
can parameterize conditional distributions using a **recurrent neural network**



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

models

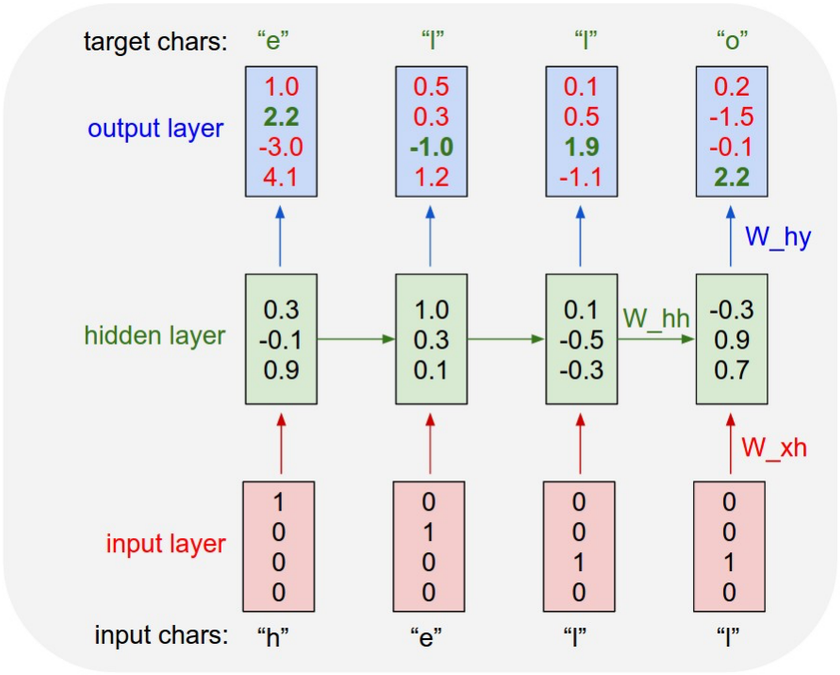
can parameterize conditional distributions using a **recurrent neural network**



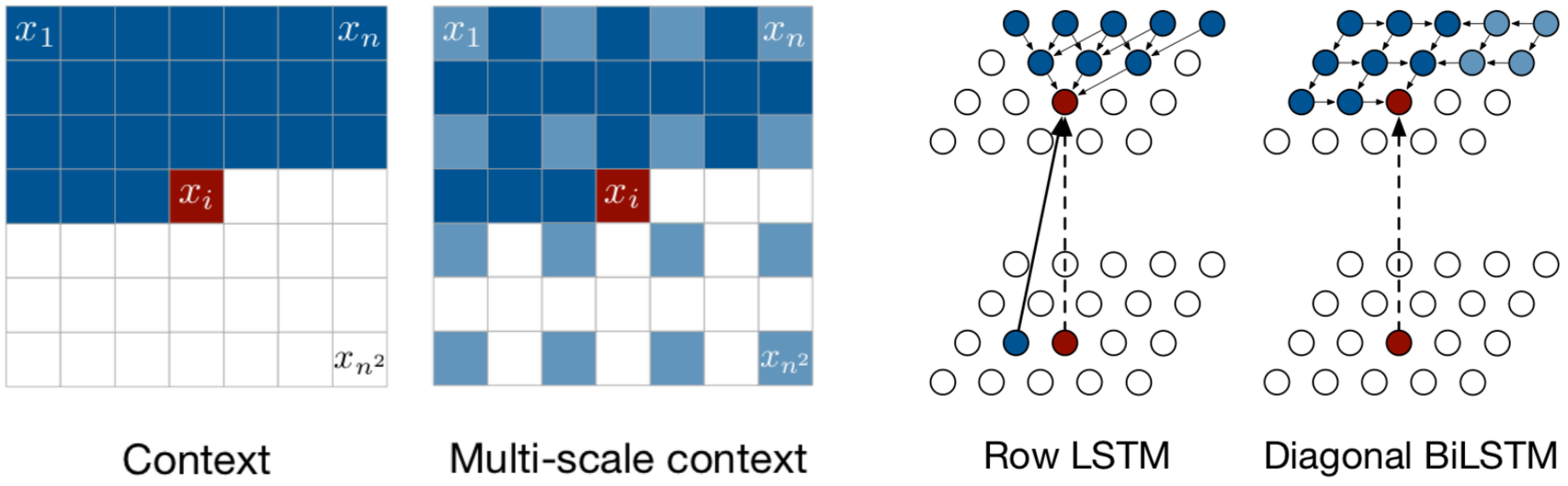
see **Deep Learning** (Chapter 10), *Goodfellow et al.*, 2016

models

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



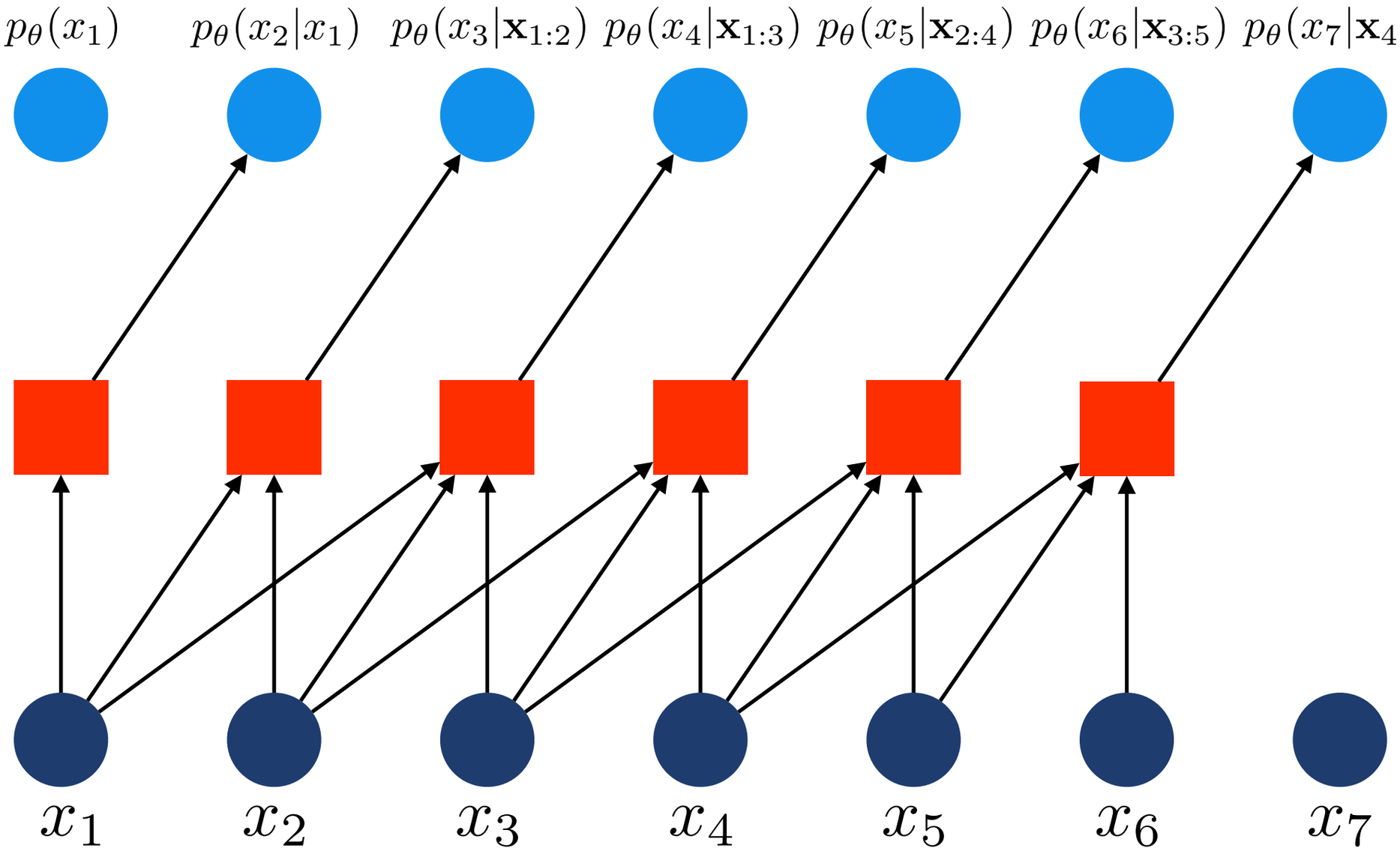
The Unreasonable Effectiveness of Recurrent Neural Networks, *Karpathy*, 2015



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

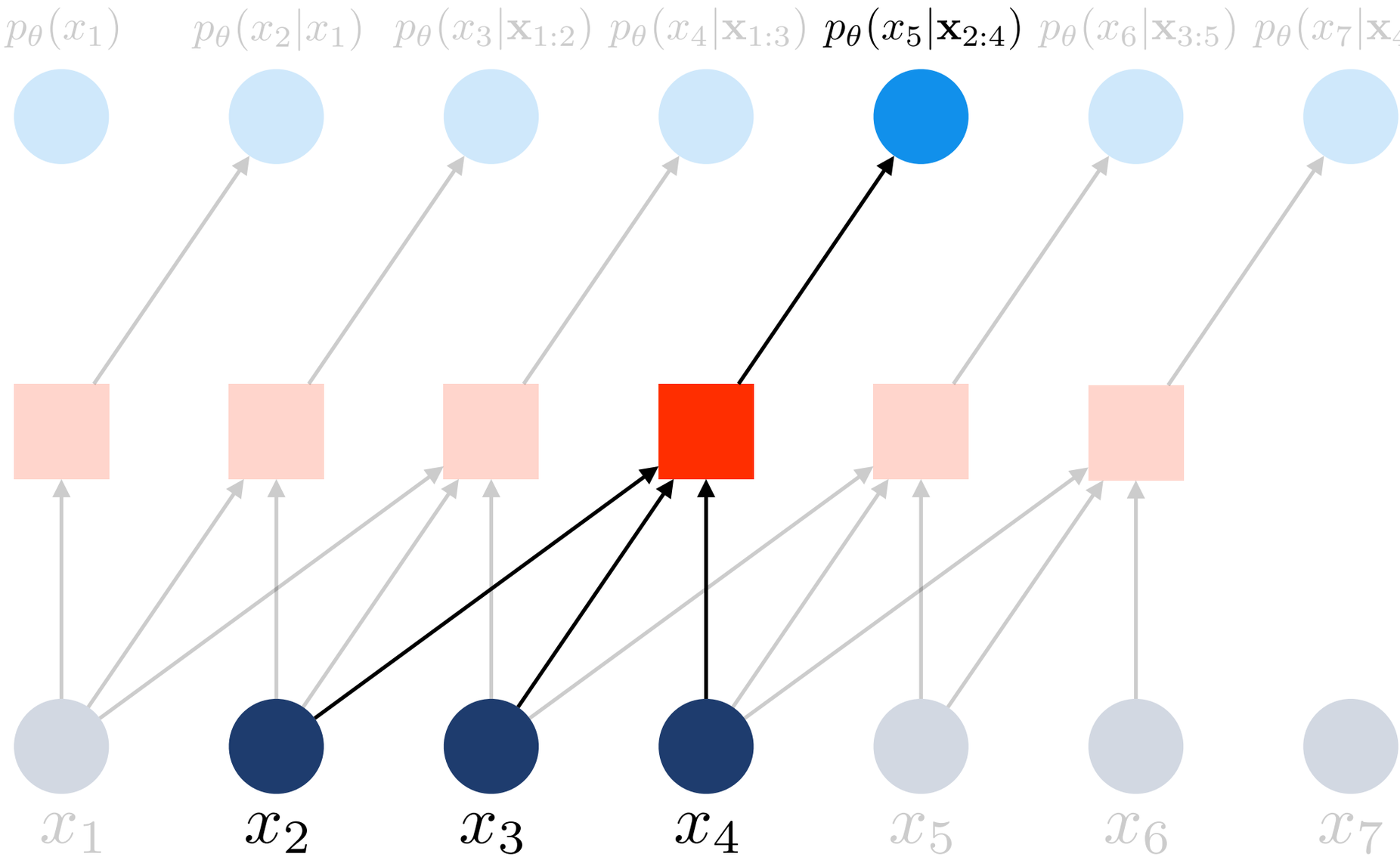
models

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



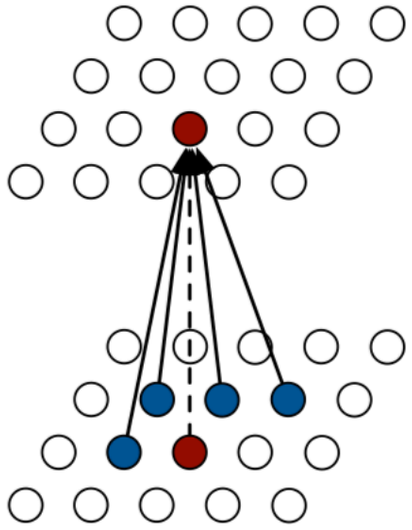
models

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



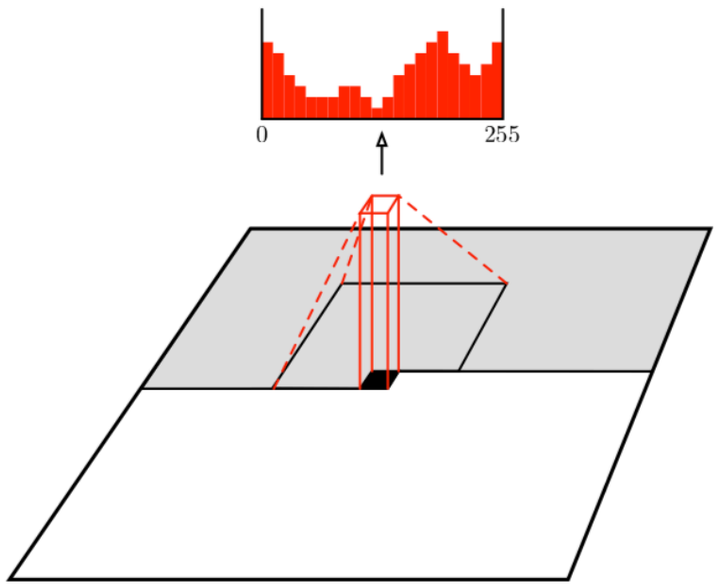
models

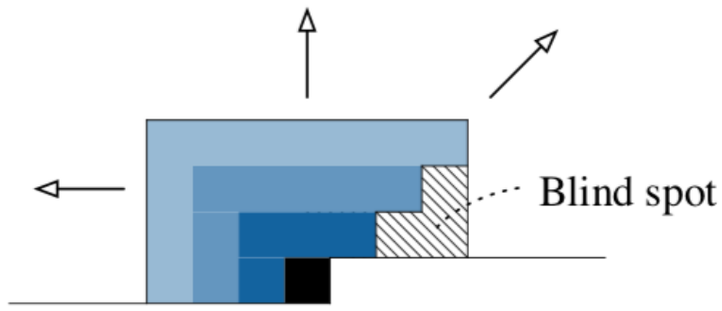
can condition on a local window using **convolutional neural networks**



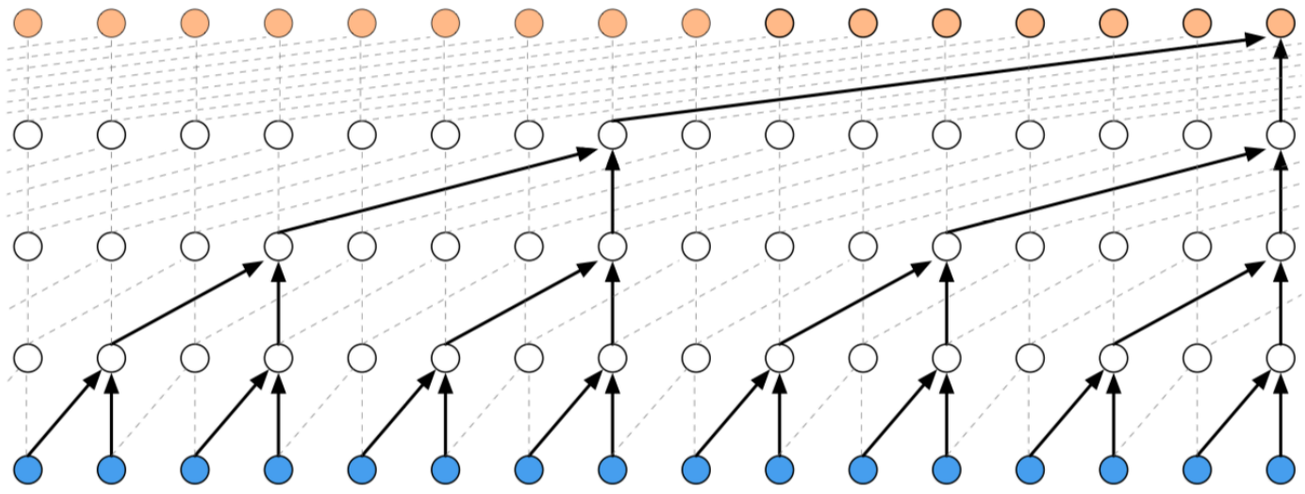
PixelCNN

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

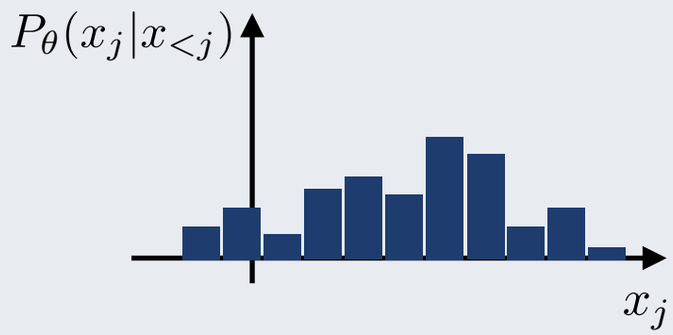
output distributions

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

model the data as…

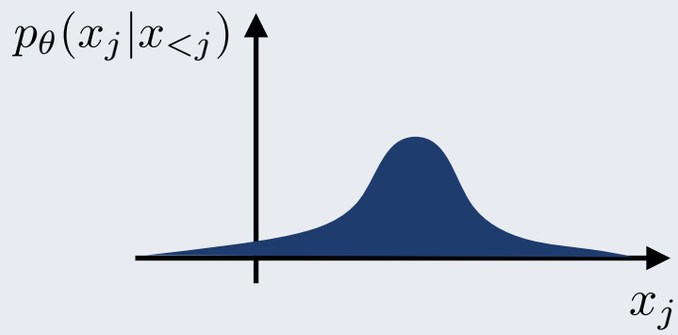
discrete *variables*

→ *categorical output*



continuous *variables*

→ *Gaussian, logistic, etc. output*



question

Image pixels take integer values between 0 and 255.

What are the trade-offs of using
***discrete* vs. *continuous* output distributions?**

Consider *continuous* $p_\theta(x_j|x_{<j}) = \mathcal{N}(x_j; \mu_\theta(x_{<j}), \sigma^2_\theta(x_{<j}))$

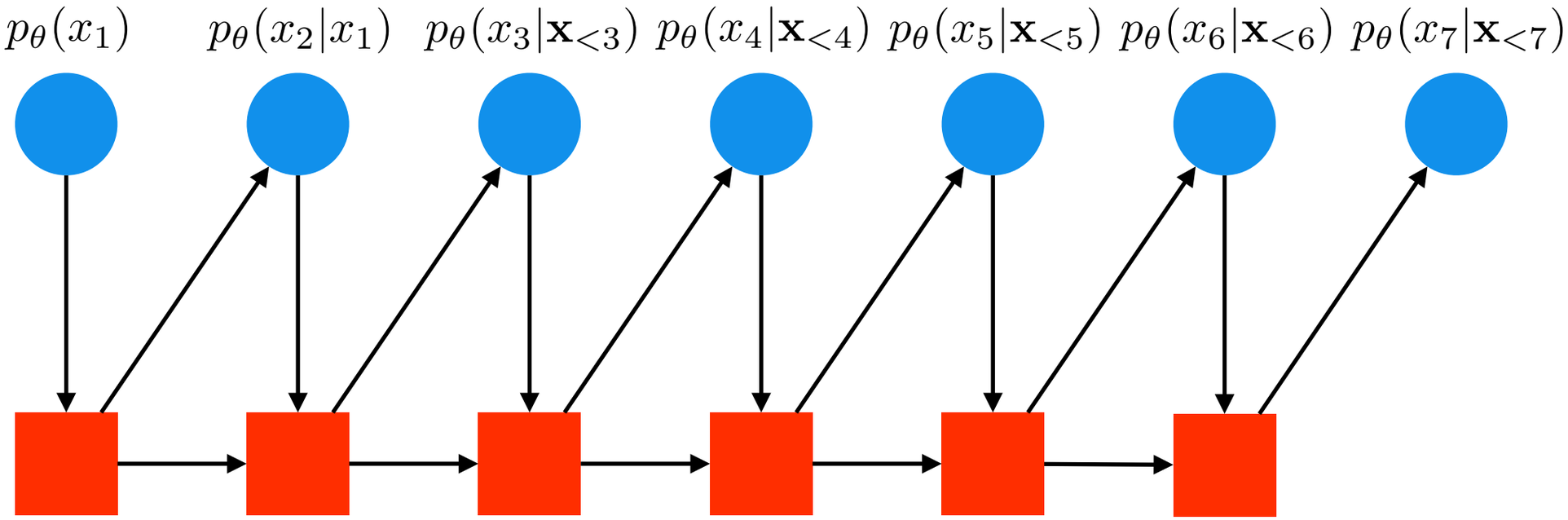
discrete $P_\theta(x_j|x_{<j}) = \text{Cat.}(x_j; \boldsymbol{\alpha}_\theta(x_{<j}))$

Which parameterization has fewer parameters? ***continuous***

Which parameterization can represent multi-modal outputs? ***discrete***

sampling

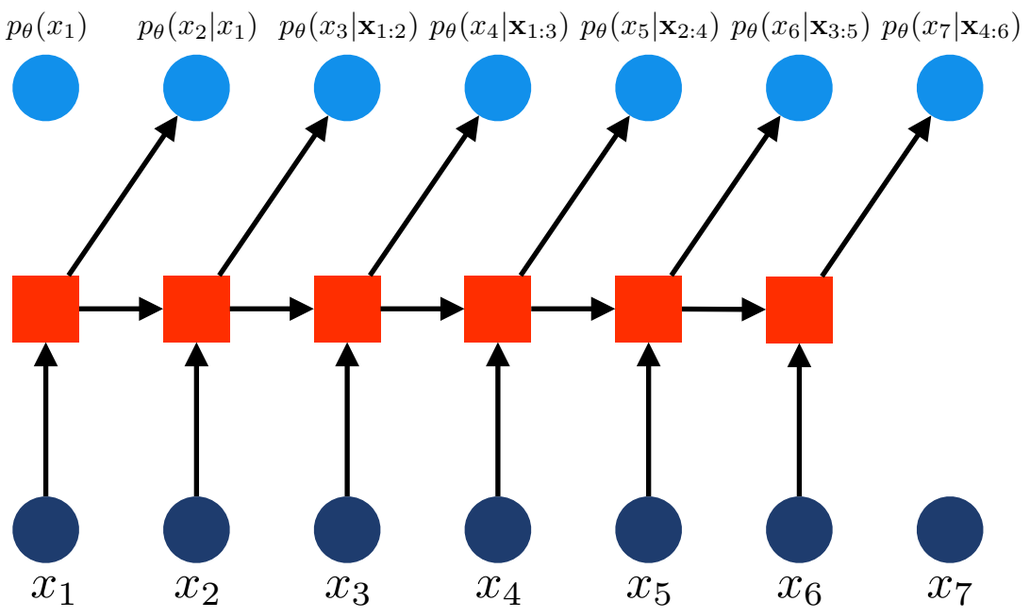
sample from the model by drawing from the output distribution



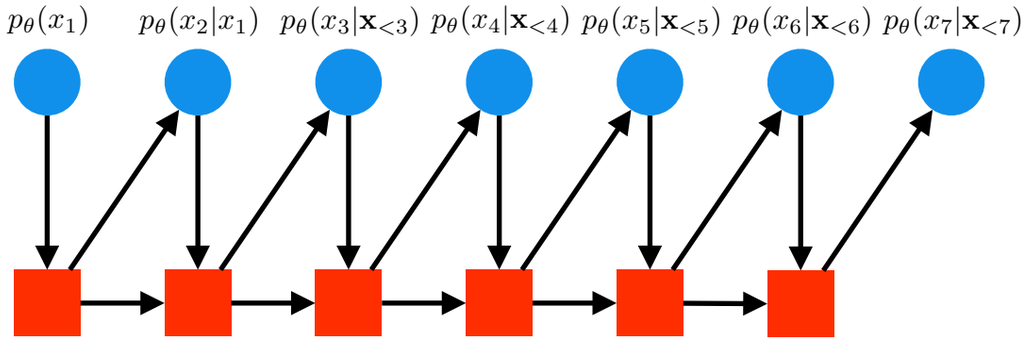
question

what issues might arise with sampling from the model?

training



sampling



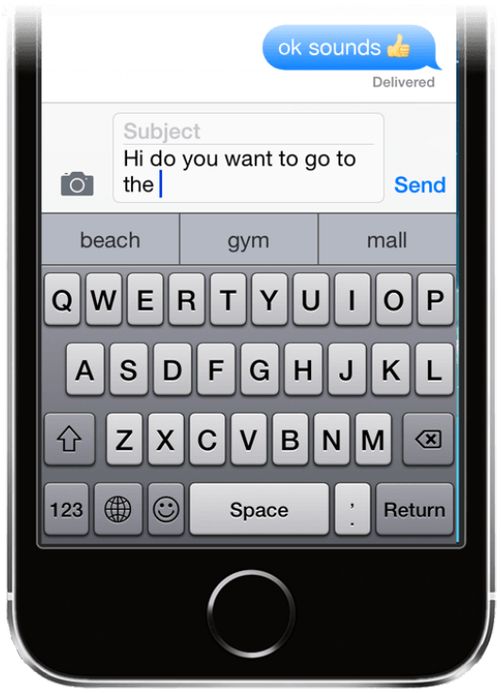
errors in the model distribution can accumulate, leading to poor samples

see *teacher forcing*, *scheduled sampling*

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

example applications

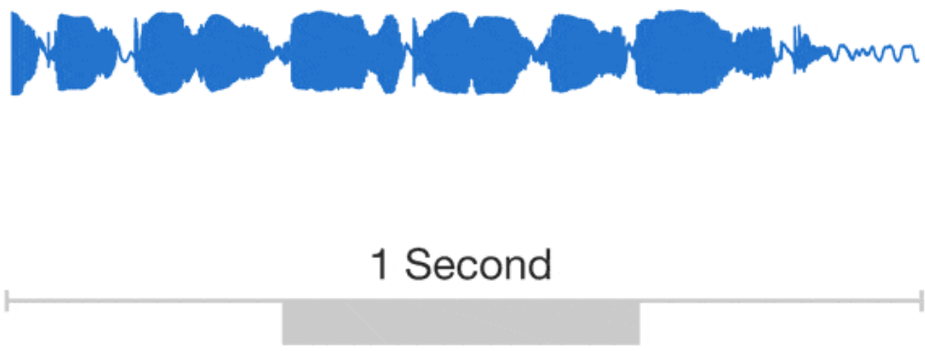
text



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.

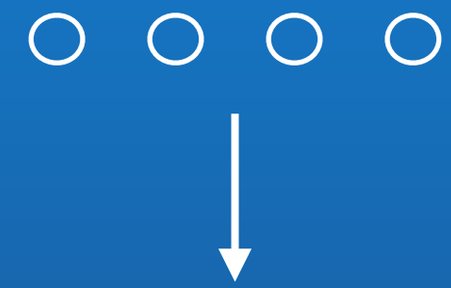
Attention is All You Need, Vaswani *et al.*, 2017

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

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

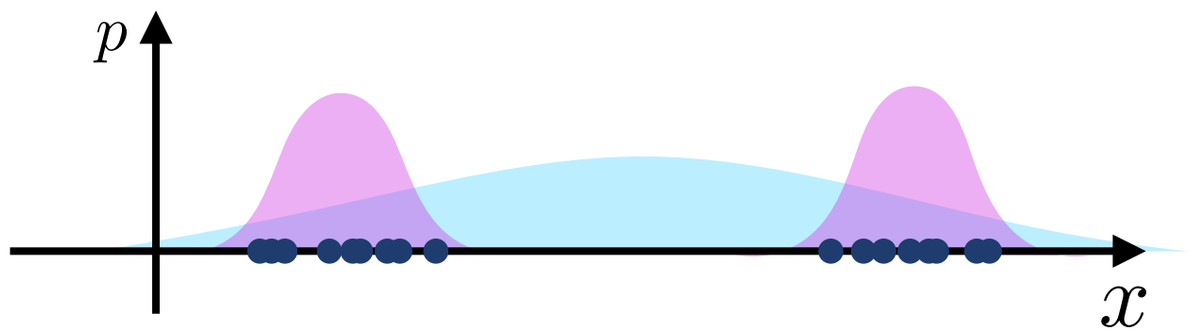
conditional autoregressive generation: *translation*

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html



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 help model the data

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

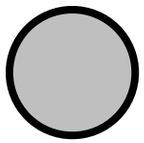
$$p_\theta(x) = \sum_z p_\theta(x, z)$$

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

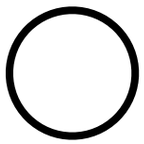
probabilistic graphical models provide a framework
for modeling relationships between random variables

PLATE NOTATION

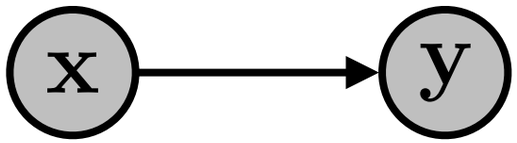
observed variable



unobserved (latent)
variable

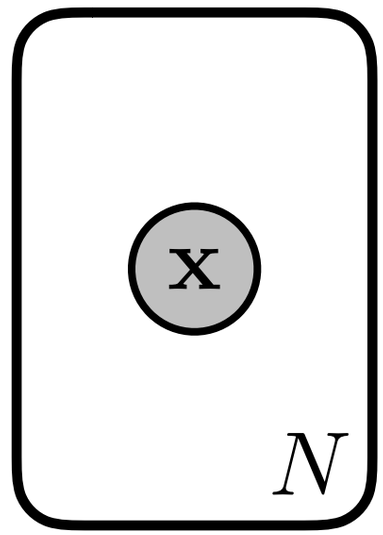


directed



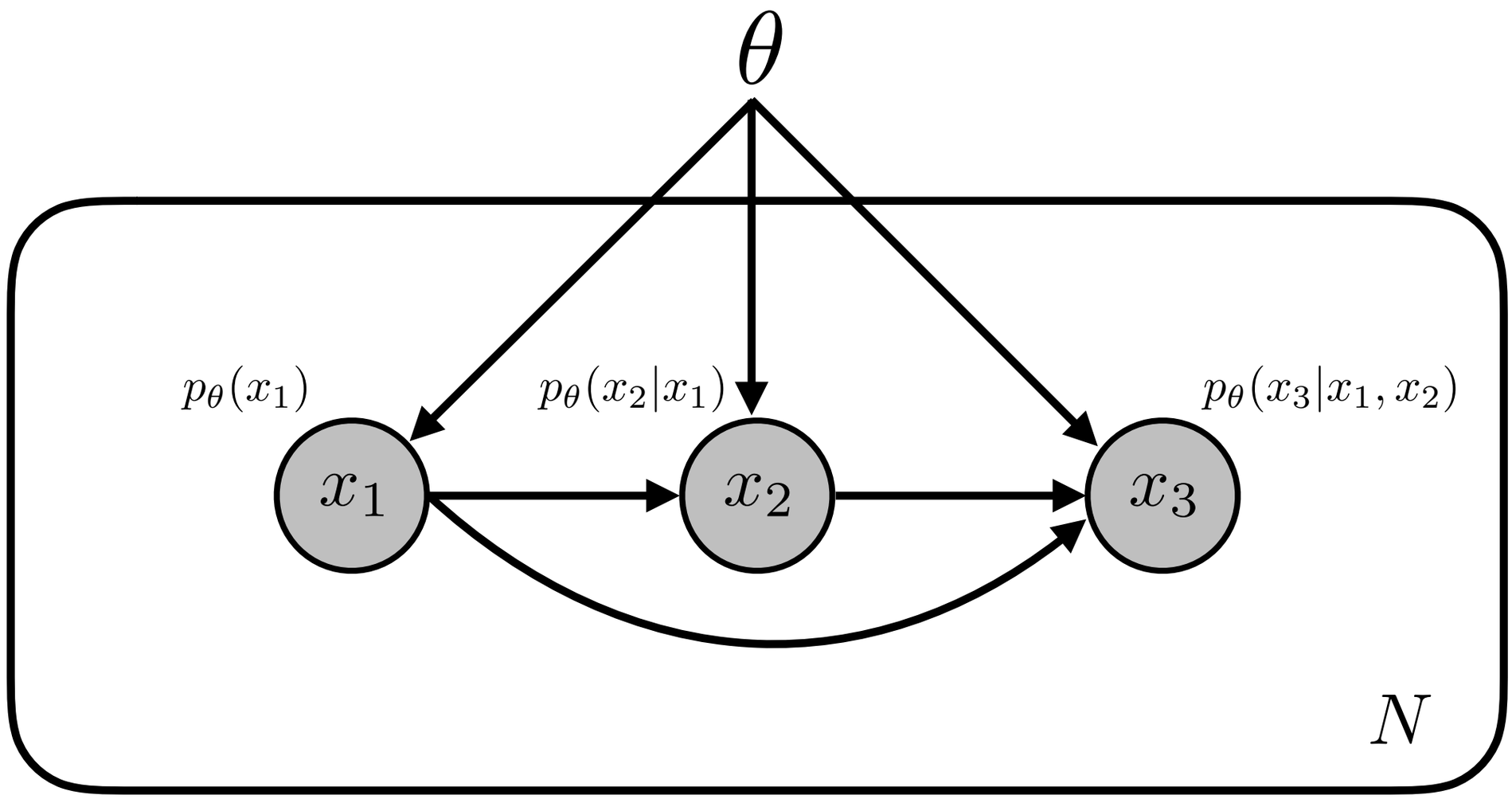
undirected

set of variables

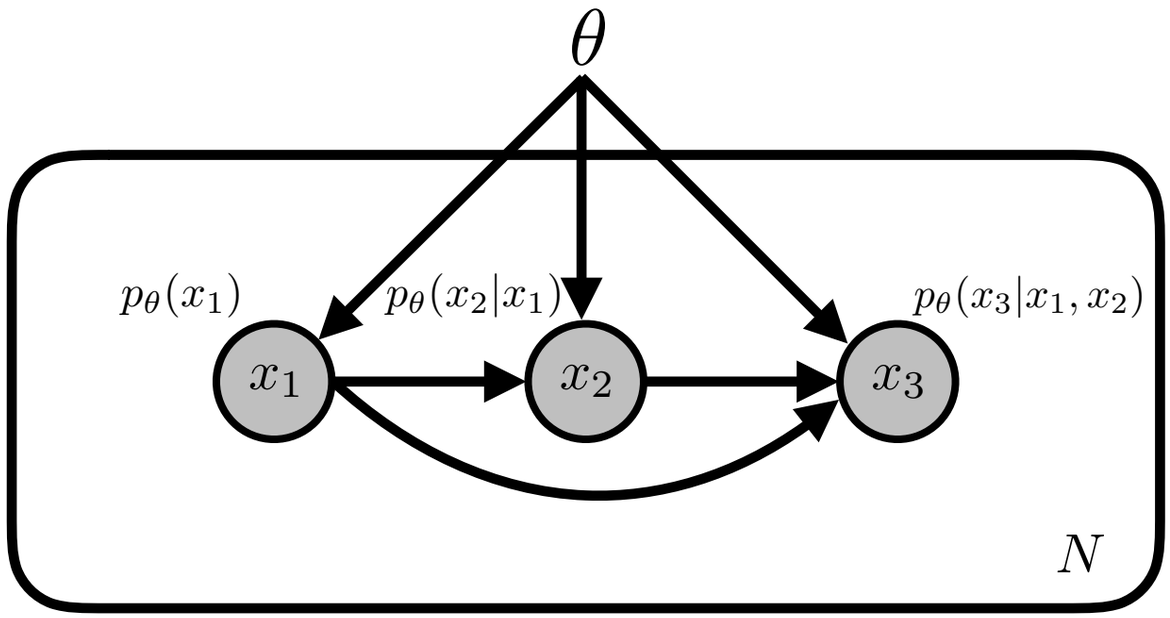


question

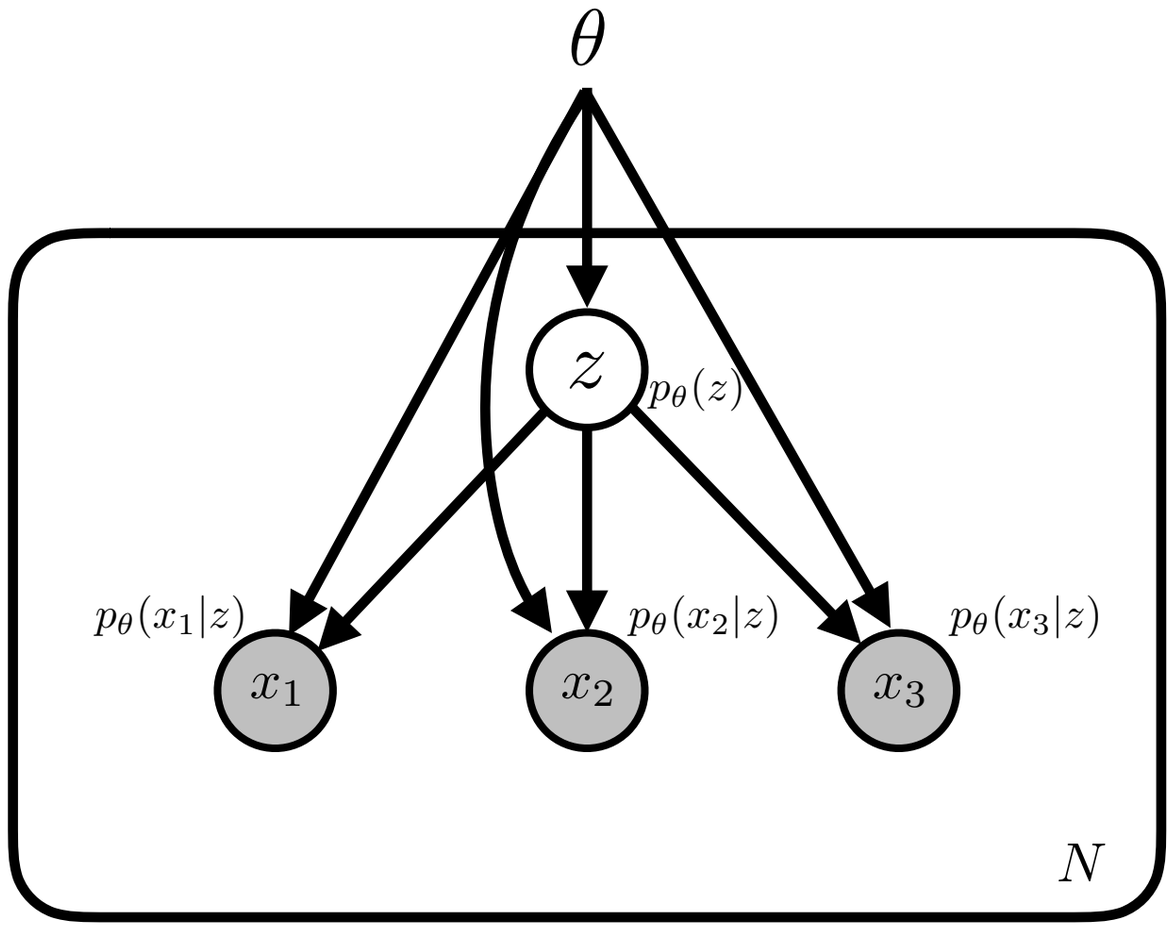
represent an auto-regressive model of 3 random variables
with plate notation



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

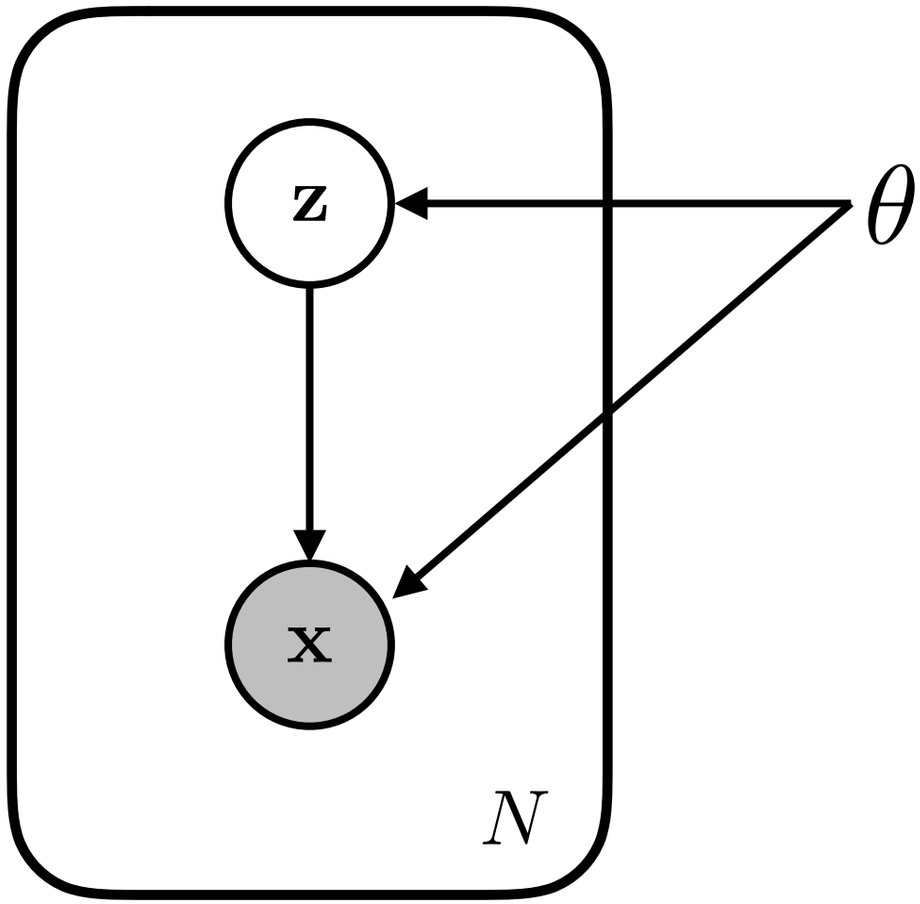


auto-regressive model



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*

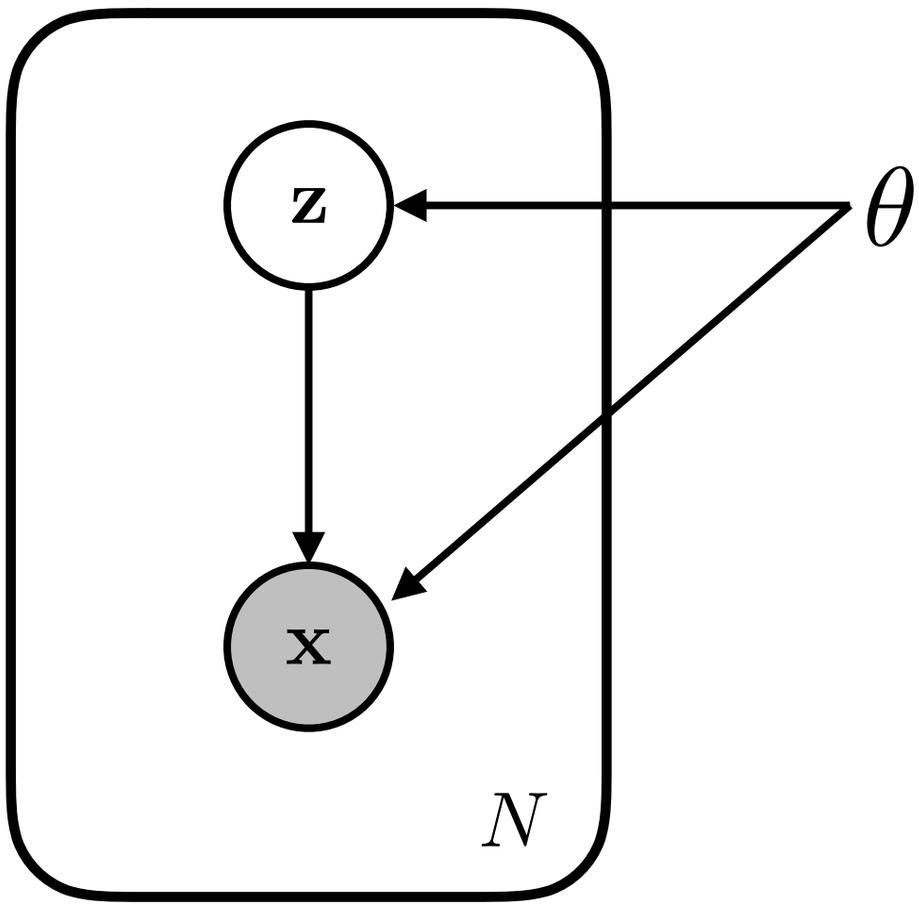
1. sample $\mathbf{z}$ from $p(\mathbf{z})$
2. use $\mathbf{z}$ samples to sample $\mathbf{x}$ from $p(\mathbf{x}|\mathbf{z})$

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

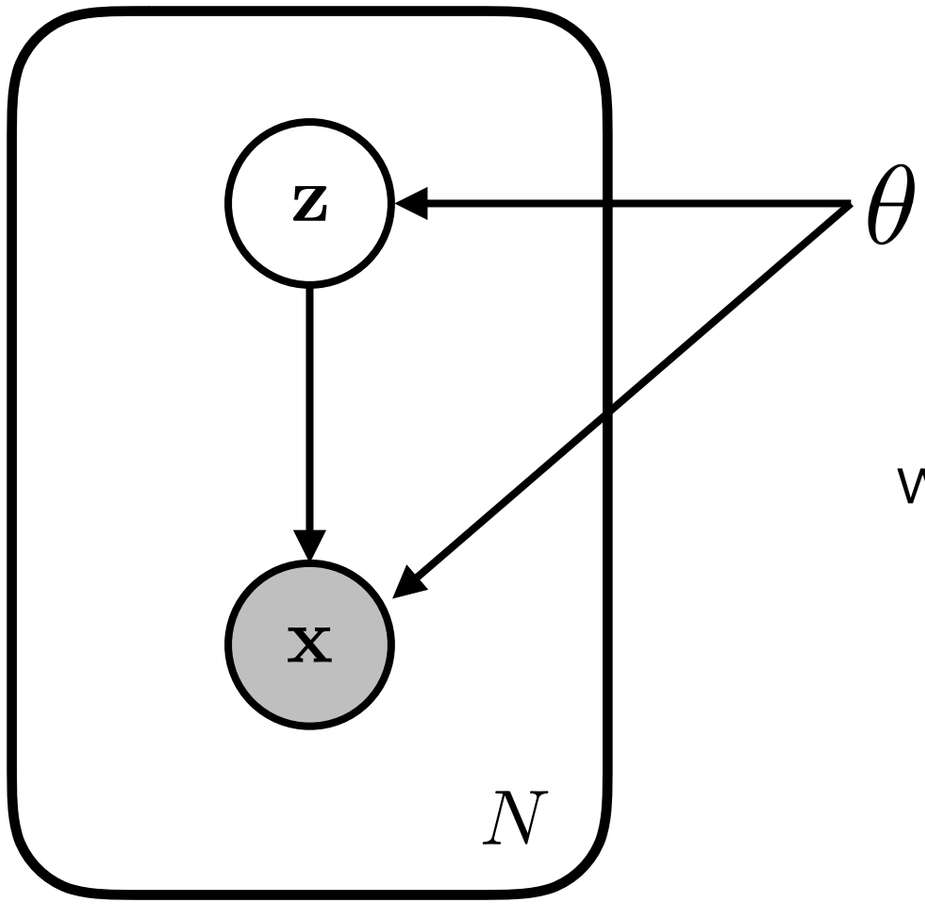
intuitive example

observation

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

$$p(\text{cat} \mid \text{image}) = \frac{p(\text{image} \mid \text{cat})\ p(\text{cat})}{p(\text{image})}$$

directed latent variable model



Model Evaluation

MARGINALIZATION

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

marginal likelihood

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*** $q(\mathbf{z}|\mathbf{x})$

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

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

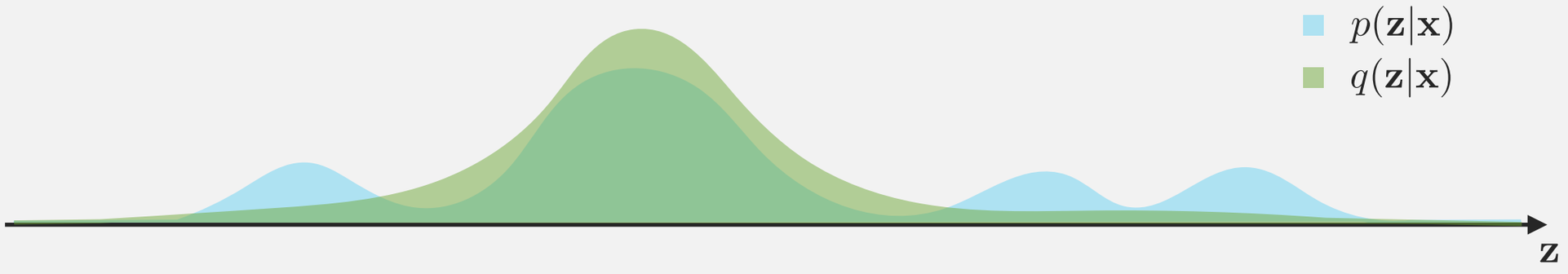
$$D_{KL} \geq 0 \longrightarrow \mathcal{L}(\mathbf{x};\theta,q) \leq \log p_\theta(\mathbf{x}) \quad \text{(lower bound)}$$

variational expectation maximization (EM)

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

M-Step: optimize $\mathcal{L}(\mathbf{x};\theta,q)$ *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

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

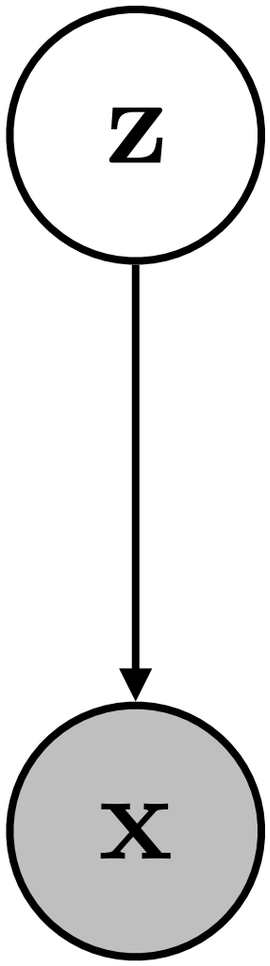
$q(\mathbf{z}|\mathbf{x})$ is optimized to **represent the data** while staying **close to the prior**

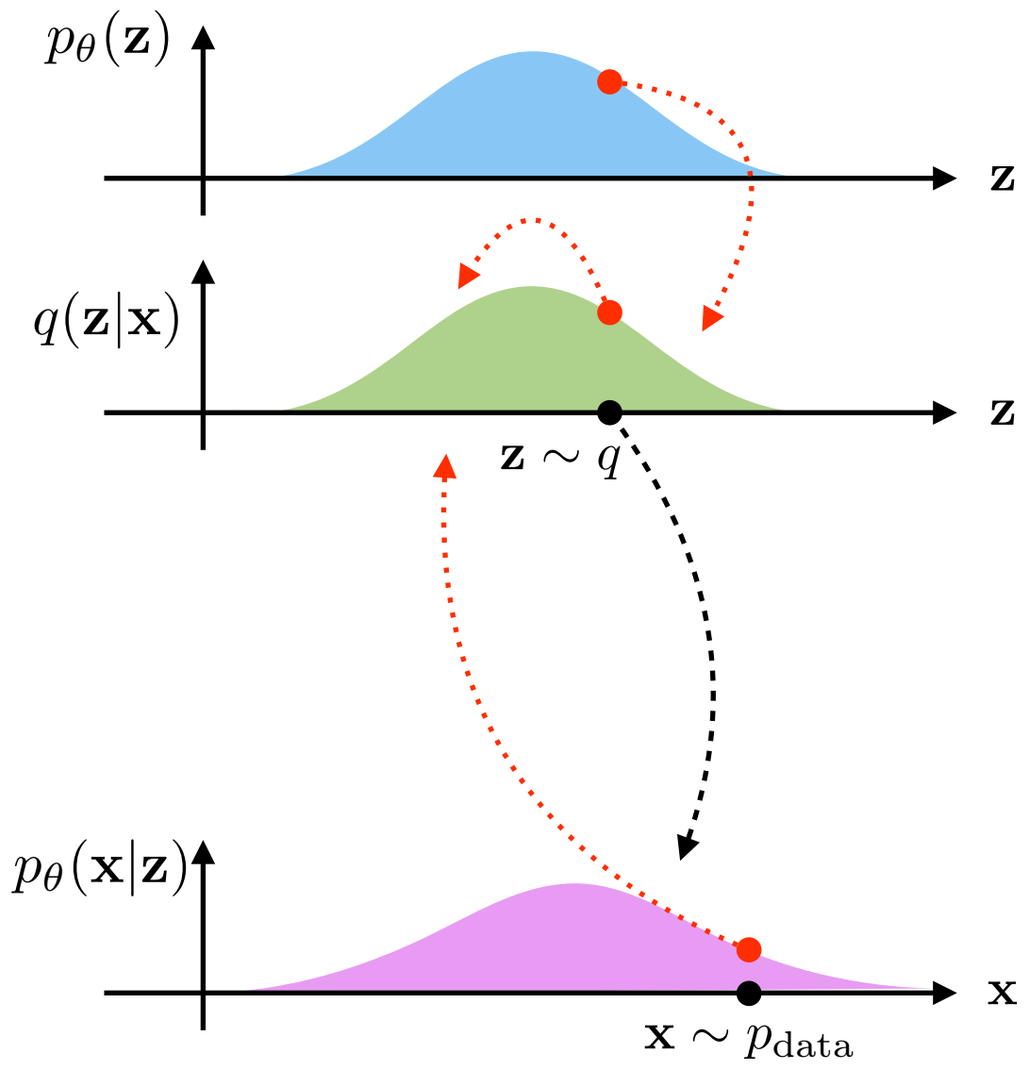
connections to *compression, information theory*
see Alemi et al., 2018

Variational Inference Optimization

$$q(\mathbf{z}|\mathbf{x}) \leftarrow \arg\max_q \mathcal{L}(\mathbf{x};\theta,q)$$

$$\mathcal{L}(\mathbf{x};\theta,q) = \mathbb{E}_q\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))$$

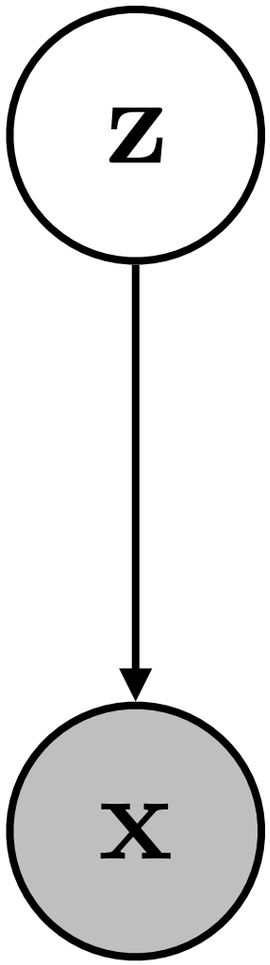


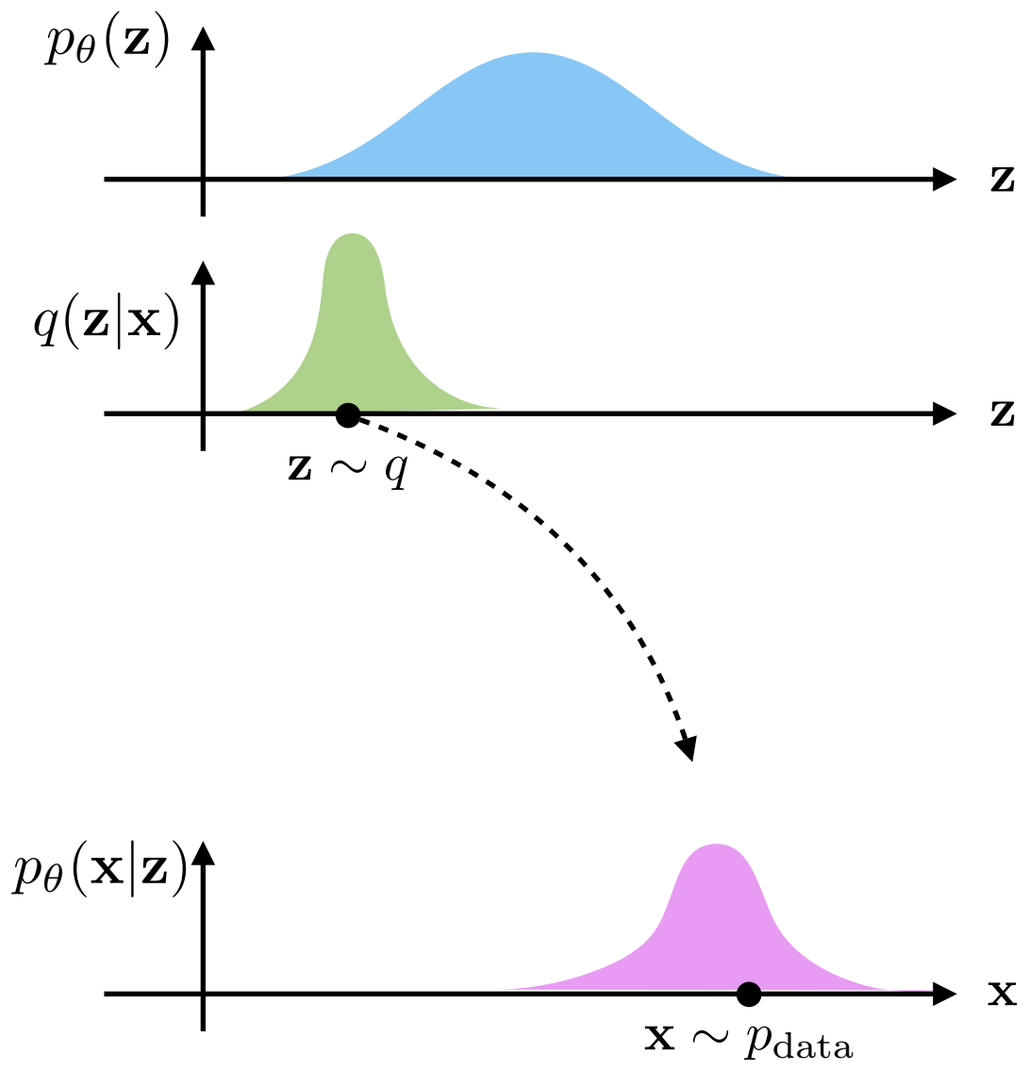


Variational Inference Optimization

$$q(\mathbf{z}|\mathbf{x}) \leftarrow \arg\max_q \mathcal{L}(\mathbf{x};\theta,q)$$

$$\mathcal{L}(\mathbf{x};\theta,q) = \mathbb{E}_q\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))$$





variational autoencoder (VAE)

variational expectation maximization (EM)

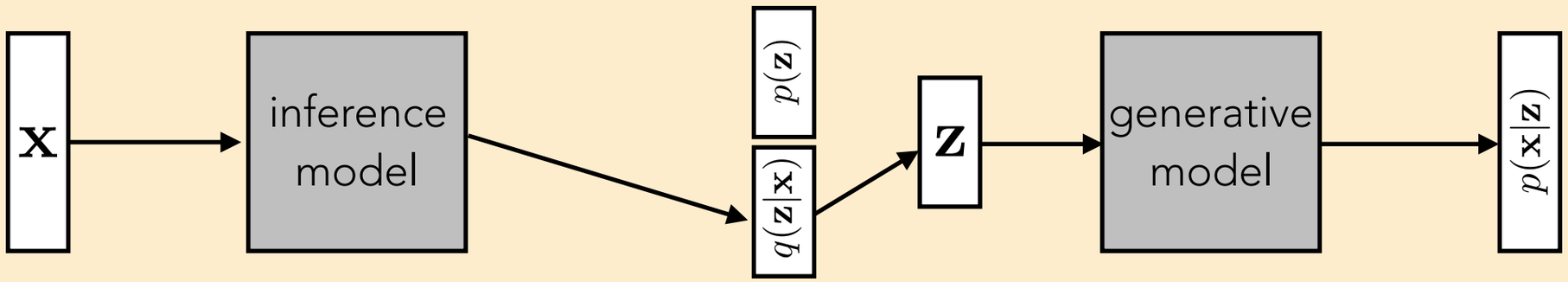
E-Step: optimize $\mathcal{L}(\mathbf{x})$ *w.r.t.* $q(\mathbf{z}|\mathbf{x})$ ***costly, must be performed per example***

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

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

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}^2_\phi(\mathbf{x}))$$

where $[\boldsymbol{\mu}_\phi(\mathbf{x}), \log \boldsymbol{\sigma}_\phi(\mathbf{x})] = \mathrm{NN}_\phi(\mathbf{x})$



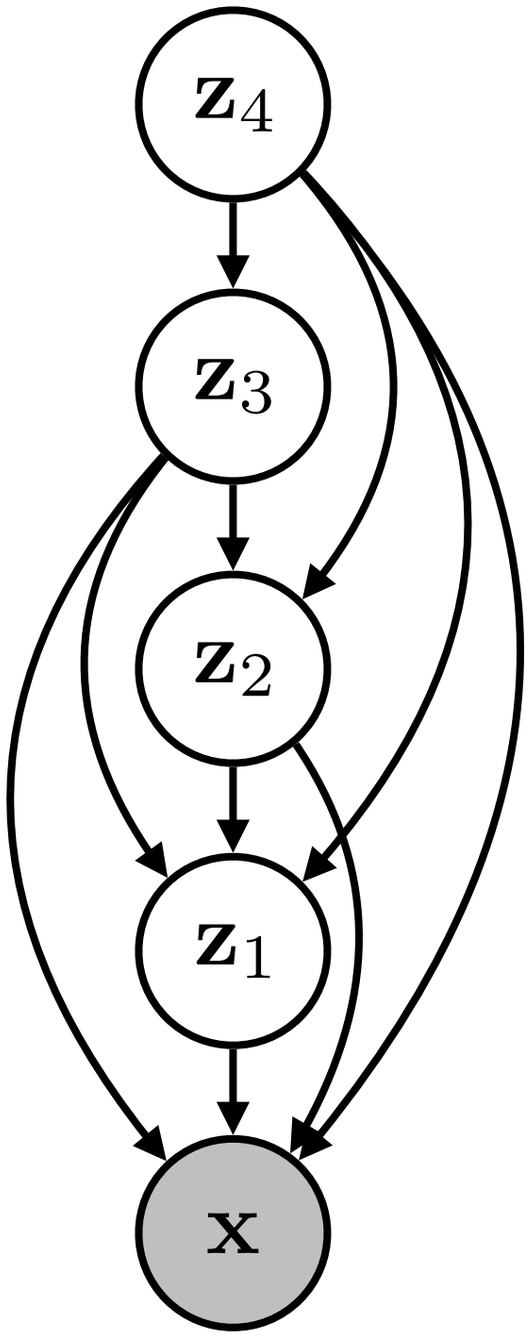
learn both models jointly using *stochastic backpropagation*

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

hierarchical latent variable models



hierarchy of latent variables $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4]$

how do we express $p_\theta(\mathbf{x}, \mathbf{z})$ *?*

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z}) \underbrace{\prod_\ell p_\theta(\mathbf{z}_\ell|\mathbf{z}_{>\ell})}_{\text{prior}}$$

hierarchy creates a more complex *empirical* prior

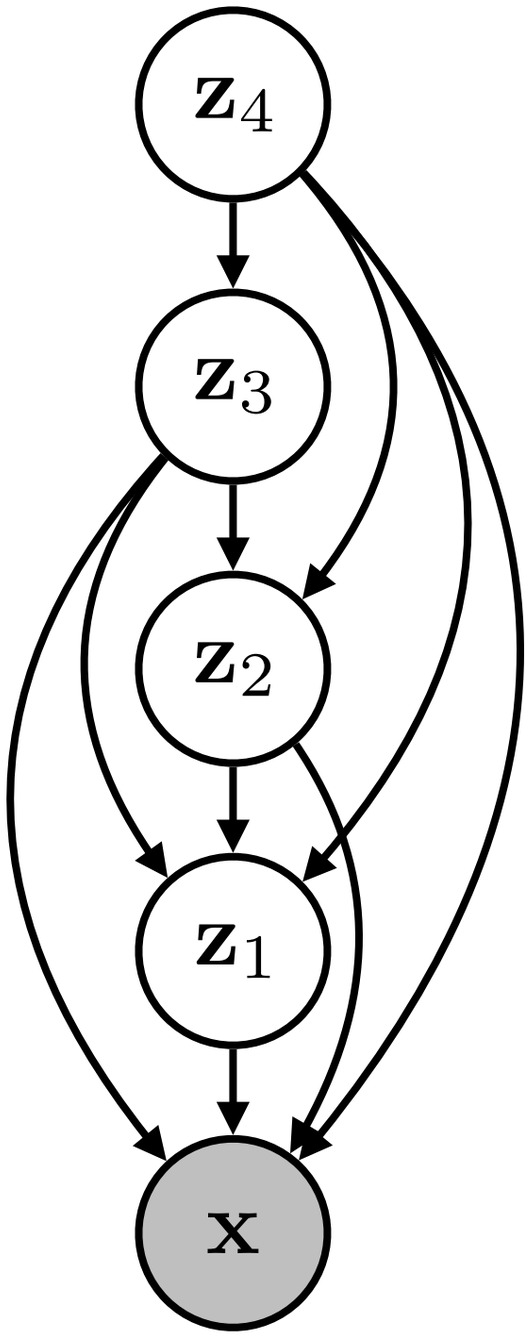
note: this prior is <u>autoregressive</u>

requires a hierarchical approximate posterior

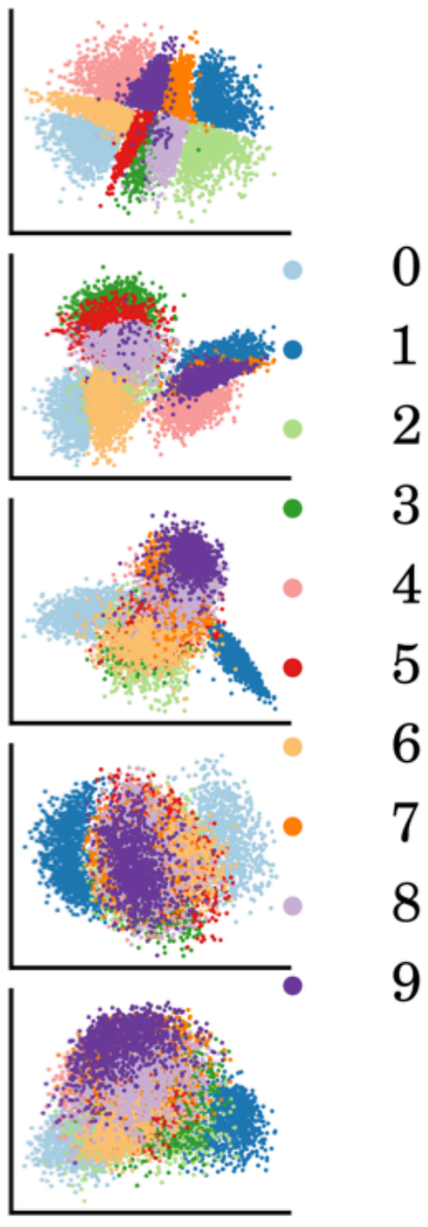
$$q(\mathbf{z}|\mathbf{x}) = \prod_\ell q(\mathbf{z}_\ell|\mathbf{x}, \mathbf{z}_{>\ell})$$

see, e.g. **Ladder VAE,** Sønderby et al., 2016

hierarchical latent variable models



MNIST



Faces

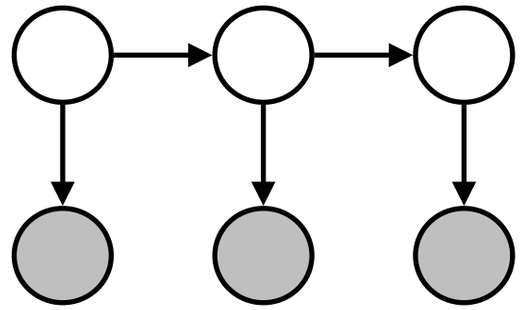
see, e.g. **Ladder VAE,** Sønderby et al., 2016

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

autoregressive latent variable models

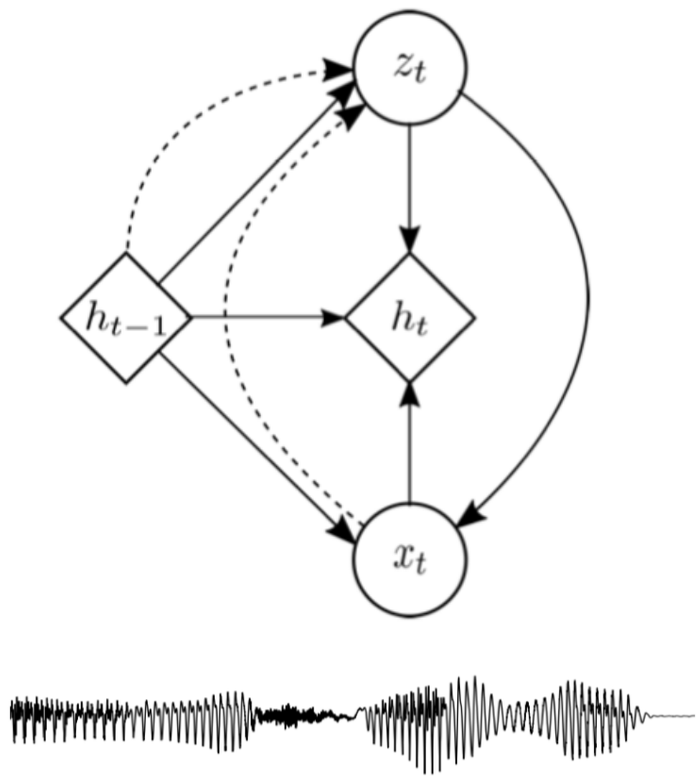
***autoregressive* latent variable models**

e.g. Markov model

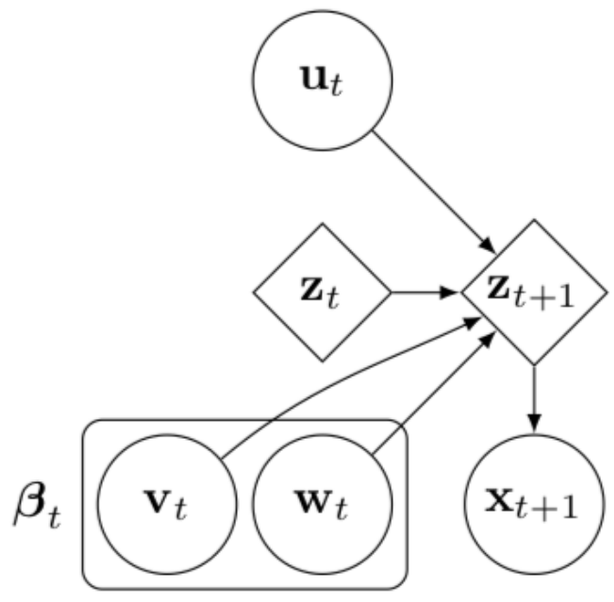


general form: $$p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) = \prod_t p_\theta(\mathbf{x}_t|\mathbf{x}_{<t}, \mathbf{z}_{\leq t}) p_\theta(\mathbf{z}_t|\mathbf{x}_{<t}, \mathbf{z}_{<t})$$

some examples:



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



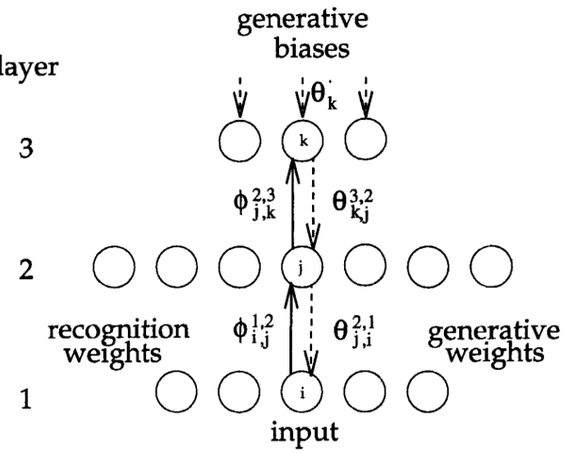
Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data, *Karl et al.*, 2016

discrete latent variable models

we often consider continuous latent variables; what about discrete variables?

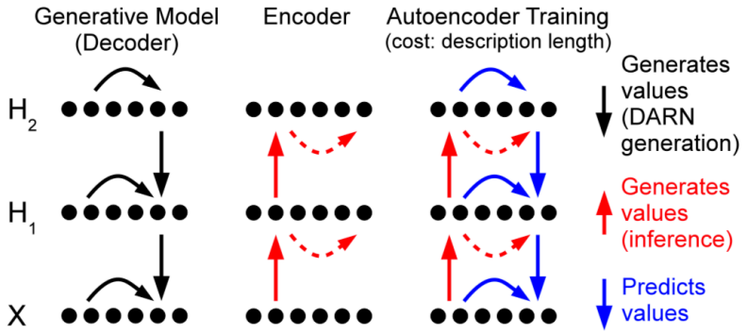
problem: cannot easily backprop through discrete sampling of $\mathbf{z}$

Helmholtz Machine / Wake-Sleep



Dayan *et al.*, 1995

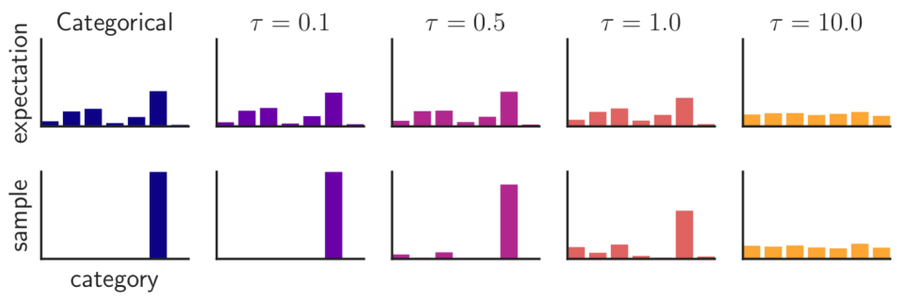
REINFORCE Gradients



Gregor *et al.*, 2014

Mnih & Gregor, 2014

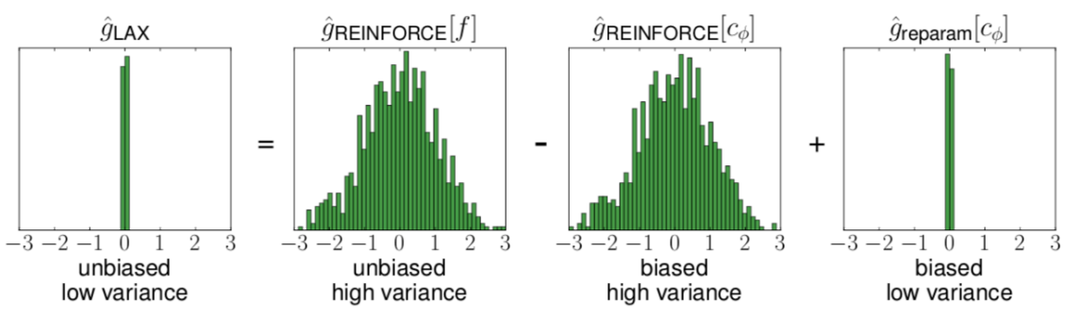
Relaxed Distributions



Jang *et al.*, 2017

Maddison *et al.*, 2017

Combinations



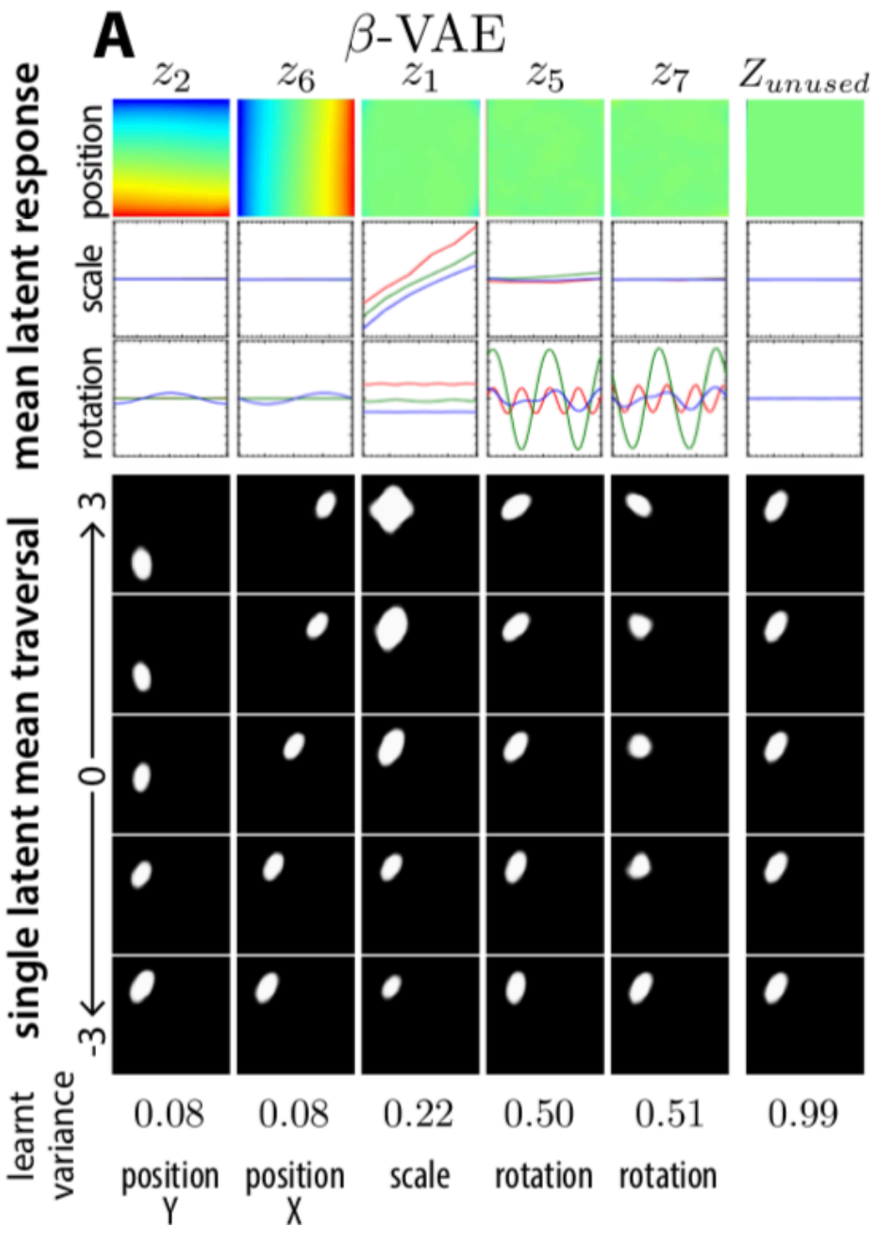
Tucker *et al.*, 2017

Grathwohl *et al.*, 2018

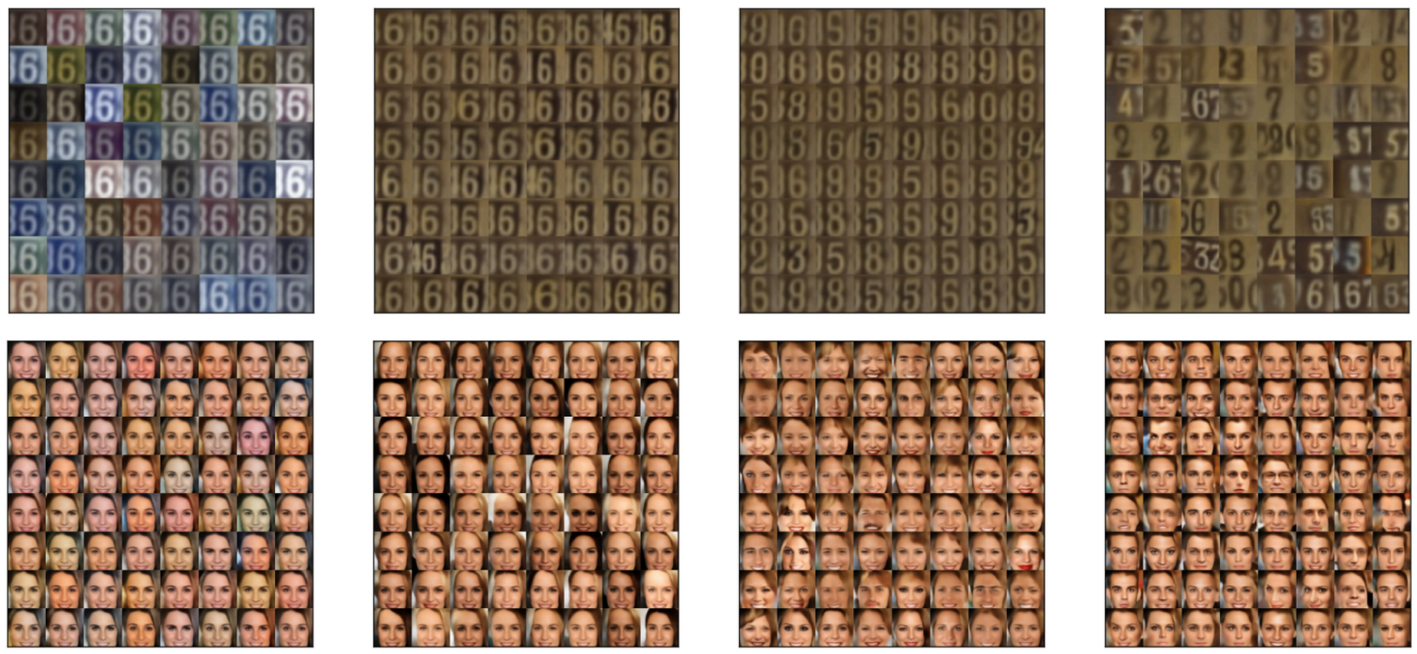
representation learning

latent variables provide a representation for downstream tasks

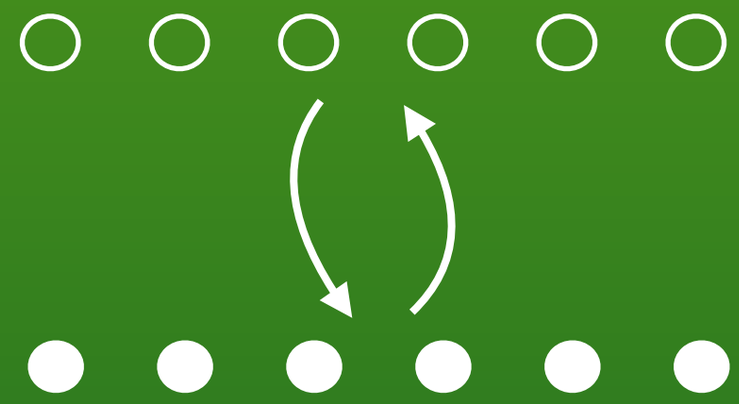
beta-VAE



Higgins *et al.*, 2017



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

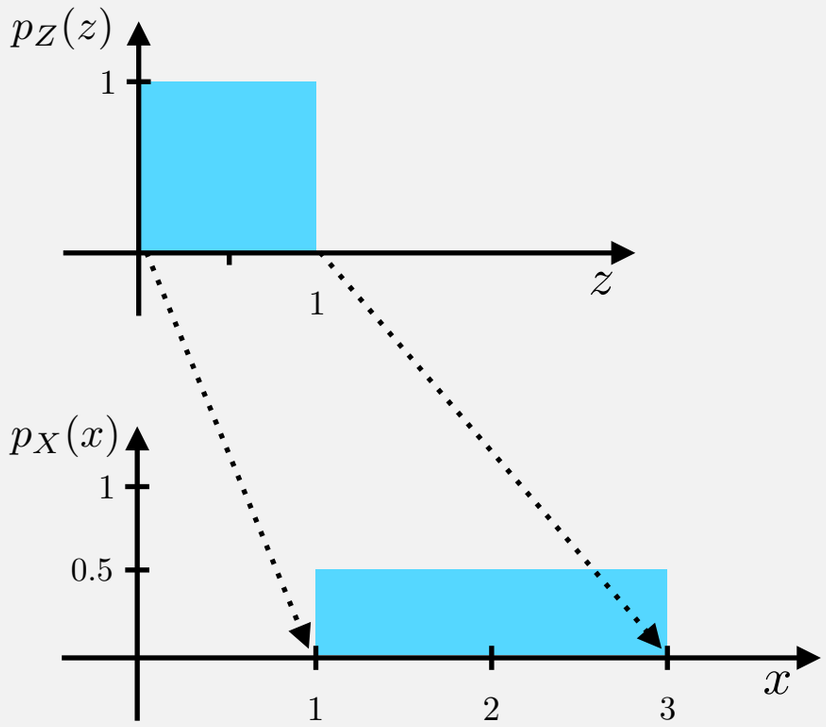


invertible explicit
latent variable models

change of variables

use an invertible mapping to directly evaluate the log likelihood

example



sample z from a *base distribution*

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

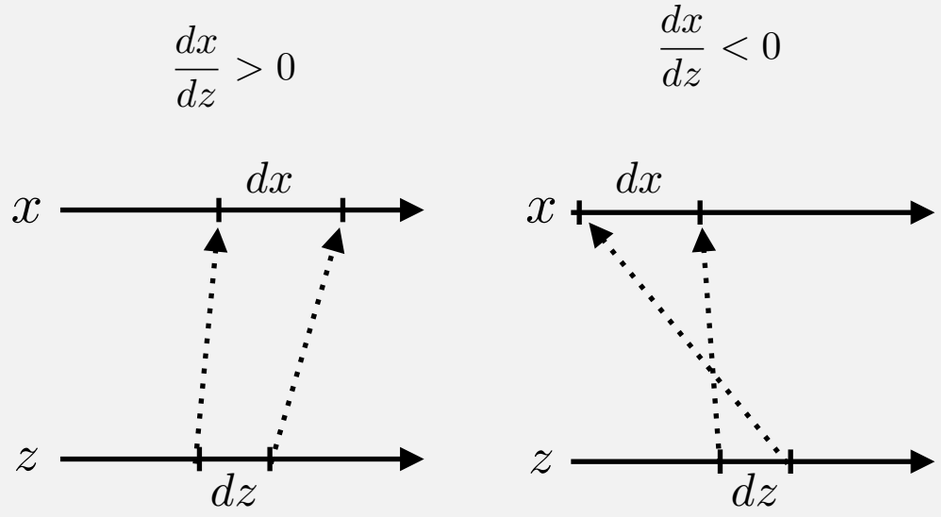
apply a transform to z to get a *transformed distribution*

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

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

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

conservation of probability mass

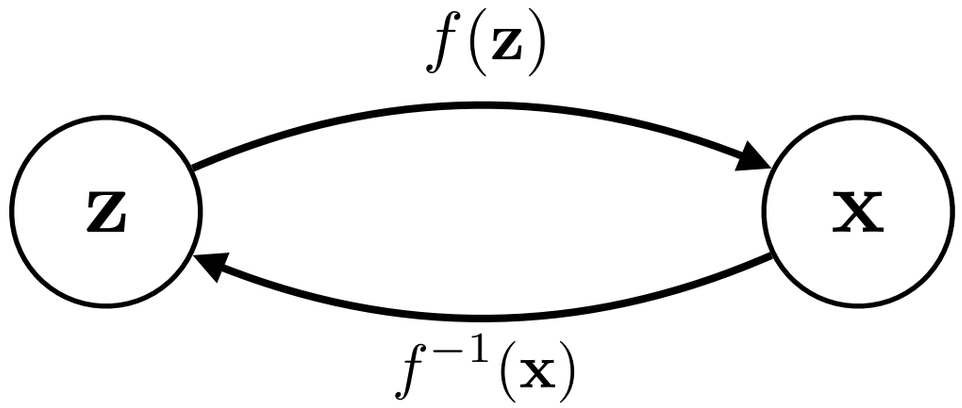


Normalizing Flows Tutorial, *Eric Jang*, 2018

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

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

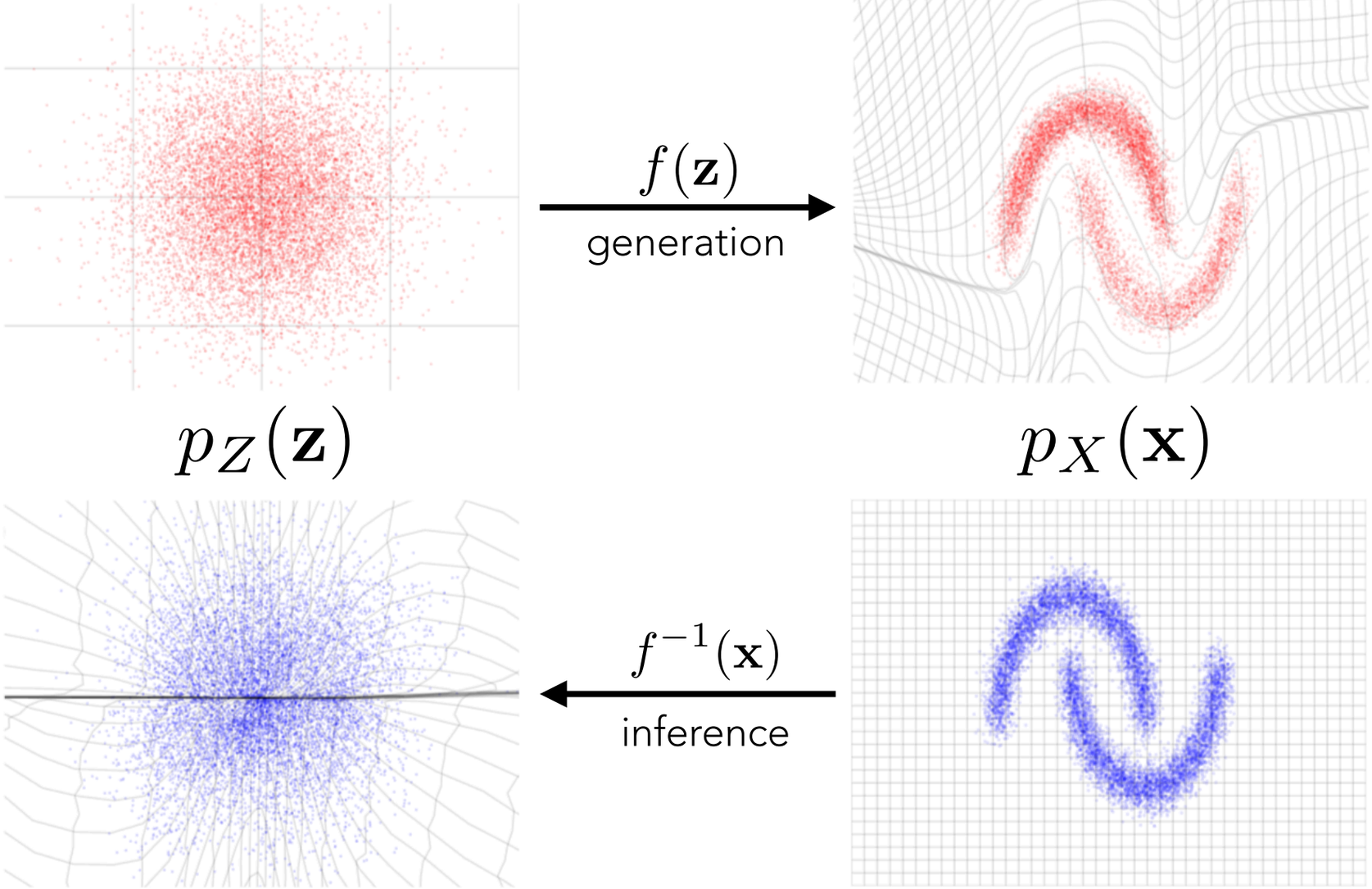
$\mathbf{J}(f^{-1}(\mathbf{x}))$ is the *Jacobian* matrix of the inverse transform

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



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 $O(N^3)$

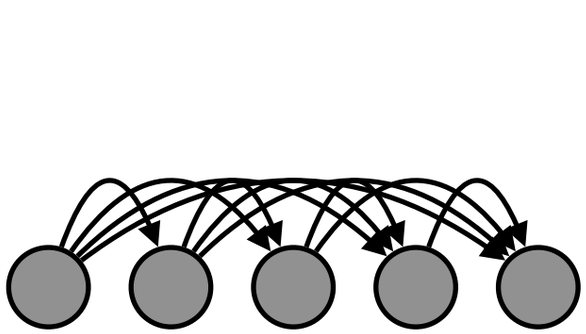
restrict the transforms to those with diagonal or triangular inverse Jacobians

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

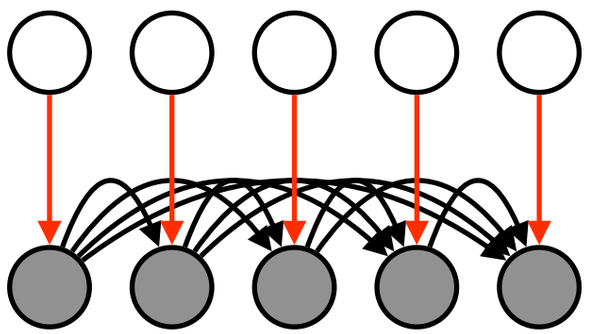
⟶ *product of diagonal entries*

masked autoregressive flow (MAF)

Gaussian *autoregressive sampling* can be interpreted as a transformed distribution



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



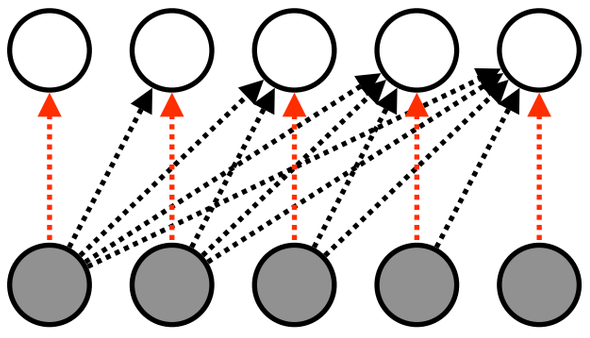
$$x_i = \mu_i(\mathbf{x}_{1:i-1}) + \sigma_i(\mathbf{x}_{1:i-1}) \cdot z_i$$

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

must generate each x_i *sequentially*

however, we can parallelize the inverse transform:

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



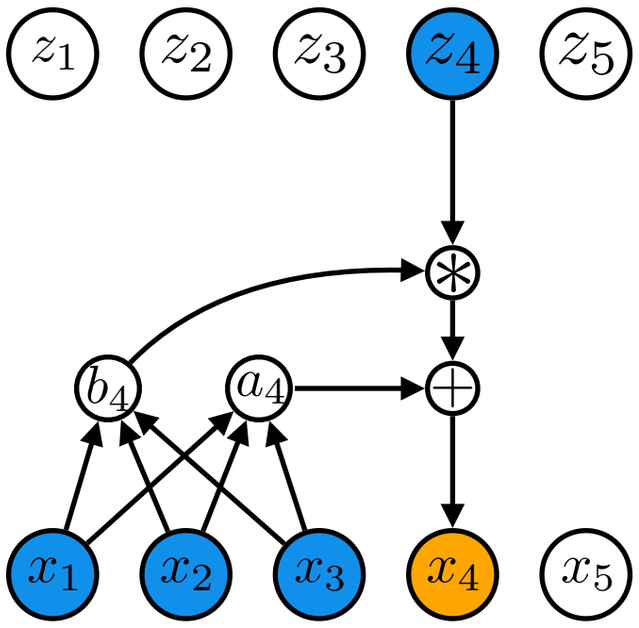
Masked Autoregressive Flow, Papamakarios *et al.*, 2017

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

masked autoregressive flow (MAF)

FORWARD TRANSFORM

base distribution

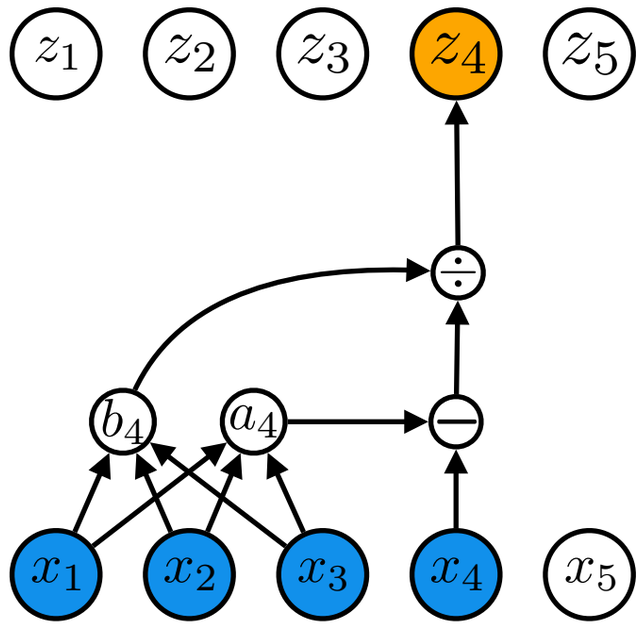


transformed distribution

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

INVERSE TRANSFORM

base distribution



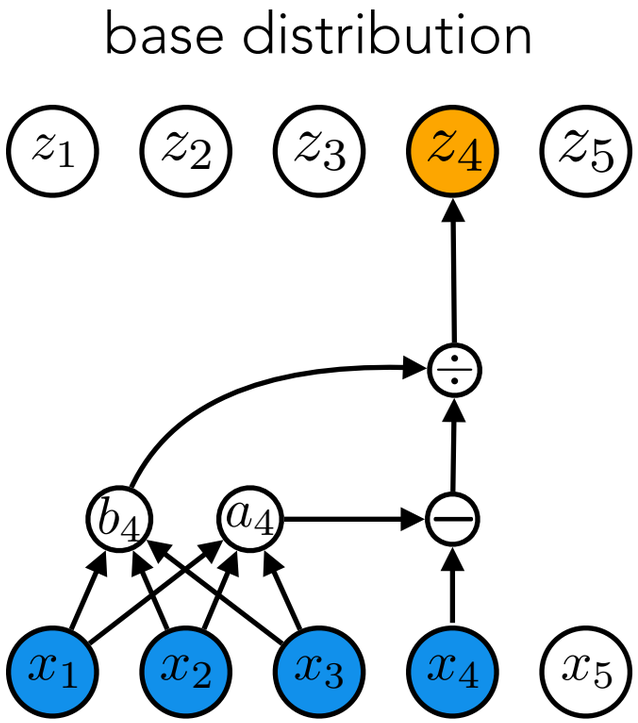
transformed distribution

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

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

question

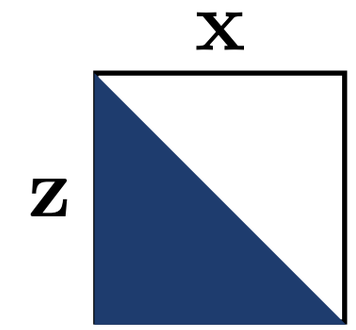
INVERSE TRANSFORM



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

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

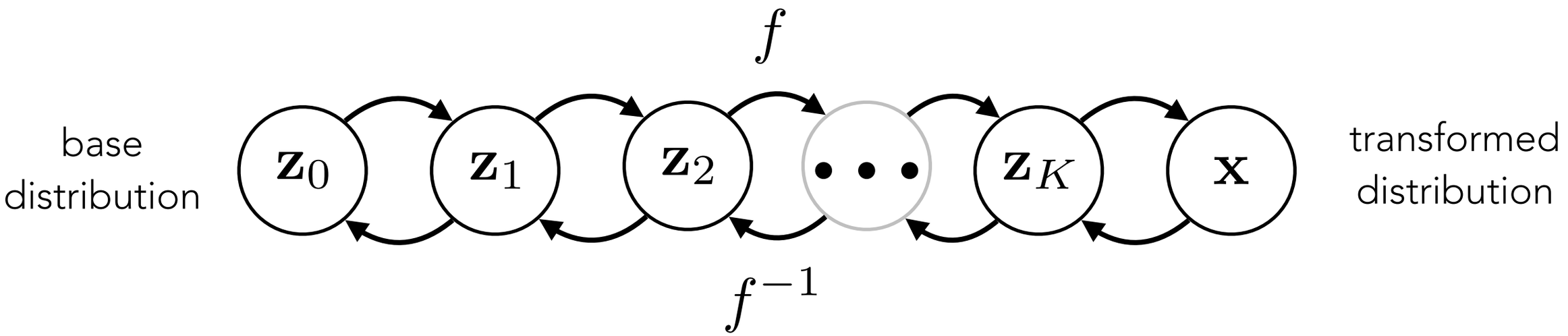
product of diagonal elements of $\mathbf{J}(f^{-1}(\mathbf{x}))$

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

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

compositions

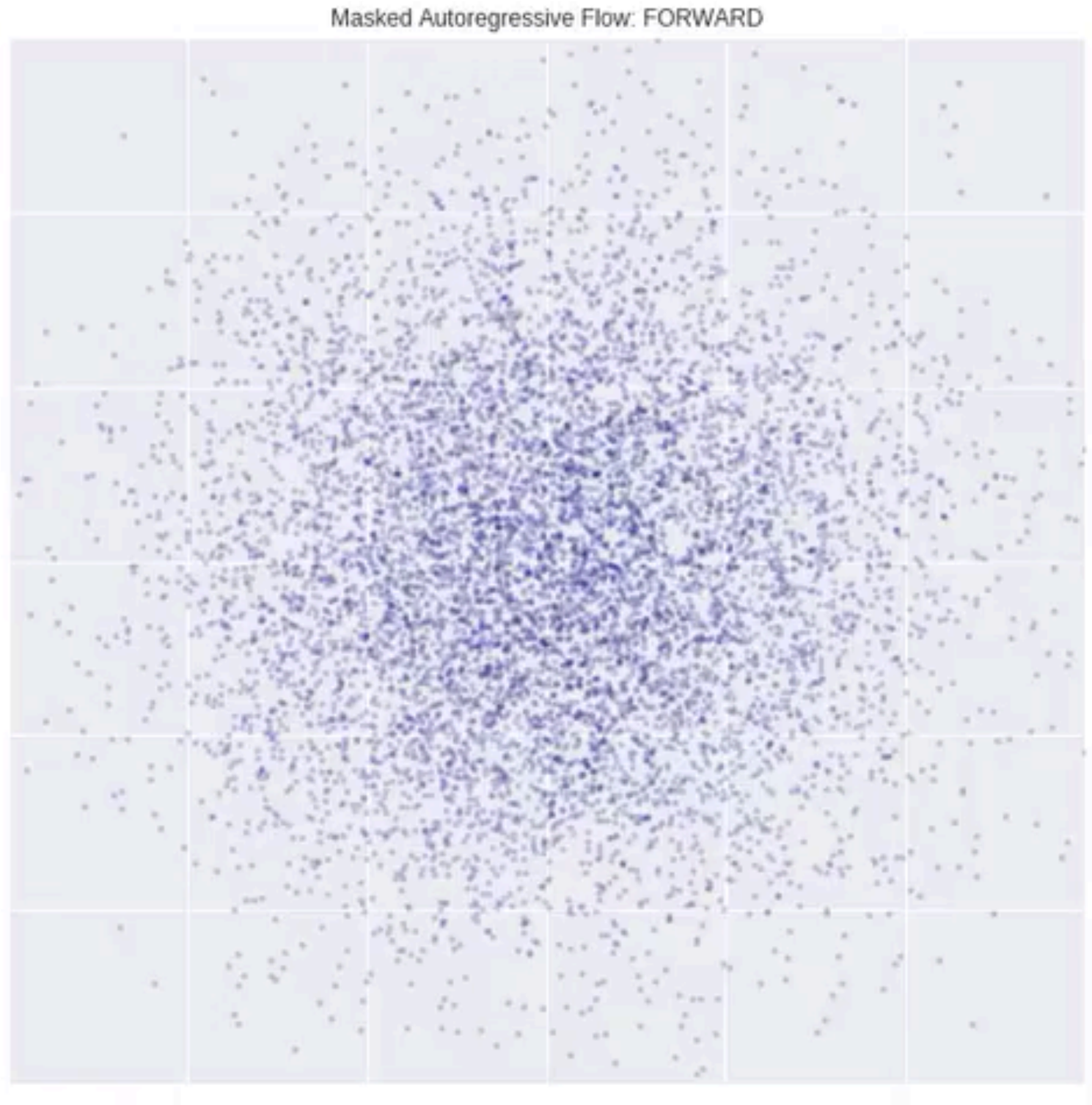
we can compose multiple transforms to create a *flow*



repeatedly apply the change of variables formula

$$p_X(\mathbf{x}) = p_Z(\mathbf{z}_0) \left| \det \frac{\partial \mathbf{x}}{\partial \mathbf{z}_K} \right|^{-1} \prod_{k=1}^{K} \left| \det \frac{\partial \mathbf{z}_k}{\partial \mathbf{z}_{k-1}} \right|^{-1}$$

compositions

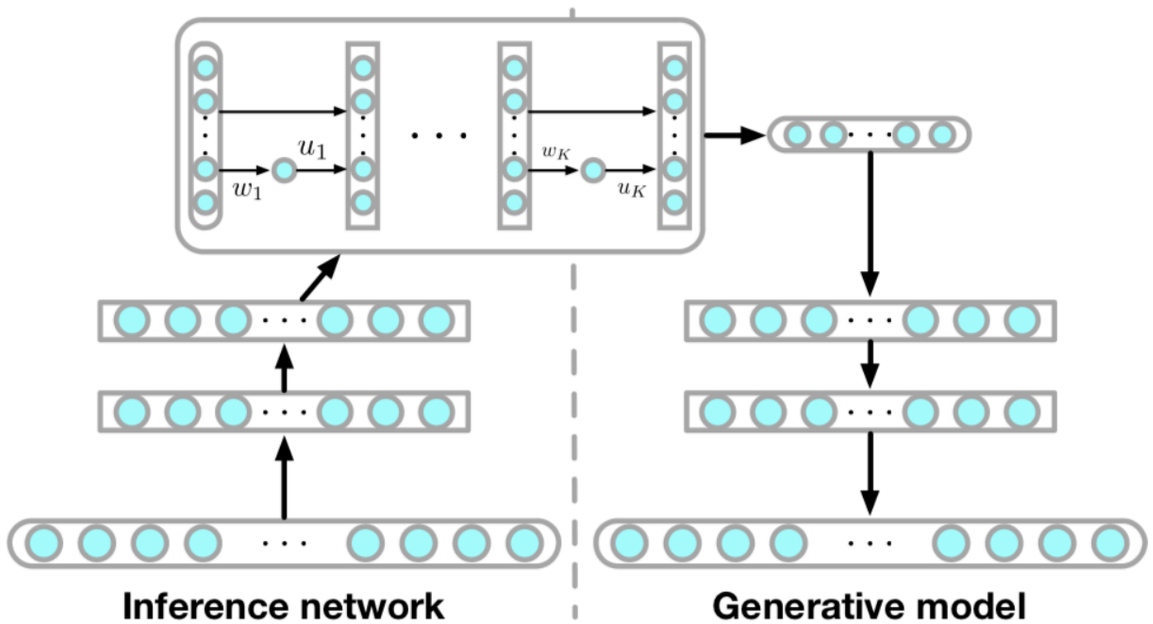


Normalizing Flows Tutorial, Jang, 2018

normalizing flows (NF)

can also use the change of variables formula for variational inference

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



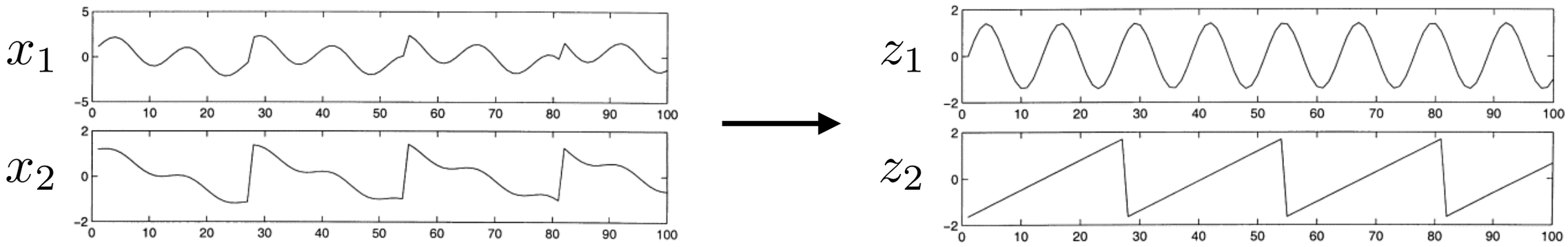
use more complex approximate posterior, but evaluate a simpler distribution

Normalizing Flows, Rezende & Mohamed, 2015

brief history

Independent Components Analysis (ICA)

Hyvarinen & Oja, 2000

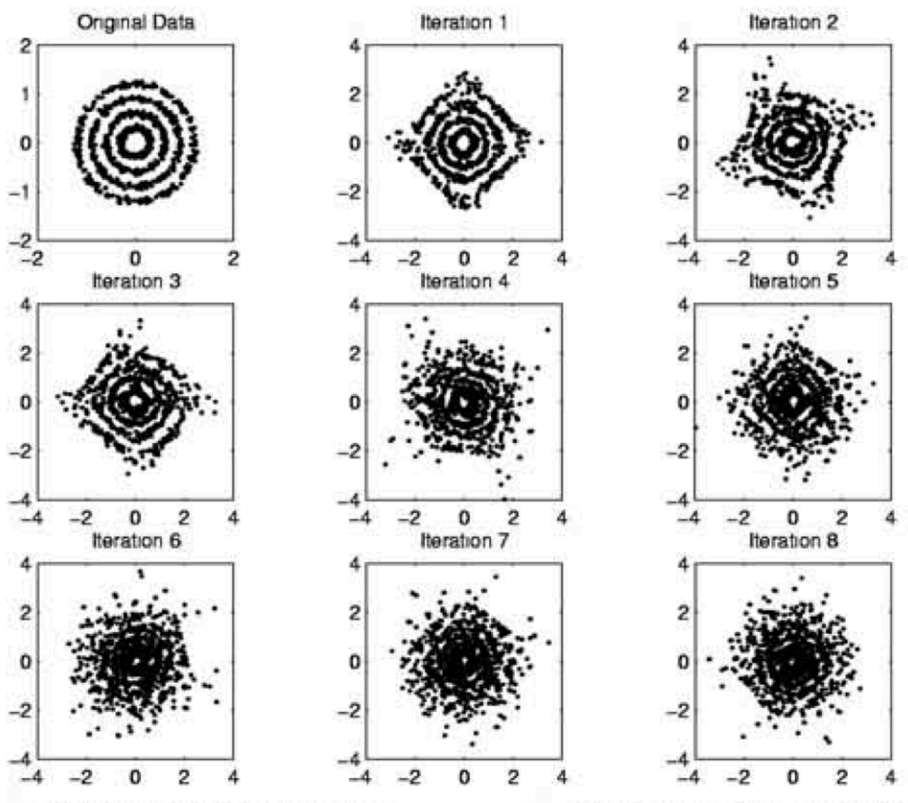


(linearly) project data to a space with least mutual information between dimensions

Iterative Gaussianization

Chen & Gopinath, 2001; Laparra et al., 2011

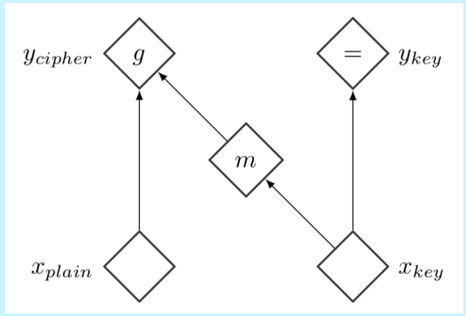
```
for k in [1, …, K]:

    apply ICA

    apply Gaussianization transform
```

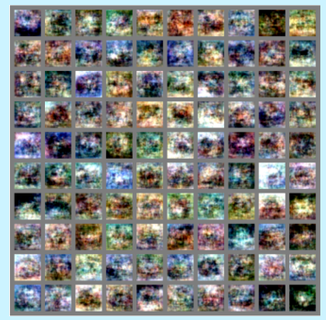


see also Deco & Brauer, 1995

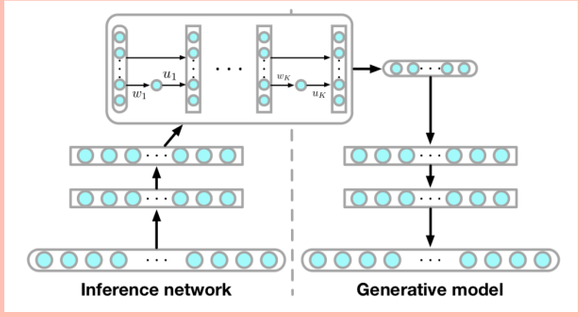
some recent work

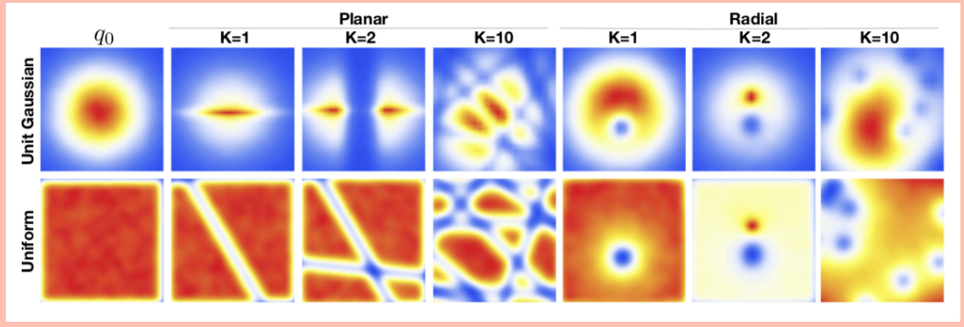
NICE: Non-linear Independent Components Estimation, Dinh *et al.*, 2014

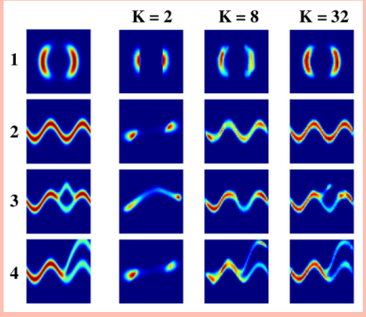




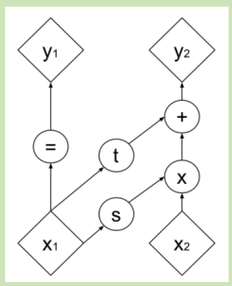
Variational Inference with Normalizing Flows, Rezende & Mohamed, 2015

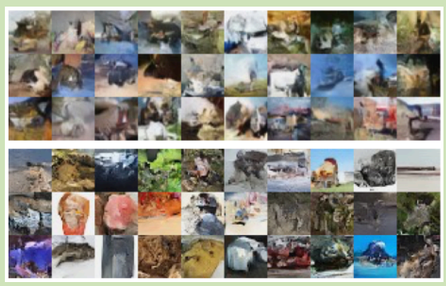




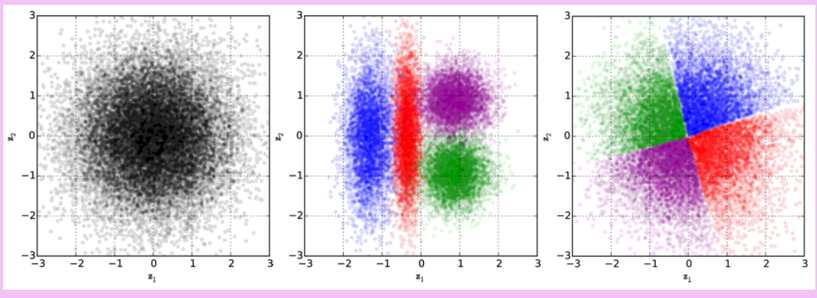


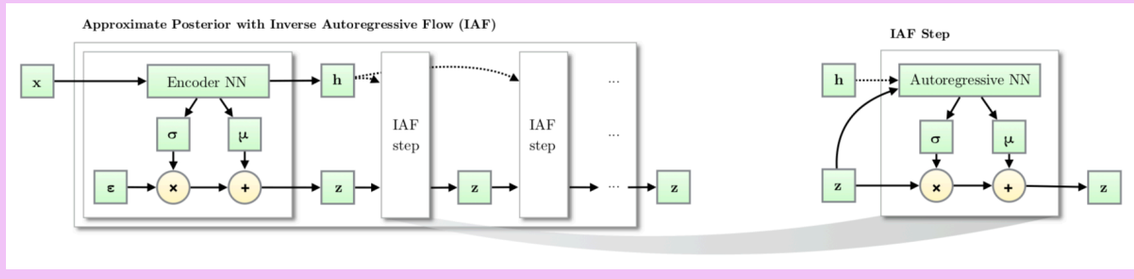
Density Estimation Using Real NVP, Dinh *et al.*, 2016





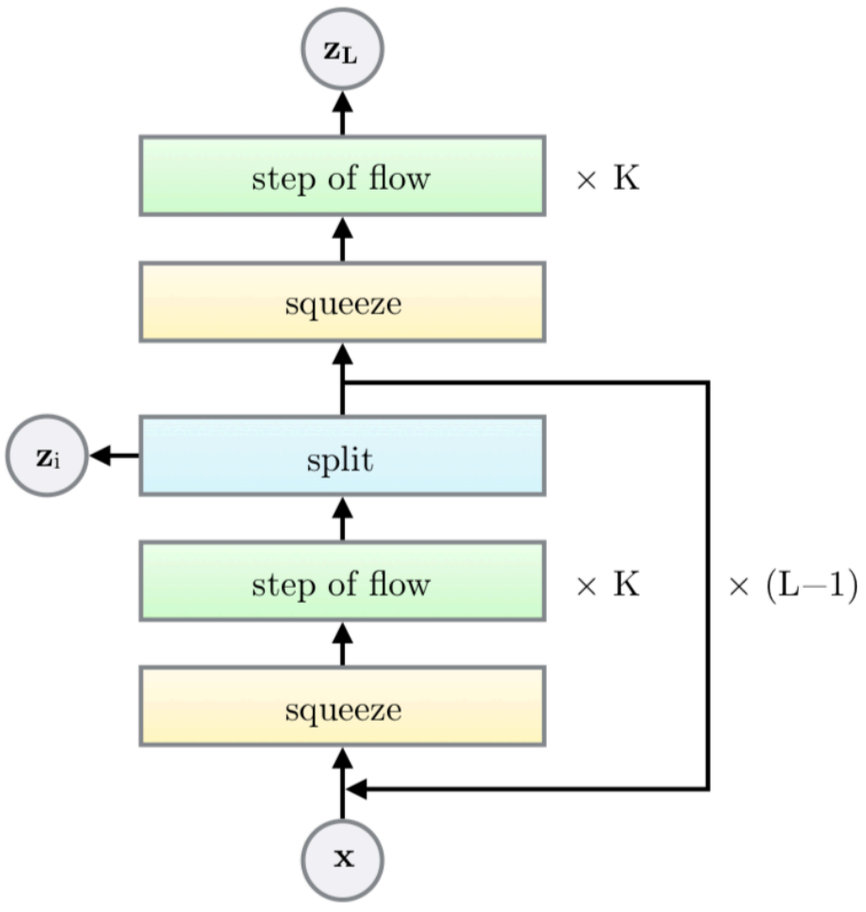
Improving Variational Inference with Inverse Autoregressive Flow, Kingma *et al.*, 2016





Glow

use 1 x 1 convolutions to perform transform



Glow, Kingma & Dhariwal, 2018

discrete flows

we can also apply invertible transforms to discrete data/variables

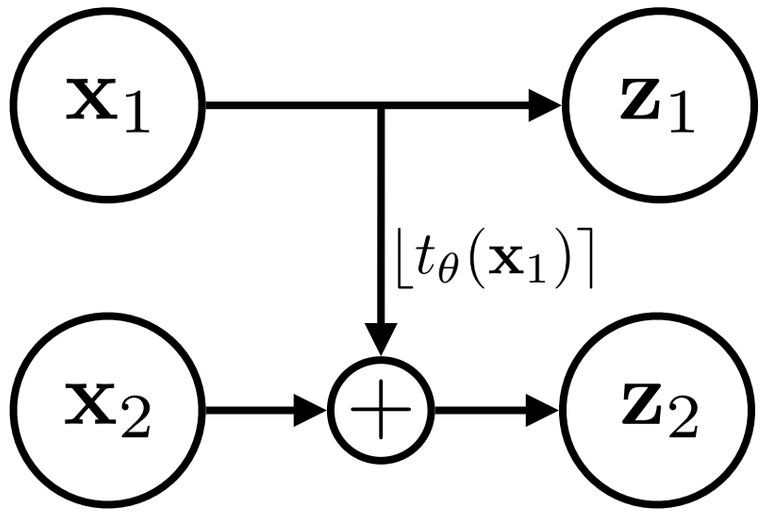
integer discrete flows

Hoogeboom et al., 2019

$$\mathbf{z}_1 = \mathbf{x}_1$$

$$\mathbf{z}_2 = \mathbf{x}_2 + \lfloor t_\theta(\mathbf{x}_1) \rceil$$

$\lfloor t_\theta(\mathbf{x}_1) \rceil$ denotes nearest-neighbor rounding



problem: rounding is not differentiable

solution: use the *straight-through estimator (Bengio et al., 2013)*

ignore rounding: $\nabla_{\mathbf{y}} \lfloor \mathbf{y} \rceil \triangleq \mathbf{I}$

see also Tran et al., 2019

Continuous Invertible Models

parameterize the transform as an ordinary differential equation (ODE)

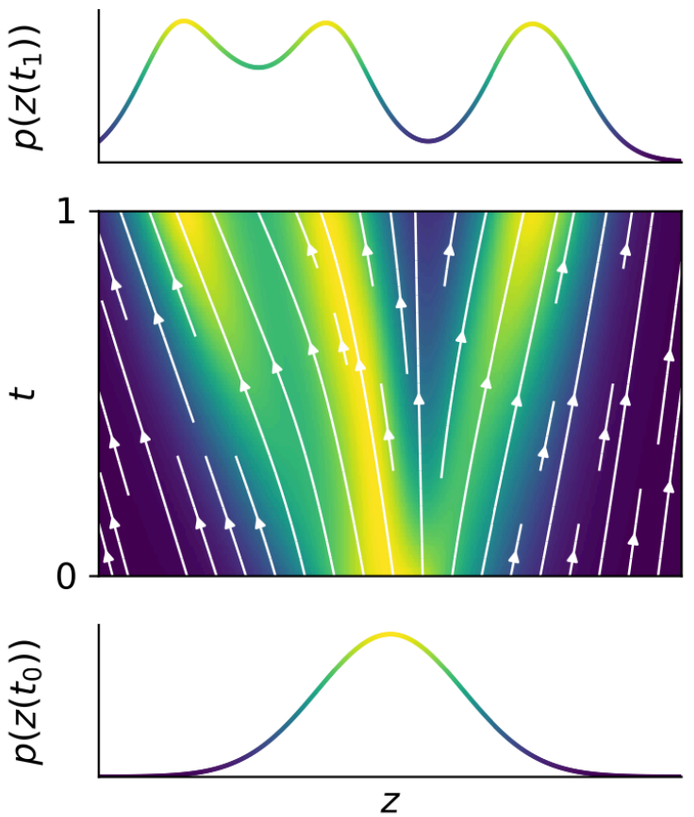
transform is a function of 'time'

$$f(\mathbf{z}(t), t; \theta)$$

instantaneous change of variables

$$\log p_\theta(\mathbf{z}(t_1)) = \log p_\theta(\mathbf{z}(t_0)) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial f}{\partial \mathbf{z}(t)}\right) dt$$

transformed distribution

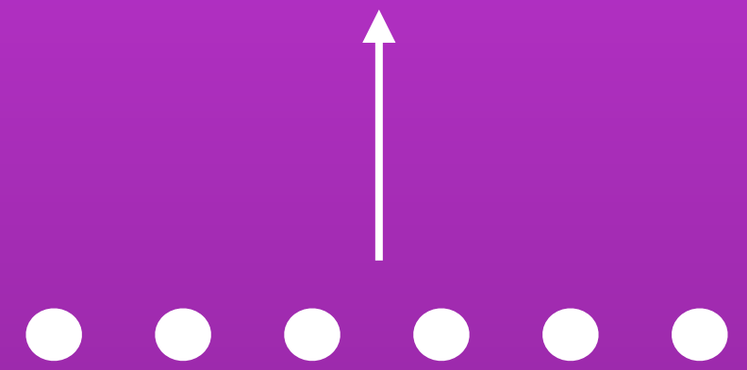


base distribution

Neural ODEs, Chen *et al.*, 2018

FFJORD, Grathwohl, Chen *et al.*, 2019

see also **Continuous-Time Flows**, Chen *et al.*, 2017



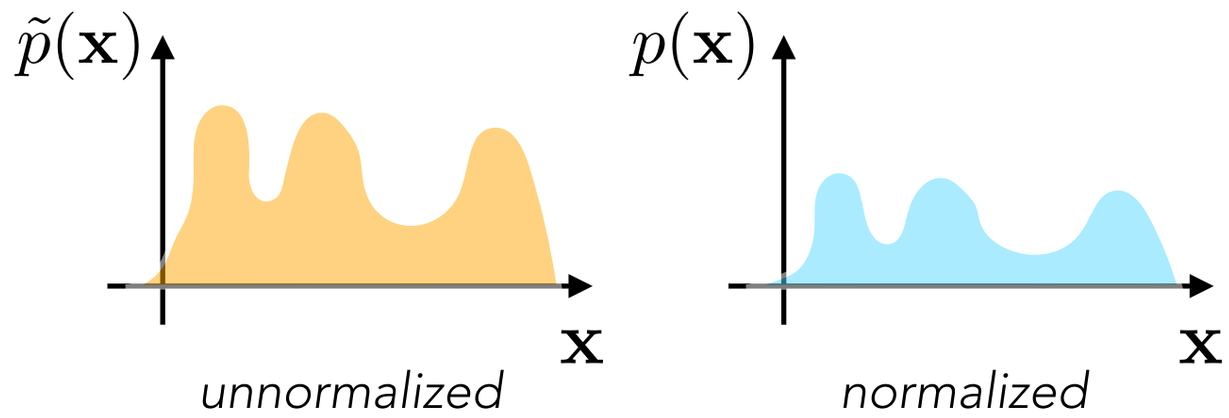
energy-based models

energy-based models

We can express a probability distribution in terms of an *unnormalized* distribution

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

(partition function) $$Z = \int \tilde{p}(\mathbf{x}) d\mathbf{x}$$



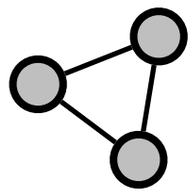
energy-based models (EBMs) define the unnormalized density as

$$\tilde{p}_\theta(\mathbf{x}) = \exp(-E_\theta(\mathbf{x}))$$ (Boltzmann)

$E_\theta(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}$ is an '*energy' function*

low energy ⟶ *high probability*

this is a special case of an undirected graphical model



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 energy-based models:

$$\log p_\theta(\mathbf{x}) = \log \left(\frac{1}{Z_\theta} \exp(-E_\theta(\mathbf{x})) \right)$$

$$= -\log Z_\theta - E_\theta(\mathbf{x})$$

Z_θ is typically computationally intractable

approximate maximum likelihood

estimate the log-likelihood gradient through sampling

$$
\begin{aligned}
\nabla_\theta \log p_\theta(\mathbf{x}) &= \nabla_\theta \left(-\log Z_\theta - E_\theta(\mathbf{x})\right) \\
&= -\frac{1}{Z_\theta}\nabla_\theta Z_\theta - \nabla_\theta E_\theta(\mathbf{x}) \\
&= -\frac{1}{Z_\theta}\nabla_\theta \int \exp(-E_\theta(\hat{\mathbf{x}}))d\hat{\mathbf{x}} - \nabla_\theta E_\theta(\mathbf{x}) \\
&= \frac{1}{Z_\theta}\int \exp(-E_\theta(\hat{\mathbf{x}}))\nabla_\theta E_\theta(\hat{\mathbf{x}})d\hat{\mathbf{x}} - \nabla_\theta E_\theta(\mathbf{x}) \\
&= \underbrace{\mathbb{E}_{\hat{\mathbf{x}}\sim p_\theta}\left[\nabla_\theta E_\theta(\hat{\mathbf{x}})\right]}_{\text{model}} - \underbrace{\nabla_\theta E_\theta(\mathbf{x})}_{\text{data}}
\end{aligned}
$$

if we follow $\nabla_\theta \log p_\theta(\mathbf{x})$*, then we* ***decrease*** *the energy for <u>data</u> examples and* ***increase*** *the energy for <u>model</u> samples*

Hinton, 2002

MCMC sampling

We can estimate $\mathbb{E}_{\mathbf{x}\sim p_\theta}\left[\nabla_\theta E_\theta(\mathbf{x})\right]$ *using various sampling techniques.*

stochastic gradient Langevin dynamics

$$\hat{\mathbf{x}}^k \leftarrow \hat{\mathbf{x}}^{k-1} - \frac{\lambda}{2}\nabla_{\mathbf{x}} E_\theta(\hat{\mathbf{x}}^{k-1}) + \boldsymbol{\omega}^k$$

where $\boldsymbol{\omega}^k \sim \mathcal{N}(\mathbf{0}, \lambda\mathbf{I})$

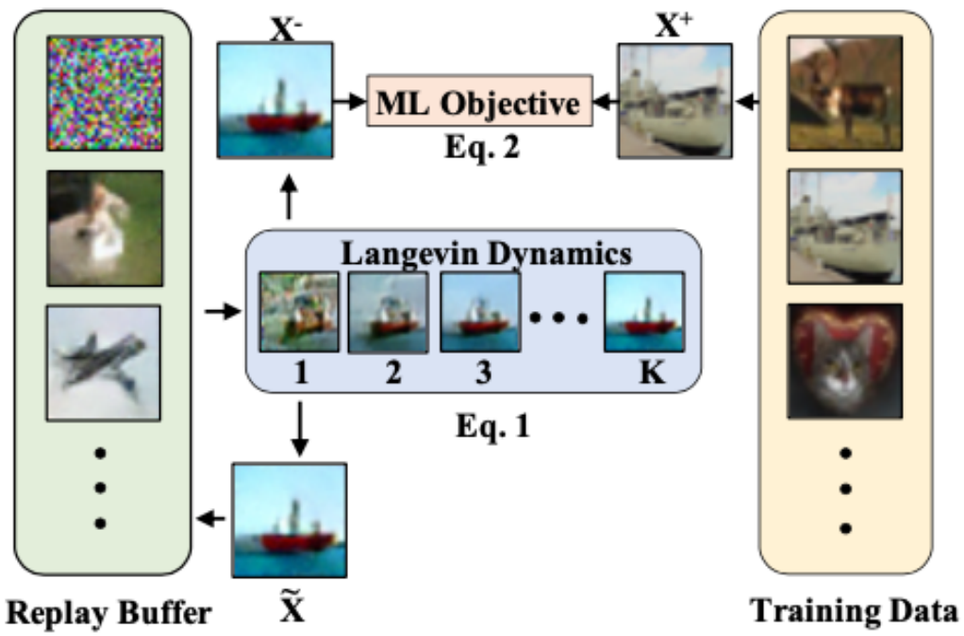
descend the energy function with injected Gaussian noise

drawback: performance depends on sampler quality and sampling tricks.

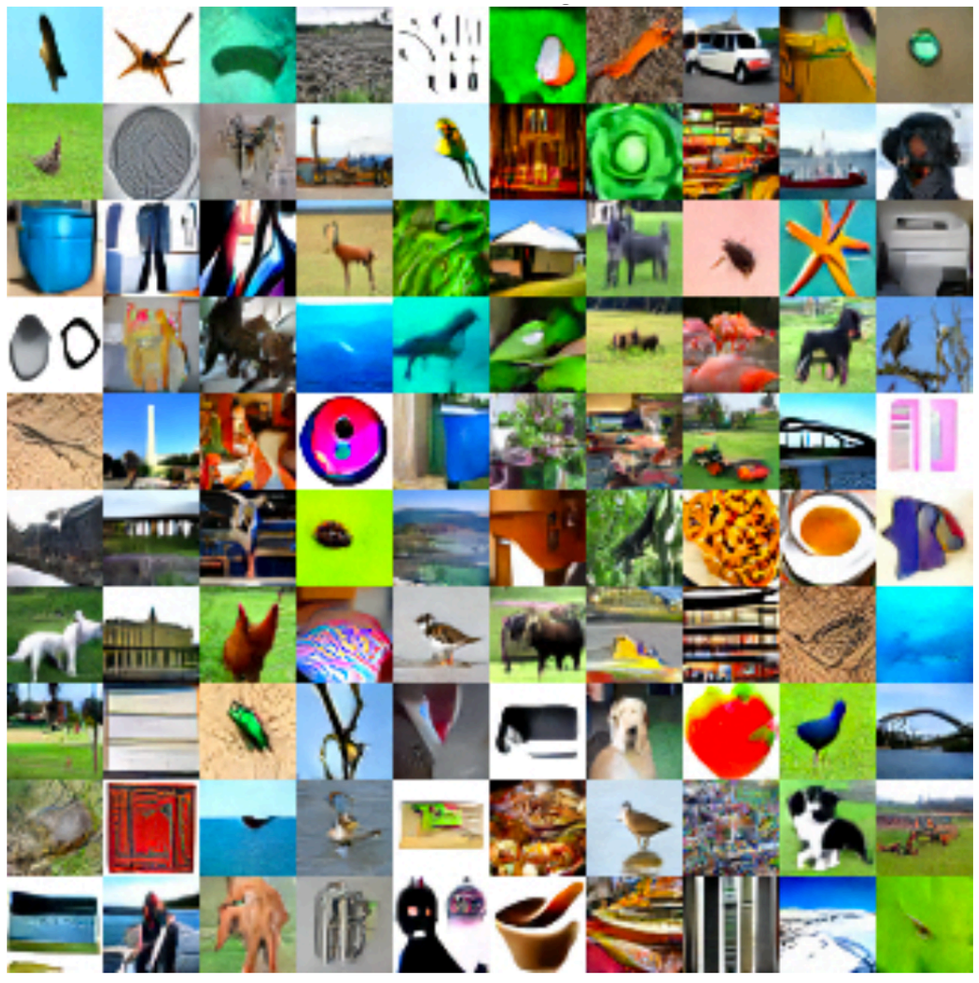
Welling & Teh, 2011

Du & Mordatch, 2019

deep energy-based models



Training pipeline



Samples (trained on ImageNet 32x32)

Du & Mordatch, 2019

EBMs + latent variables

Can also introduce latent variables into EBMs:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \frac{1}{Z_\theta} \int \exp(-E_\theta(\mathbf{x}, \mathbf{z})) d\mathbf{z}$$

Define $\mathcal{F}_\theta(\mathbf{x}) = -\log \int \exp(-E_\theta(\mathbf{x}, \mathbf{z})) d\mathbf{z}$.

Then $p_\theta(\mathbf{x}) = \frac{1}{Z_\theta} \exp(-\mathcal{F}_\theta(\mathbf{x}))$.

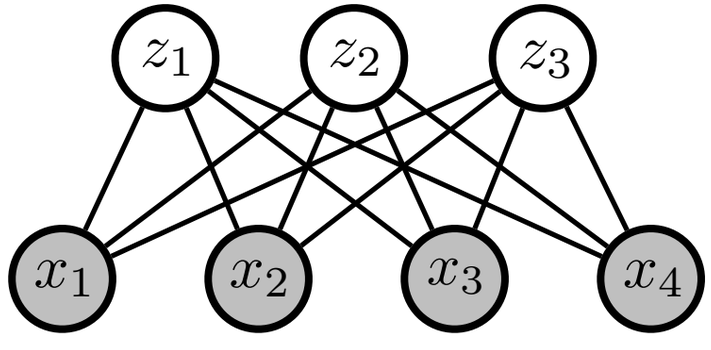
Marginal energy-based model with the
'free energy', $\mathcal{F}_\theta(\mathbf{x})$, given by the log-sum-exp of the joint energy.

restricted Boltzmann machines (RBMs)

structure

restricted Boltzmann machines consist of observed variables $\mathbf{x}$ and latent variables $\mathbf{z}$

with connections *restricted* to a bipartite graph:



functional form

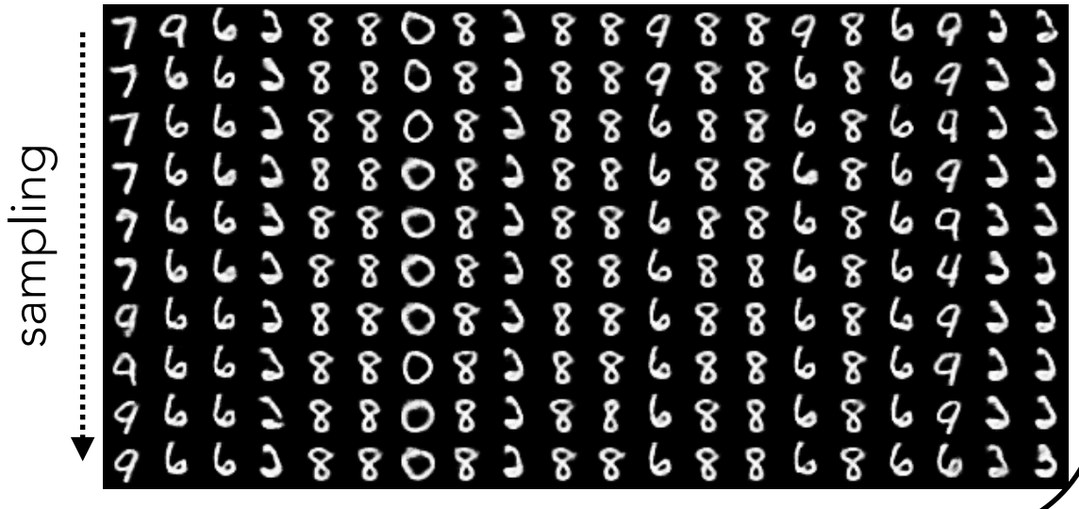
define the energy function as

$$E_\theta(\mathbf{x}, \mathbf{z}) = -\mathbf{b}^\intercal \mathbf{x} - \mathbf{c}^\intercal \mathbf{z} - \mathbf{x}^\intercal \mathbf{W} \mathbf{z}$$

where $\mathbf{b}, \mathbf{c}, \mathbf{W}$ are learnable parameters

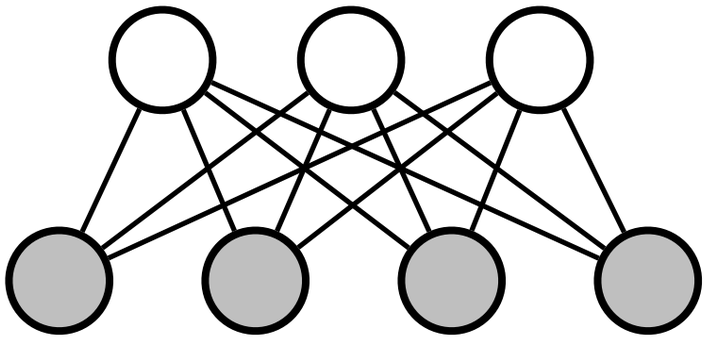
training

typically trained using block Gibbs sampling

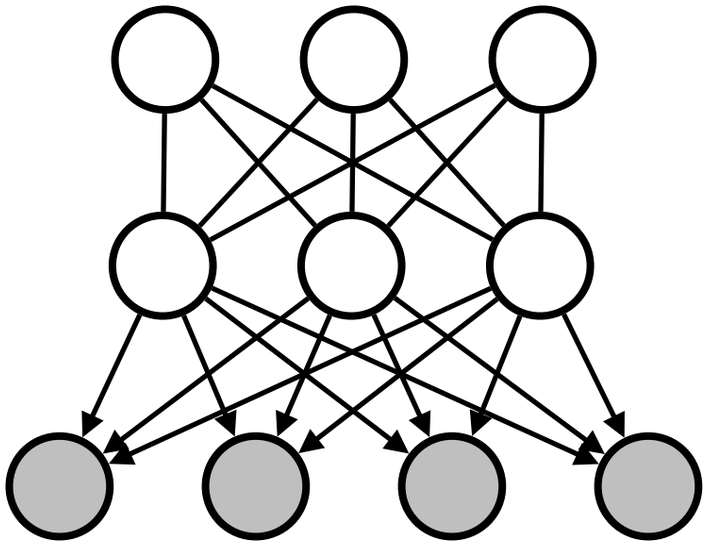


Deep Learning (Chapter 16), *Goodfellow et al.*, 2016

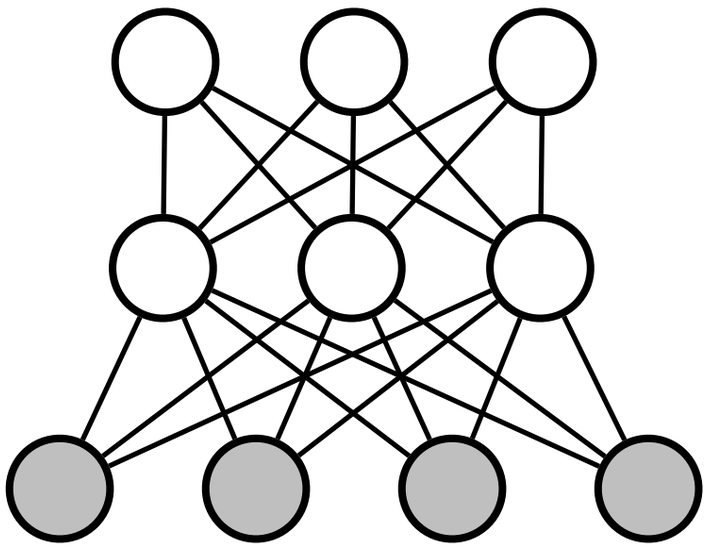
deep energy-based models



Restricted Boltzmann Machine



Deep Belief Network



Deep Boltzmann Machine

Deep Learning (Chapter 20), *Goodfellow et al.*, 2016

classifiers as energy-based models

Classifiers are often trained using $p_\theta(y|\mathbf{x}) = \dfrac{\exp(f_\theta(\mathbf{x})[y])}{\sum_{y'} \exp(f_\theta(\mathbf{x})[y'])}$

Can interpret Softmax logits as joint *energies*

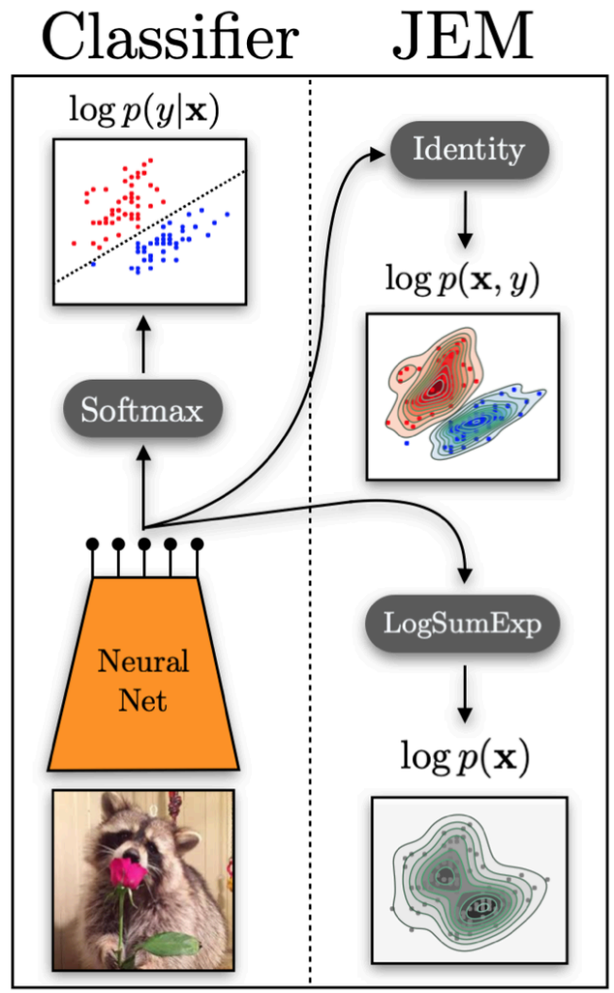
$$E_\theta(\mathbf{x}, y) = -f_\theta(\mathbf{x})[y]$$

Marginalize over classes to get a generative model

$$p_\theta(\mathbf{x}) = \sum_y p_\theta(\mathbf{x}, y) = \frac{1}{Z_\theta} \sum_y \exp(f_\theta(\mathbf{x})[y])$$

Can jointly train classifier and generative model

$$\log p_\theta(\mathbf{x}, y) = \log p_\theta(\mathbf{x}) + \log p_\theta(y|\mathbf{x})$$

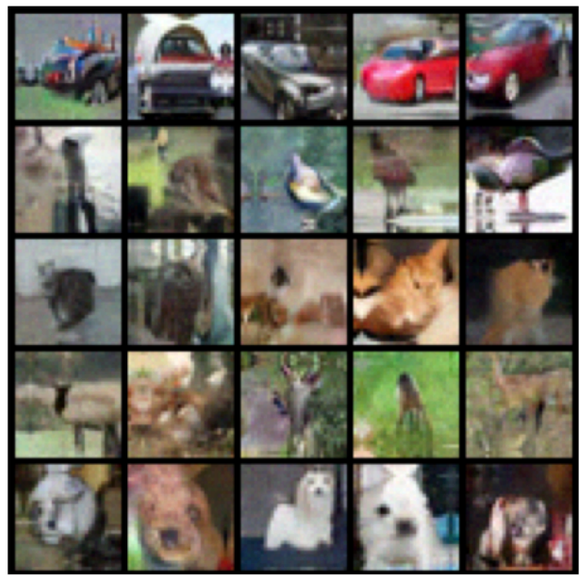


Grathwohl et al., 2020

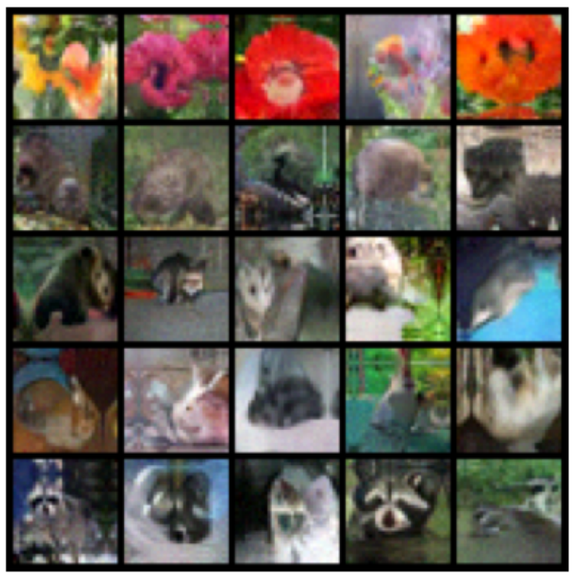
classifiers as energy-based models

samples:

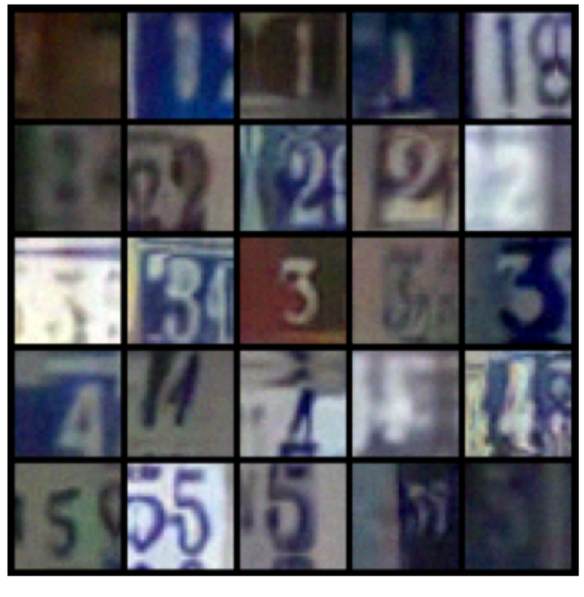
CIFAR-10

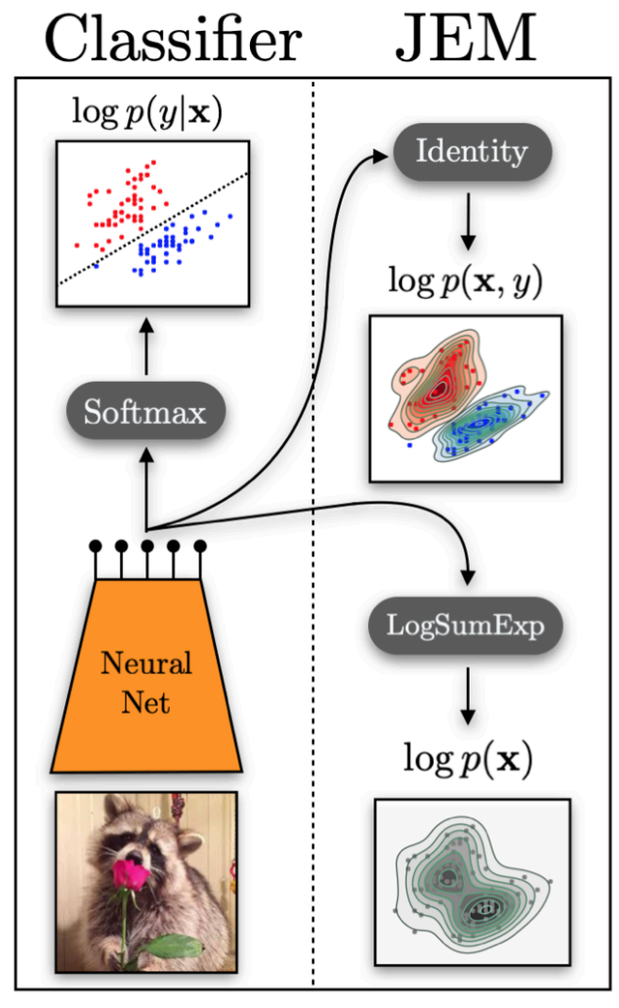


CIFAR-100



SVHN





Grathwohl et al., 2020

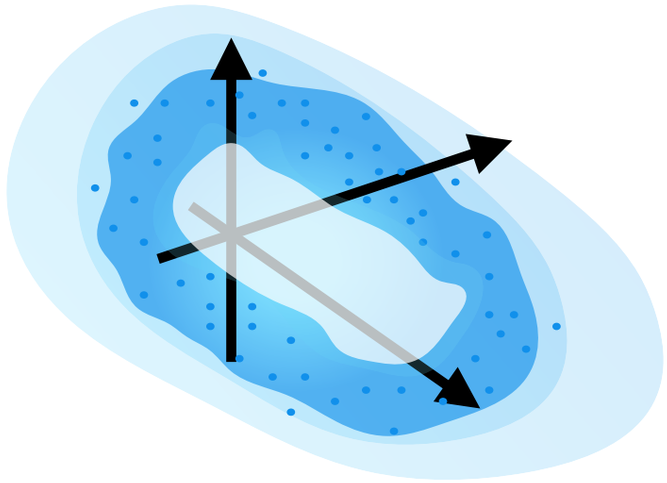
DISCUSSION

summary

generative model: a model of the data distribution

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

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

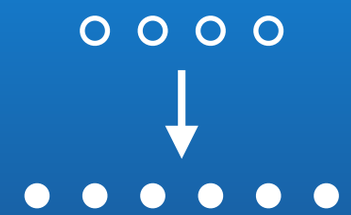


maximum likelihood (minimum KL)

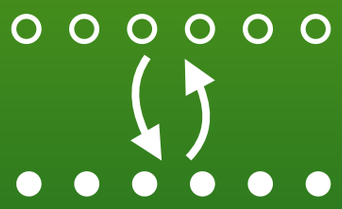
$$\theta^* = \arg\min_\theta D_{KL}(p_{\text{data}}(\mathbf{x})||p_\theta(\mathbf{x}))$$

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

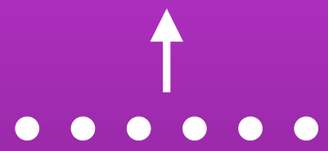
autoregressive models



explicit latent variable models



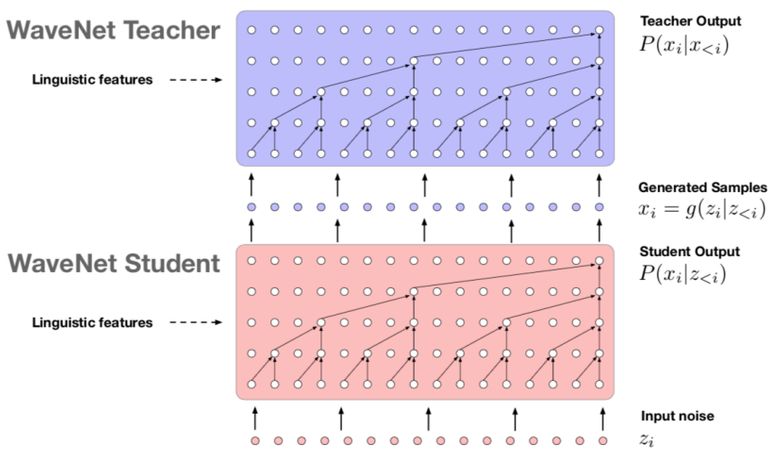
invertible explicit latent variable models



energy-based models

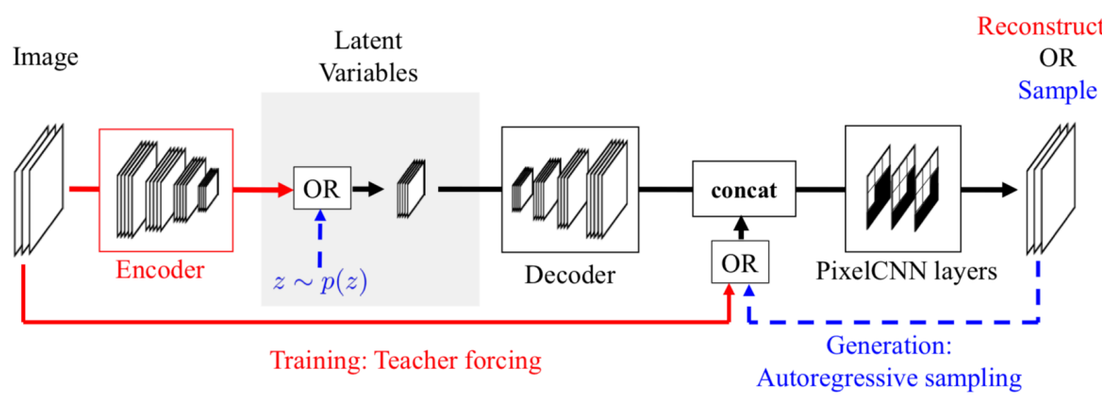
combining models

autoregressive + invertible model



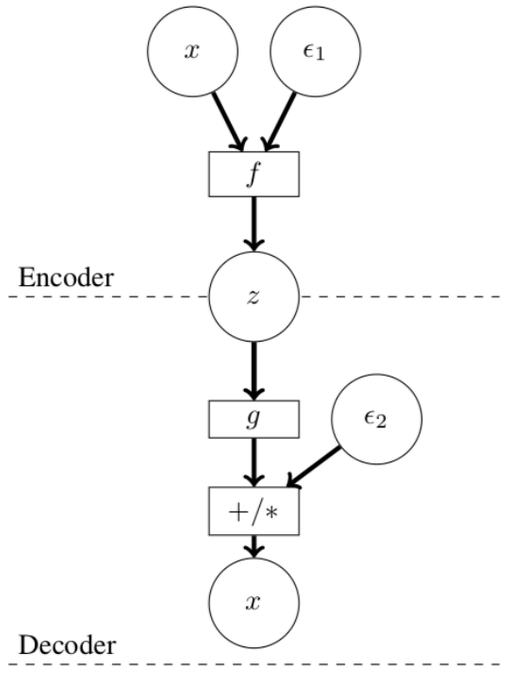
Parallel WaveNet, van den Oord *et al.*, 2018

autoregressive + explicit latent variable model



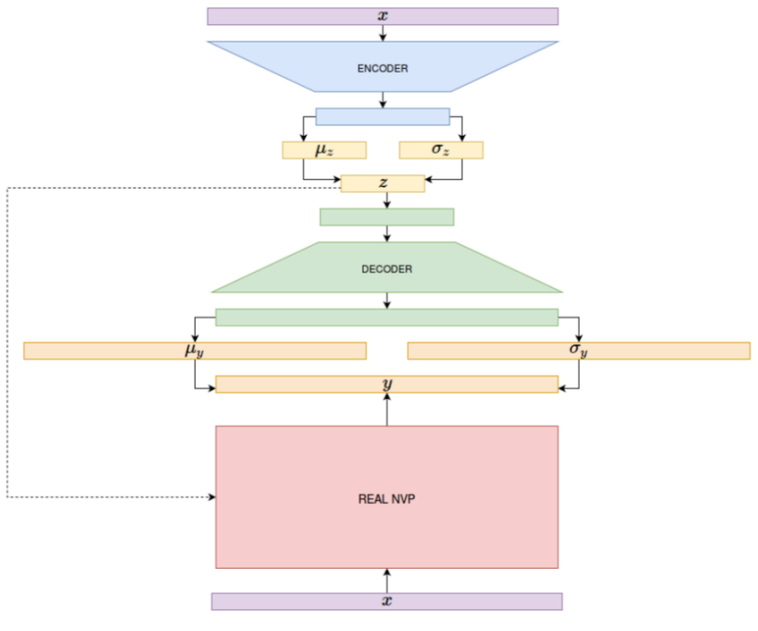
PixelVAE, Gulrajani *et al.*, 2017

explicit + implicit latent variable model



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

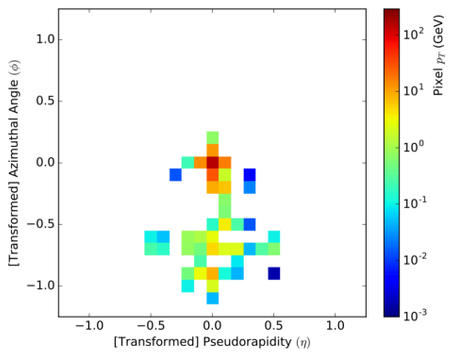
explicit + invertible latent variable model

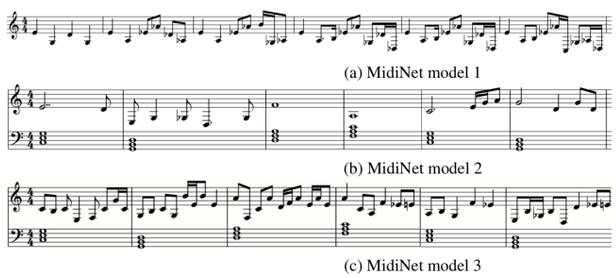


Deep Variational Inference Without Pixel-Wise Reconstruction,
Agrawal & Dukkipati, 2016

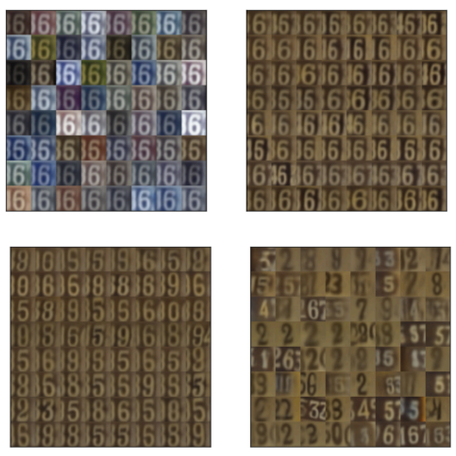
what are generative models good for?

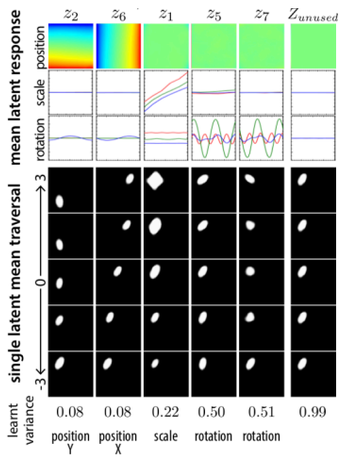
1. can generate and simulate data





2. can extract structure from data

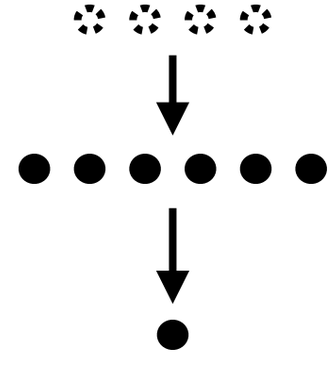




other ways of parameterizing generative models

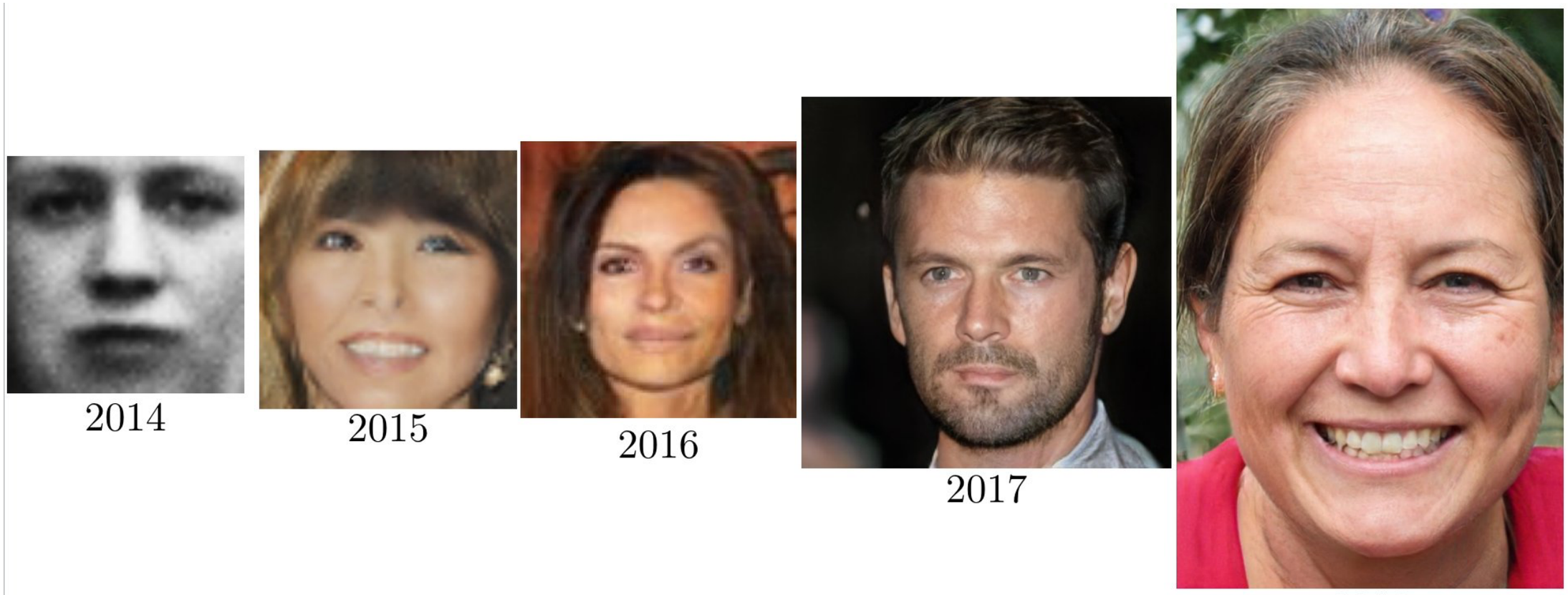
implicit models (e.g. GANs):

model is defined in terms of an implicit sampling procedure



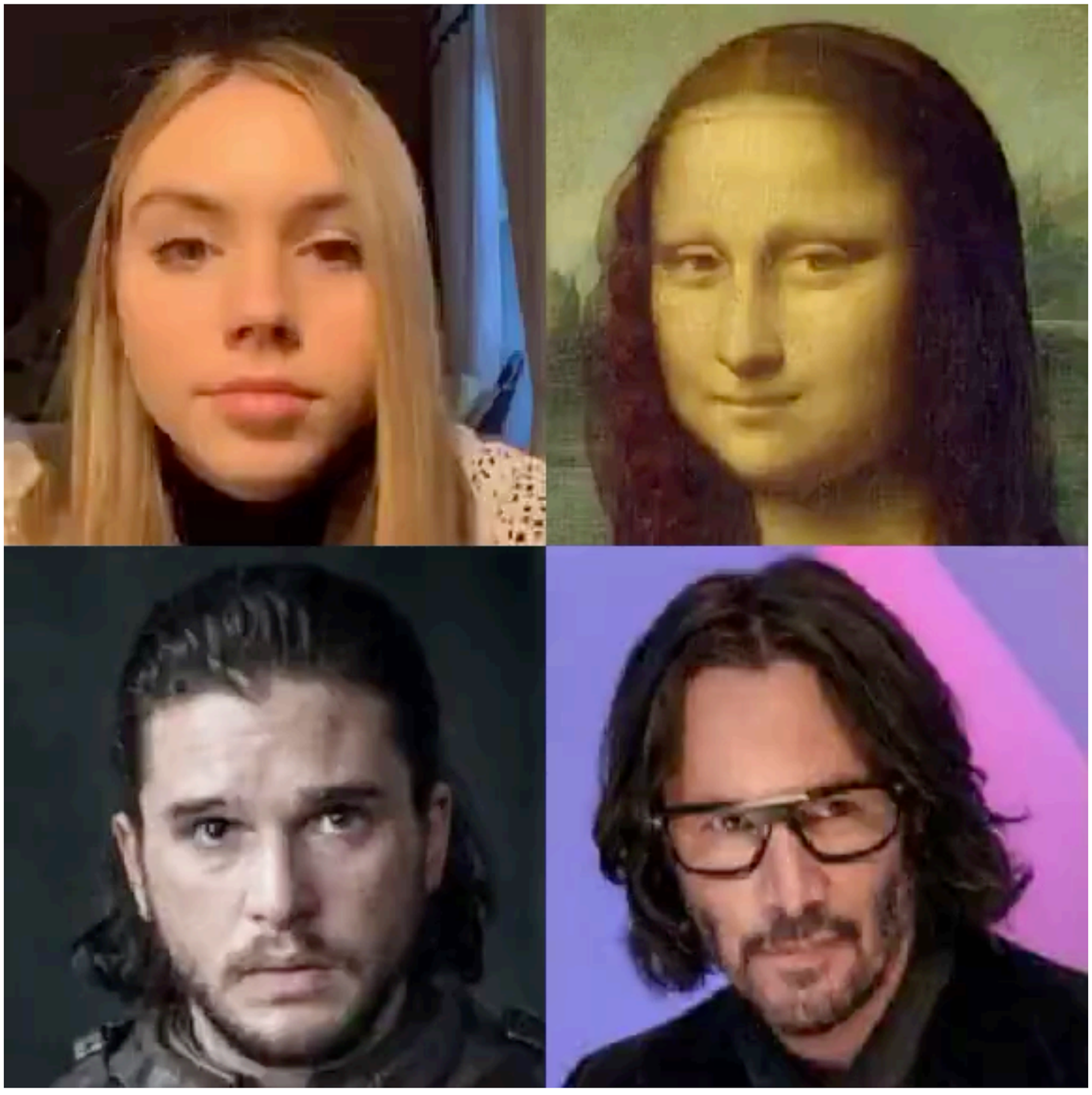
other training objectives

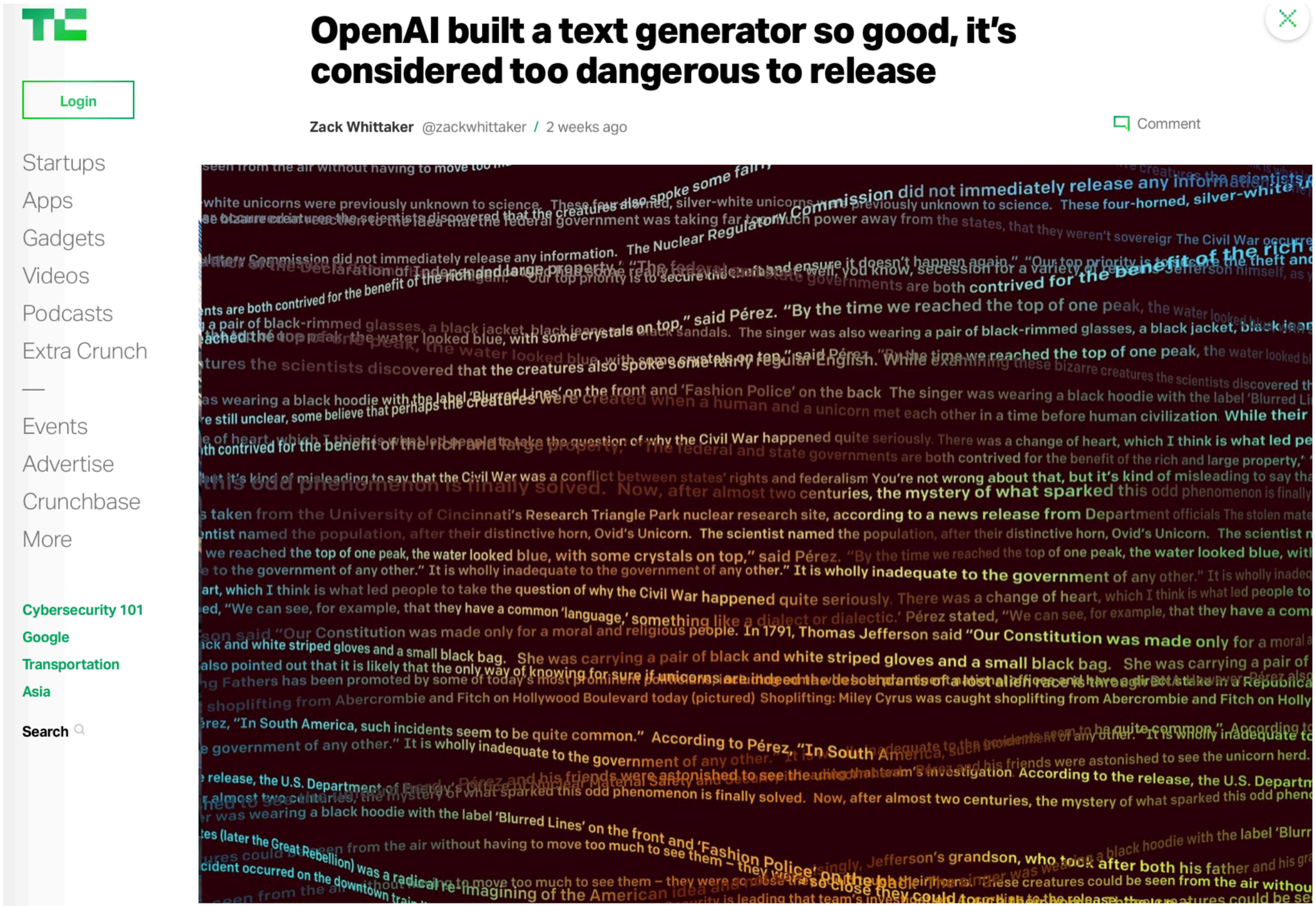
Jensen-Shannon Divergence, Wasserstein distance



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

StyleGAN, Karras *et al.*, 2018





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