Deep Generative Models

杨 帆

fanyang01@zju.edu.cn

2019年6月20日

Outline

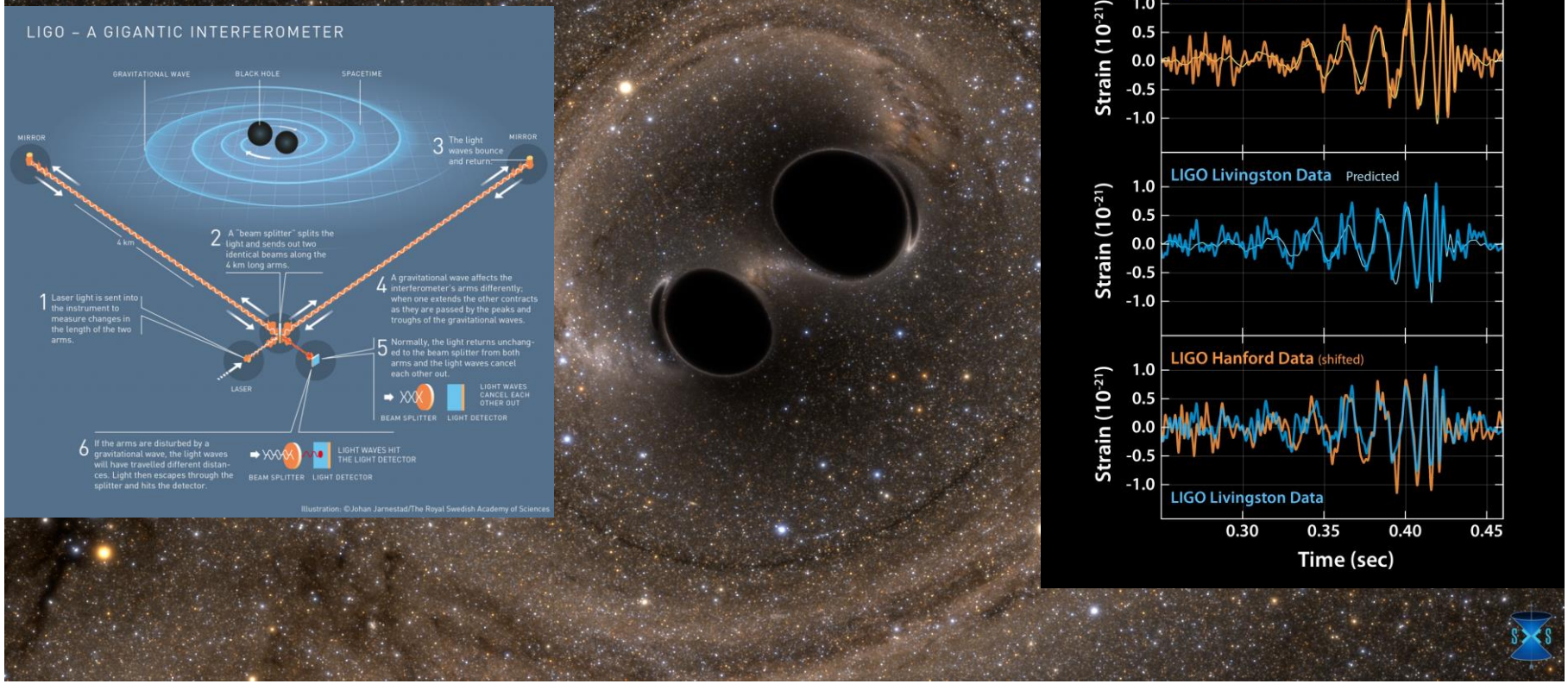
- Deep Generative Models
  - What; Why; Landscape of DGMs; Sequential DGMs

Outline

- **Deep Generative Models**
  - **What; Why;** Landscape of DGMs; Sequential DGMs

The Nature of Data

- All data were collected by observing some generating processes

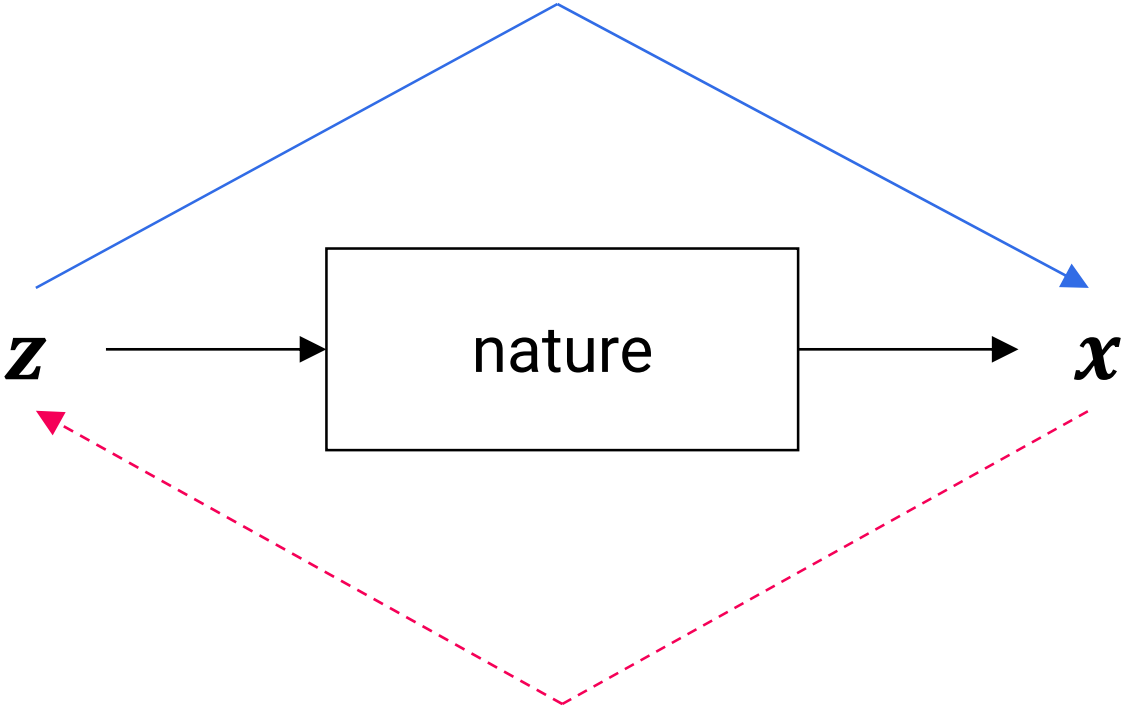


Discriminative vs Generative

- Generative model

$$\boldsymbol{x} = g(\boldsymbol{z}, \boldsymbol{\varepsilon}; \boldsymbol{\theta})$$

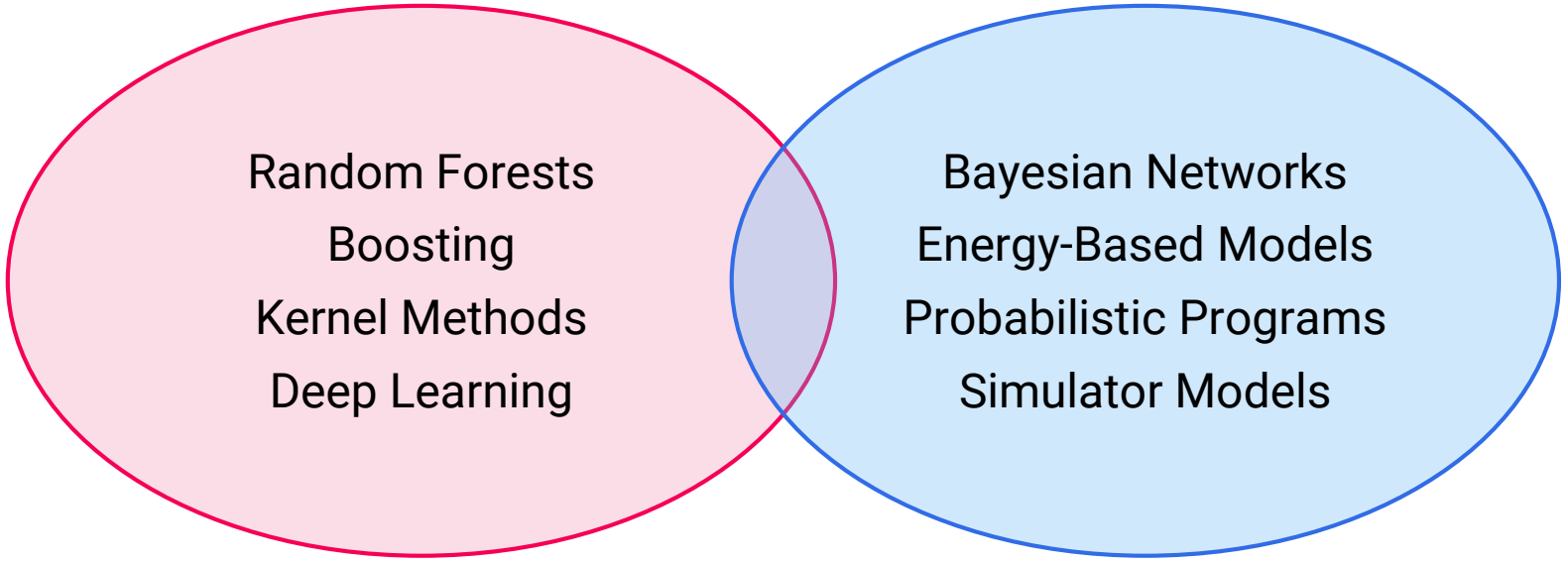
- Data generating process



- Discriminative model

$$\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\theta})$$

Discriminative vs Generative



- Flexible map from input to target
- Efficient training algorithms
- Very successful and accurate
- Good at **inference**
- ML: the CS version?
- Alchemy?

- Model causal relations and physics
- Used by most scientific disciplines
- Interpretable and data efficient
- Good at **prediction**
- ML: the STAT version?
- True science?

Statistical Science
2001, Vol. 16, No. 3, 199–231

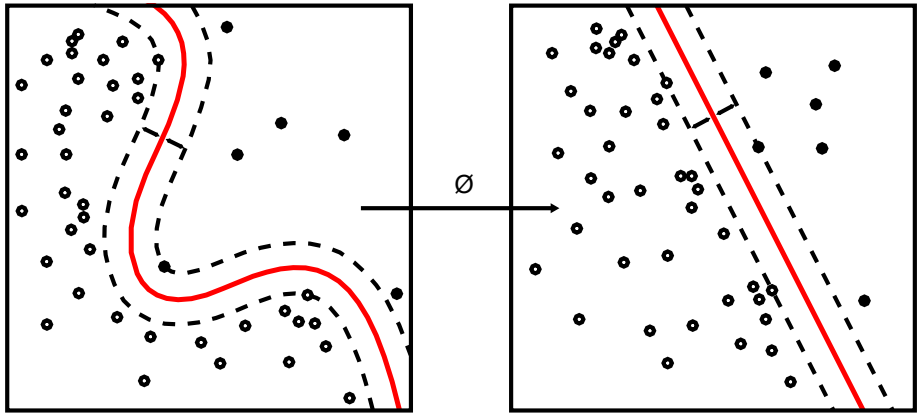
2001

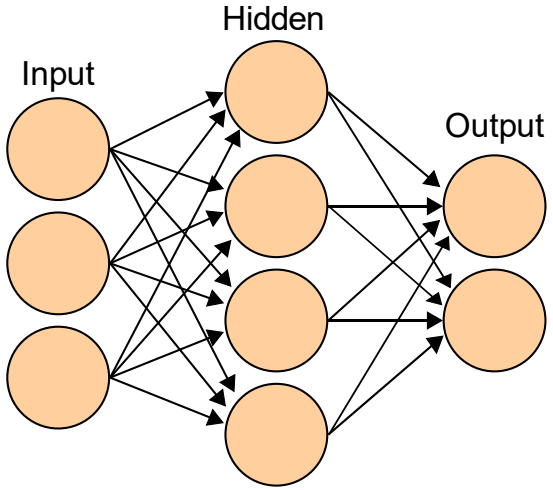
Statistical Modeling: The Two Cultures

Leo Breiman

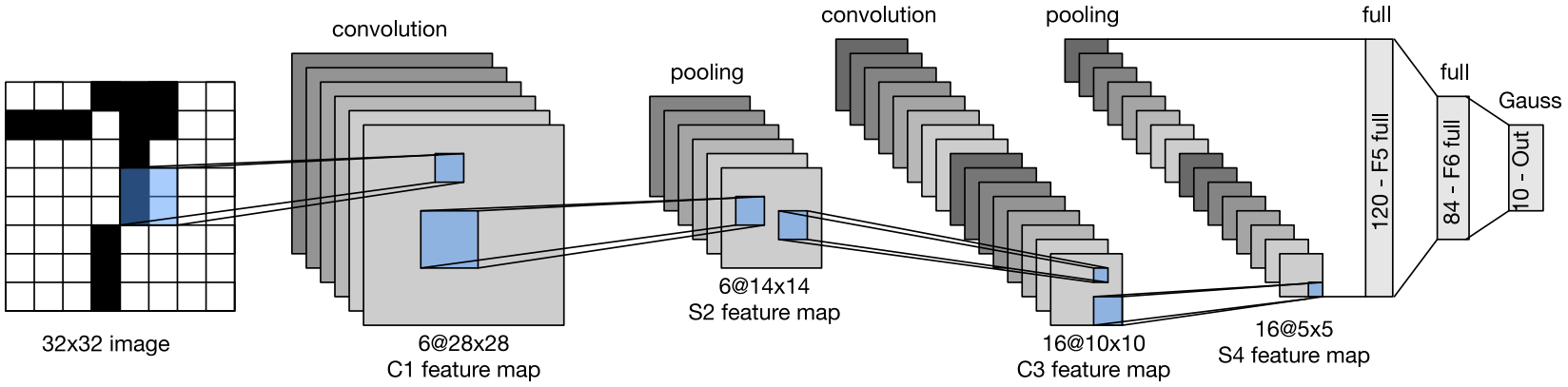
Abstract. There are two cultures in the use of statistical modeling to reach conclusions from data. One assumes that the data are generated by a given stochastic data model. The other uses algorithmic models and treats the data mechanism as unknown. The statistical community has been committed to the almost exclusive use of data models. This commitment has led to irrelevant theory, questionable conclusions, and has kept statisticians from working on a large range of interesting current problems. Algorithmic modeling, both in theory and practice, has developed rapidly in fields outside statistics. It can be used both on large complex data sets and as a more accurate and informative alternative to data modeling on smaller data sets. If our goal as a field is to use data to solve problems, then we need to move away from exclusive dependence on data models and adopt a more diverse set of tools.

1990 ~ 2010





2010 ~ Now



2014 ~ : The Rise of Probabilistic/Bayesian DL

Auto-Encoding Variational Bayes

Diederik P. Kingma
Machine Learning Group
Universiteit van Amsterdam
dpkingma@gmail.com

Max Welling
Machine Learning Group
Universiteit van Amsterdam
welling.max@gmail.com

Abstract

How can we perform efficient inference and learning in directed probabi models, in the presence of continuous latent variables with intractable post distributions, and large datasets? We introduce a stochastic variational infer and learning algorithm that scales to large datasets and, under some mild d entiability conditions, even works in the intractable case. Our contributio two-fold. First, we show that a reparameterization of the variational lower b yields a lower bound estimator that can be straightforwardly optimized using dard stochastic gradient methods. Second, we show that for i.i.d. datasets continuous latent variables per datapoint, posterior inference can be made cially efficient by fitting an approximate inference model (also called a rec tion model) to the intractable posterior using the proposed lower bound estin Theoretical advantages are reflected in experimental results.

Generative Adversarial Nets

Ian J. Goodfellow[*], **Jean Pouget-Abadie**[†], **Mehdi Mirza**, **Bing Xu**, **David Warde-Farley**, **Sherjil Ozair**[‡], **Aaron Courville**, **Yoshua Bengio**[§]
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

Abstract

We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to $\frac{1}{2}$ everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples. Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of the generated samples.

Why should we care?

ML as a bag of tricks

Fast special cases:

- K-means
- Kernel Density Estimation
- SVMs
- Boosting
- Random Forests

Extensible family:

- Mixture of Gaussians
- Latent variable models
- Gaussian processes
- Deep neural nets
- Bayesian neural nets

Regularization as bag of tricks

Fast special cases:

- Early stopping
- Ensembling
- L2 Regularization
- Gradient noise
- Dropout

Extensible family:

- Stochastic variational inference

Losses are log-likelihoods

- Squared loss is just unnormalized Normal log-pdf
- "Cross-entropy" now means Categorical log-pmf ?!
  - Actual definition: $H(p, q) = -\sum_x p(x) \log q(x)$
- "Teacher forcing" is just evaluating the likelihood of a sequential model $p(x) = \prod_i p_\theta(x_i | x_{<i})$

AI as a bag of tricks

Russel and Norvig's parts of AI:

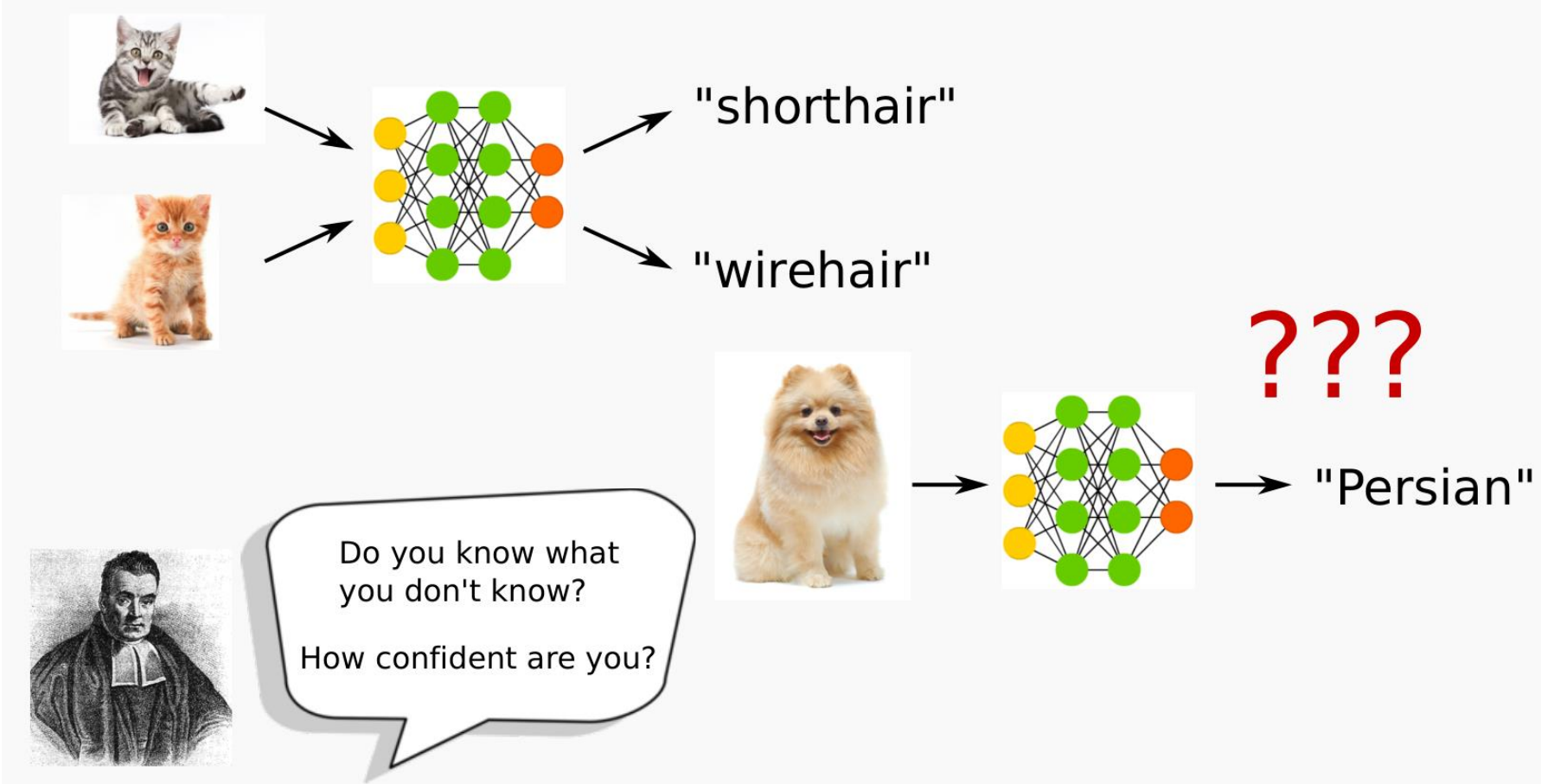
- Machine learning
- Natural language processing
- Knowledge representation
- Automated reasoning
- Computer vision
- Robotics

Extensible family:

- Deep probabilistic latent-variable models + decision theory

Generative Models. David Duvenaud. Deep Learning Summer School 2018

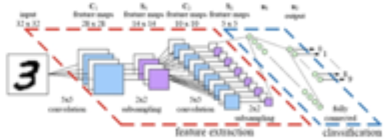
Why should we care?



Efficient Computation for Bayesian Deep Learning. Yingzhen Li. 2018

Why should we care?

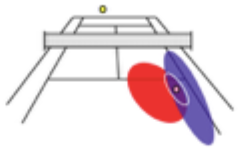
Two Streams of Machine Learning



Deep Learning

+ Rich non-linear models for classification and sequence prediction.
+ Scalable learning using stochastic approximation and conceptually simple.
+ Easily composable with other gradient-based methods.

- Only point estimates.
- Hard to score models, do selection and complexity penalisation.



Probabilistic Reasoning

- Mainly conjugate and linear models.
- Potentially intractable inference, computationally expensive or long simulation time.

+ Unified framework for model building, inference, prediction and decision making.
+ Explicit accounting for uncertainty and variability of outcomes.
+ Robust to overfitting; tools for model selection and composition.

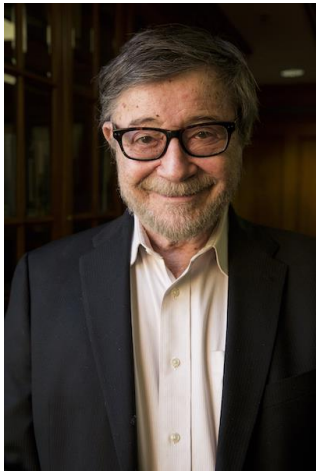
Complementary strengths, making it natural to combine them

Tutorial on Deep Generative Models. Shakir Mohamed and Danilo Rezende. UAI 2017 Tutorial

Bayesian Inference

Monte Carlo Sampling

Probabilistic Graphical Models



Bayesian NN

MLP, CNN, RNN, GNN

Variational Inference

Gradient Decent, Back-propagation

Deep Generative Models

Seq2Seq, Attention

Generative modeling is density estimation

Generative modeling is the art and science of engineering a family of probability distributions that is simultaneously ***rich, parsimonious, and tractable.***

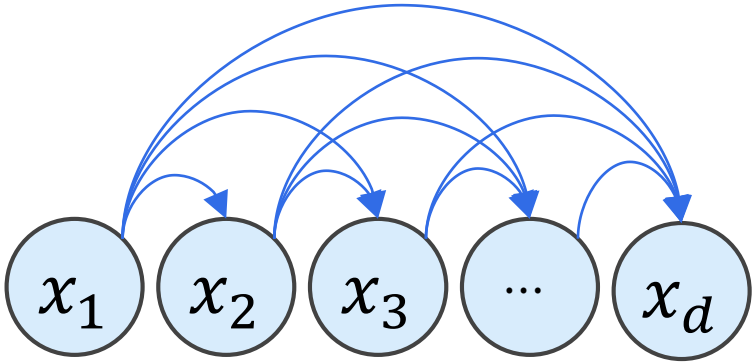
A Tutorial on Deep Probabilistic Generative Models. Ryan P. Adams. Machine Learning Summer School 2018

Outline

- **Deep Generative Models**
  - What; Why; **Landscape of DGMs**; Sequential DGMs

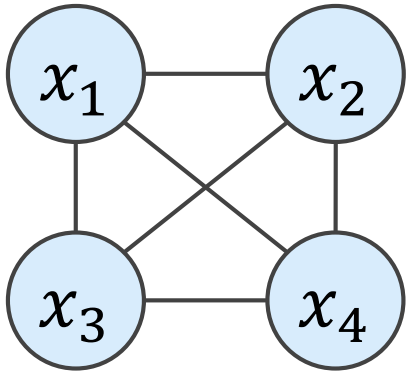
Landscape of (Deep) Generative Models

Autoregressive Models



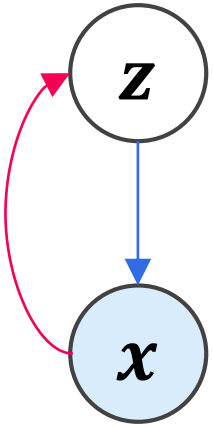
$$p_\theta(\boldsymbol{x}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i})$$

Energy-Based Models



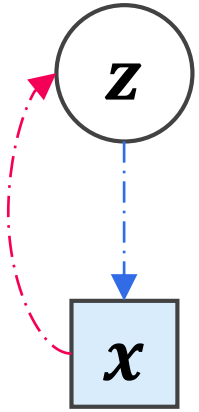
$$p_\theta(\boldsymbol{x}) = \frac{e^{-E_\theta(\boldsymbol{x})}}{\int e^{-E_\theta(\boldsymbol{x})}\, \mathrm{d}\boldsymbol{x}}$$

Latent Variable Models



$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}$$

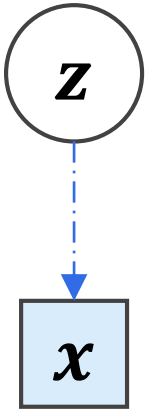
Normalizing Flows



$$\boldsymbol{x} = f_\theta(\boldsymbol{z})$$

$$p_\theta(\boldsymbol{x}) = p_\theta(\boldsymbol{z}) \left| \det \frac{\partial f_\theta^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right|$$

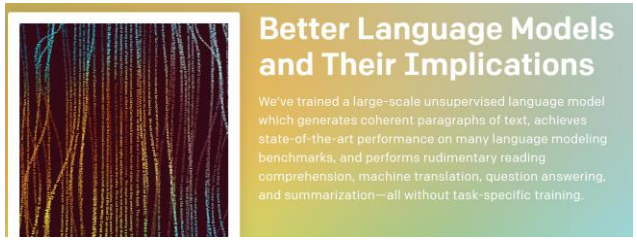
Implicit Models



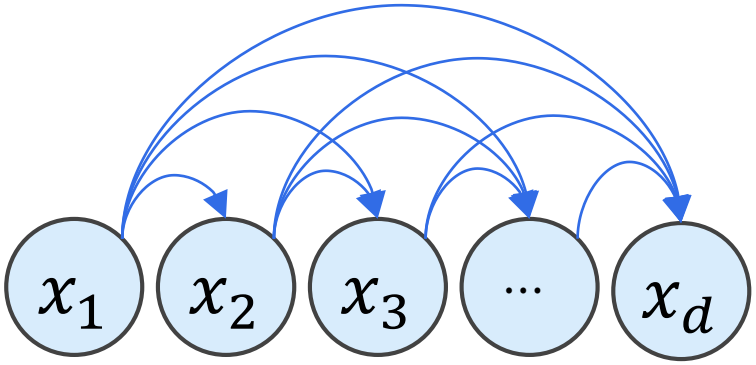
$$p_\theta(\boldsymbol{x}) = ?$$

Autoregressive Models

- Any distribution can be factorized into an AR form [***chain rule***]
- Exact likelihood, easy to train
- Very generic and powerful
- Modern NNs make the training parallelizable
  - WaveNet, Causal CNN, Transformer



https://openai.com/blog/better-language-models/



$$p_\theta(\boldsymbol{x}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i})$$

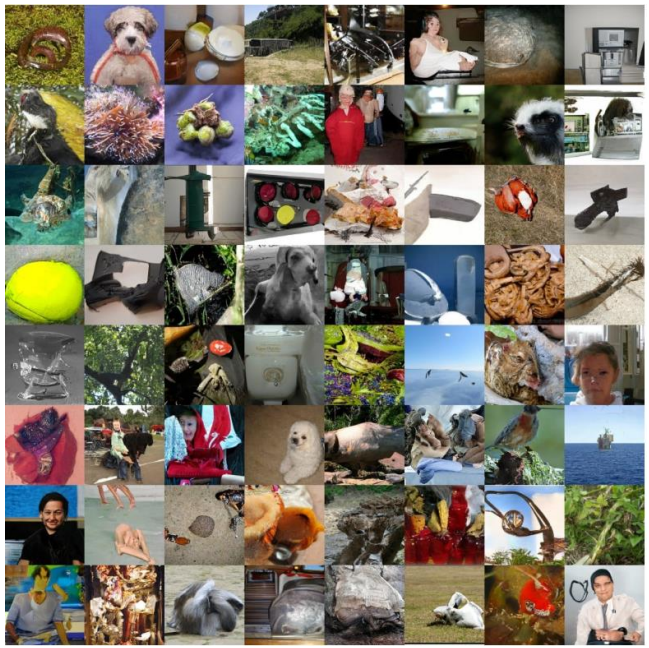
$$p_\theta(\boldsymbol{x}|\boldsymbol{c}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i}, \boldsymbol{c})$$

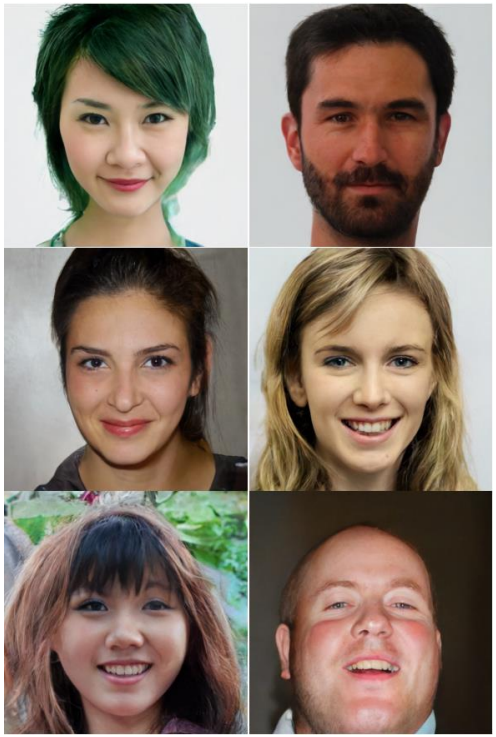
(conditional, e.g., seq2seq)

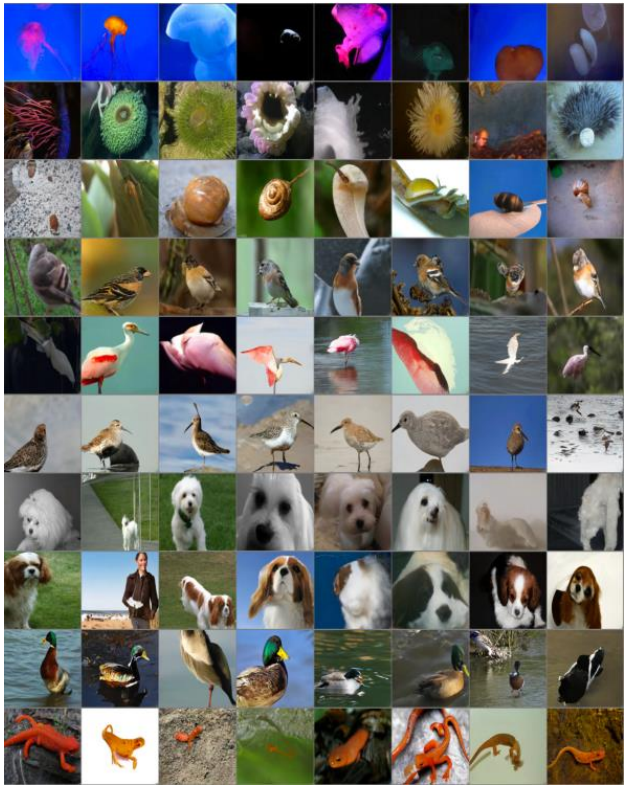
Hierarchical Autoregressive Image Models with Auxiliary Decoders. Jeffrey De Fauw, et al. Arxiv 2019

Autoregressive Models

- More state-of-the-art (SOTA) results…
  - Also see Transformer-based AR models for text/audio/music/video/graph



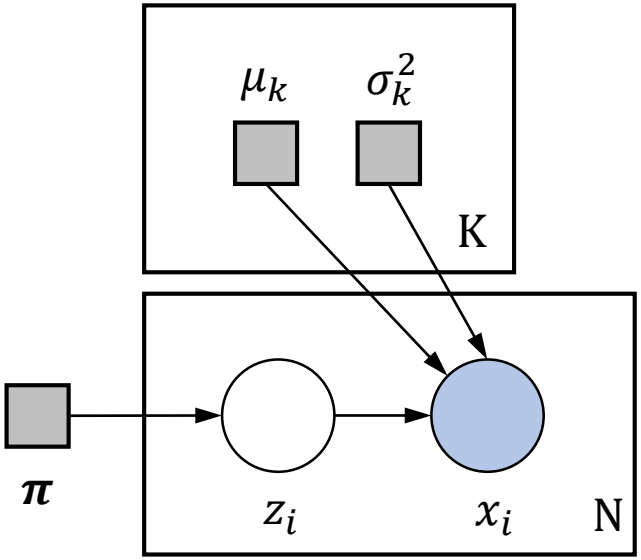




Generating High Fidelity Images with Subscale Pixel Networks and Multidimensional Upscaling. Jacob Menick, Nal Kalchbrenner. ICLR 2019
Generating Diverse High-Fidelity Images with VQ-VAE-2. Ali Razavi, et al. Arxiv 2019

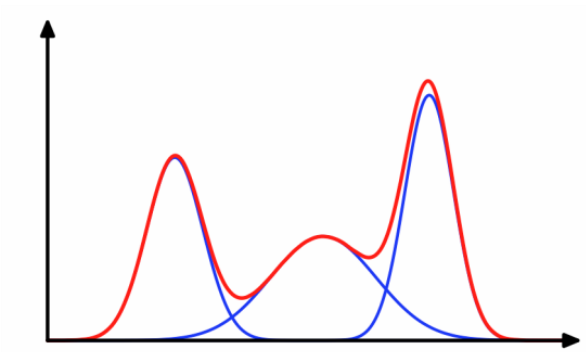
Latent Variable Models

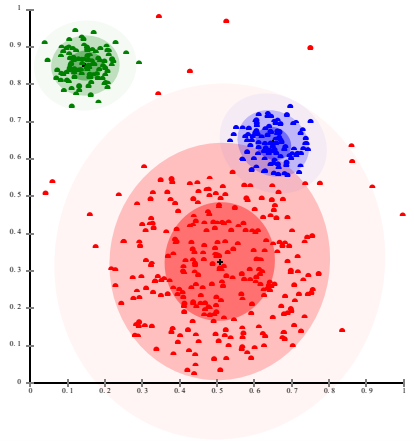
- Example: Gaussian Mixture Models



$$z_i \sim \text{categorical}(\pi_1, \ldots, \pi_K), \quad i = 1, \ldots, N,$$

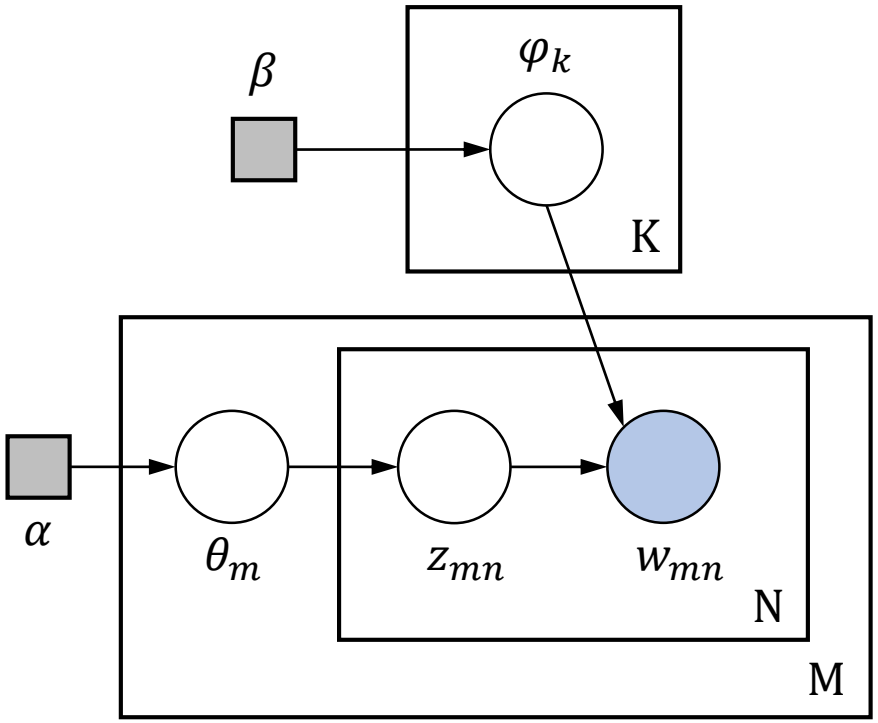
$$x_i | z_i \sim \mathcal{N}(\mu_{z_i}, \sigma_{z_i}^2) \qquad i = 1, \ldots, N.$$





Latent Variable Models

- Example: Latent Dirichlet Allocation

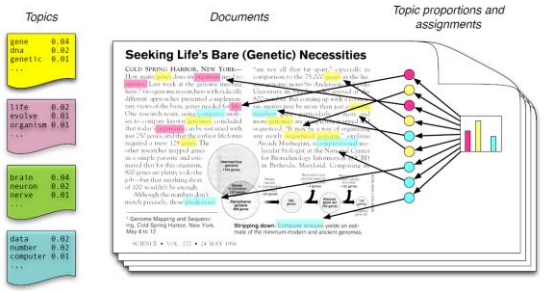


$$\varphi_k \sim Dir(\beta), \quad k = 1, \dots, K,$$

$$\theta_m \sim Dir(\alpha), \quad m = 1, \dots, \mathrm{M},$$

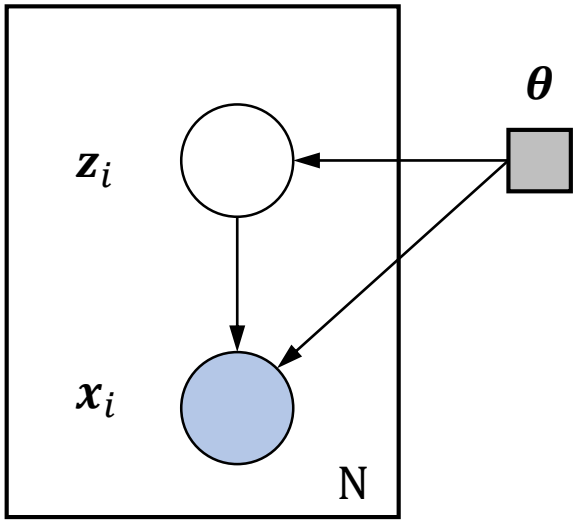
$$z_{mn} \big| \theta_m \sim \mathrm{categorical}(\theta_m),$$

$$w_{mn} \big| \boldsymbol{\varphi}, z_{mn} \sim \mathrm{categorical}\left(\varphi_{z_{mn}}\right)$$



Latent Variable Models

- Example: Deep Latent Gaussian Models



$$\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbb{I}), \qquad i = 1, \ldots, N,$$

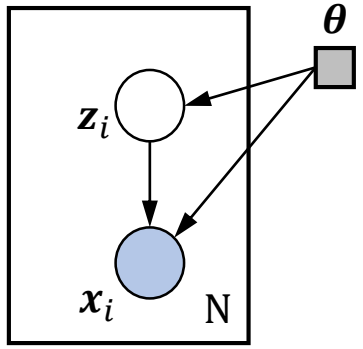
$$\boldsymbol{x}_i | \boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{z}_i), \boldsymbol{\sigma}^2(\boldsymbol{z}_i)\mathbb{I}), \quad i = 1, \ldots, N.$$

where $\boldsymbol{\mu}(\cdot)$ and $\boldsymbol{\sigma}(\cdot)$ are neural networks,
$\boldsymbol{\theta} = \{\text{parameters of } \boldsymbol{\mu} \text{ and } \boldsymbol{\sigma}\}$

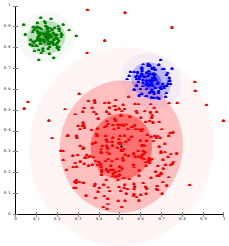
BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. Lars Maaløe, et al. Arxiv 2019

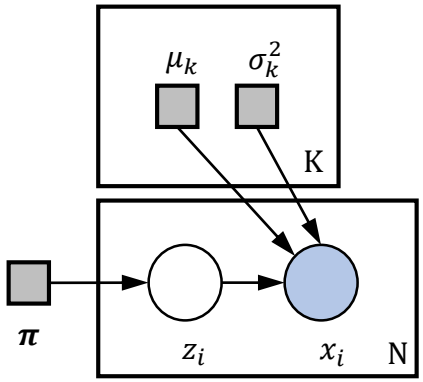
Latent Variable Models

- Learning: fit the model to data
  - Find the maximum likelihood estimation of the parameters $\boldsymbol{\theta}$
- Inference: compute unknown probability distributions
  - Posterior distribution of latent variable $\boldsymbol{z}$
  - Marginal likelihood of observations $\boldsymbol{x}$



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

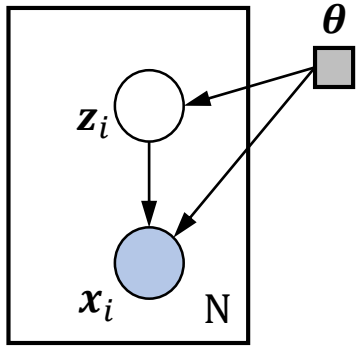




LVM: The Need for **Approximate** Inference

- Posterior distribution of latent variables $\boldsymbol{z} = \{\boldsymbol{z}_i\}, i = 1 \ldots \mathrm{M}$
- Marginal likelihood of observations $\boldsymbol{x} = \{\boldsymbol{x}_i\}, i = 1 \ldots \mathrm{N}$

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{x})} = \frac{\overbrace{p(\boldsymbol{x}|\boldsymbol{z})}^{\text{likelihood}}\overbrace{p(\boldsymbol{z})}^{\text{prior}}}{\underbrace{\int p(\boldsymbol{x},\boldsymbol{z})d\boldsymbol{z}}_{\text{marginal/evidence}}}$$

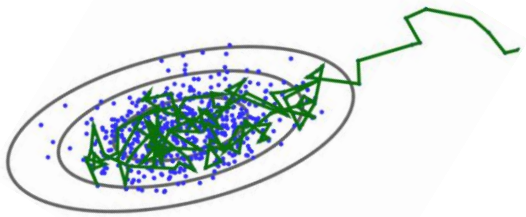


For many models, the integral (or sum) is **intractable**:

- Unavailable in closed form, or
- Requires exponential time to compute.

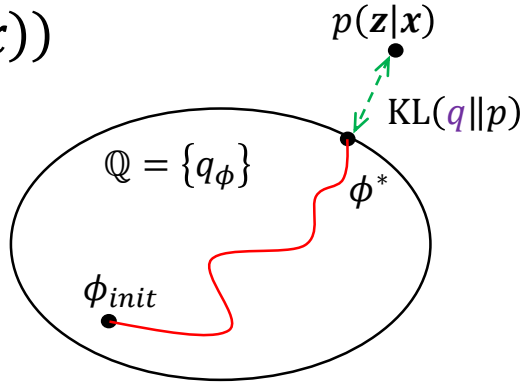
LVM: Approximate Inference of $p(\boldsymbol{z}|\boldsymbol{x})$

- Sampling: MCMC (Metropolis-Hasting or Gibbs sampling)
  - Approximate the posterior using samples
  - √ Converge to the posterior asymptotically
  - X Computationally intensive



- Variational Inference: turn inference into an optimization problem
  - Set up a ***family*** of approximate densities $\mathbb{Q}$ over the latent variables
  - Find the member q^* in the family $\mathbb{Q}$ that is closest to the exact posterior

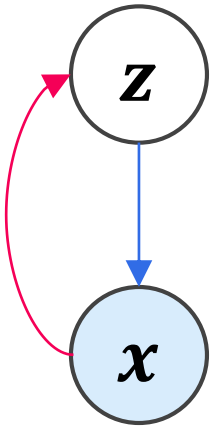
$$q^*(\boldsymbol{z}) = \arg\min_{q(\boldsymbol{z}) \in \mathbb{Q}} \mathrm{KL}(q(\boldsymbol{z}) \| p(\boldsymbol{z}|\boldsymbol{x}))$$



  - √ Faster and easier to scale to large datasets
  - X Biased

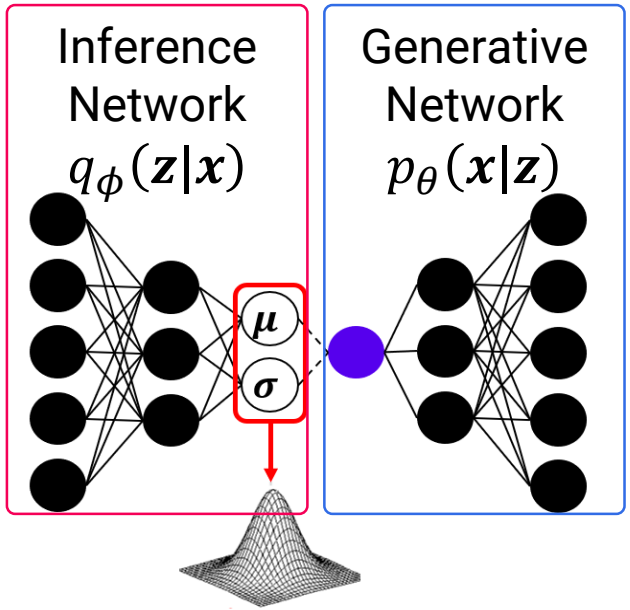
LVM: Variational Autoencoders

- VAE uses *amortized* VI by introducing a parameterized ***inference network*** (i.e., encoder)
- Parameters of the encoder and the decoder are jointly optimized
  - By SGD on the Evidence Lower Bound Objective (ELBO)
- Make the probabilistic graphical models great again
  - Enable us to build PGMs with neural nets in a flexible way



$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}, \boldsymbol{z})\, \mathrm{d}\boldsymbol{z}$$

$$\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$$



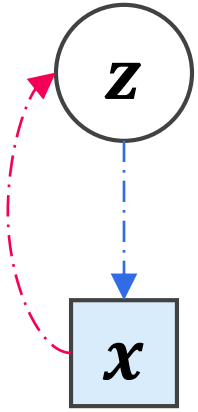
Latent Variable Models

- SOTA VAEs are lagging behind AR models on image generation
- But deep SSMs (e.g., stochastic RNNs) tend to outperform RNN-based AR models on datasets that exhibit natural sequential order
  - More on this later

BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. Lars Maaløe, et al. Arxiv 2019

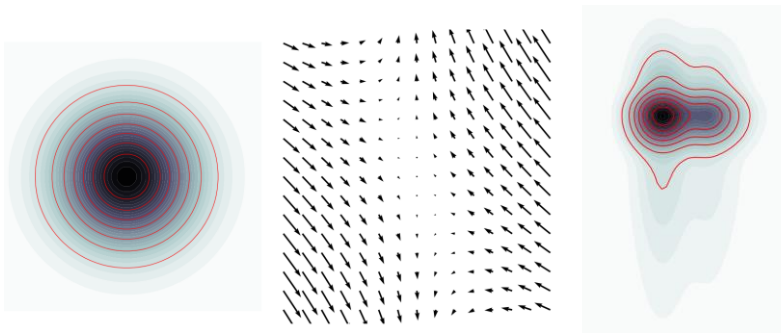
Normalizing Flows

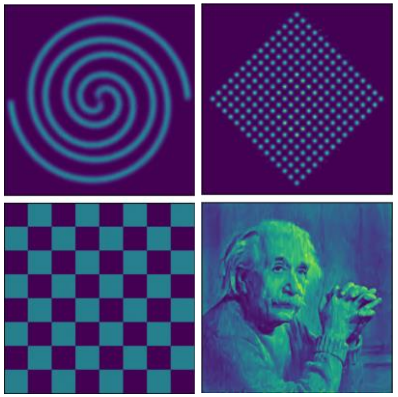
- Transform a simple density to a complex one
  - Using multilayer ***invertible*** mappings (or ODEs)
- Exact likelihood by the ***change of variable*** formula
- Usually fast sampling
- Exciting new area, many interesting ideas
  - Bonus: can help reduce memory footprint

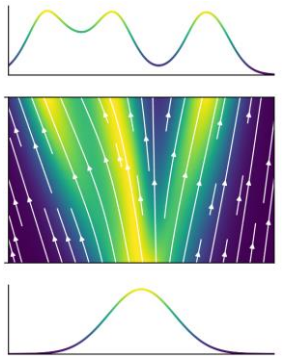


$$x = f_\theta(z)$$

$$p_\theta(x) = p_\theta(z)\left|\det \frac{\partial f_\theta^{-1}(x)}{\partial x}\right|$$







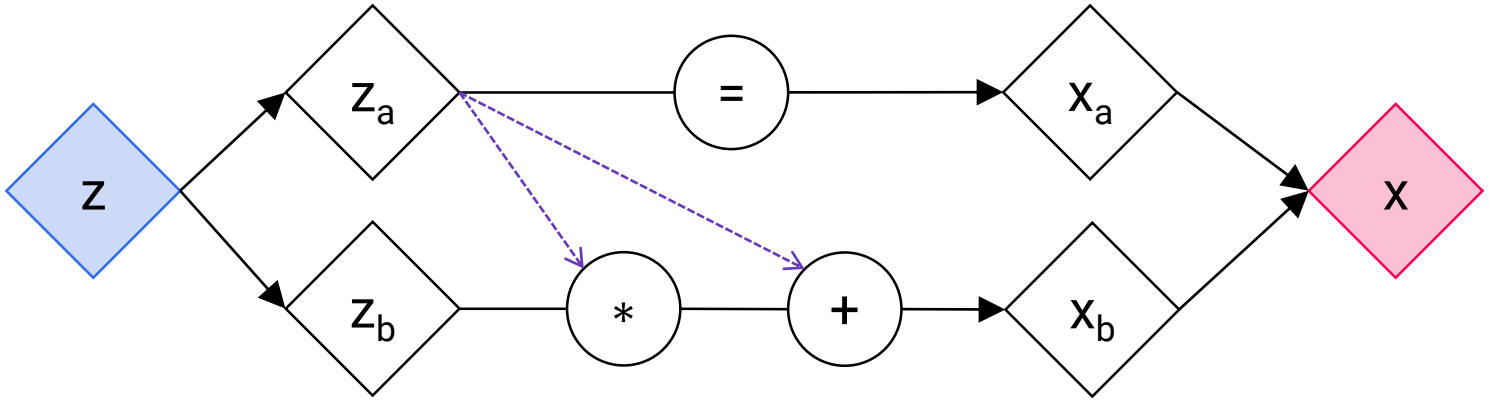
A Tutorial on Deep Probabilistic Generative Models. Ryan P. Adams. Machine Learning Summer School 2018
FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models. W Grathwohl, et al. ICLR 2019

Normalizing Flows

- Example: Design an invertible mapping $f_\theta: \mathbb{R}^D \to \mathbb{R}^D$ [***coupling layer***]

$$z_{1:d}^{(l+1)} = z_{1:d}^{(l)}$$

$$z_{d+1:D}^{(l+1)} = z_{d+1:D}^{(l)} \odot \exp\left(s_\theta\left(z_{1:d}^{(l)}\right)\right) + t_\theta\left(z_{1:d}^{(l)}\right)$$

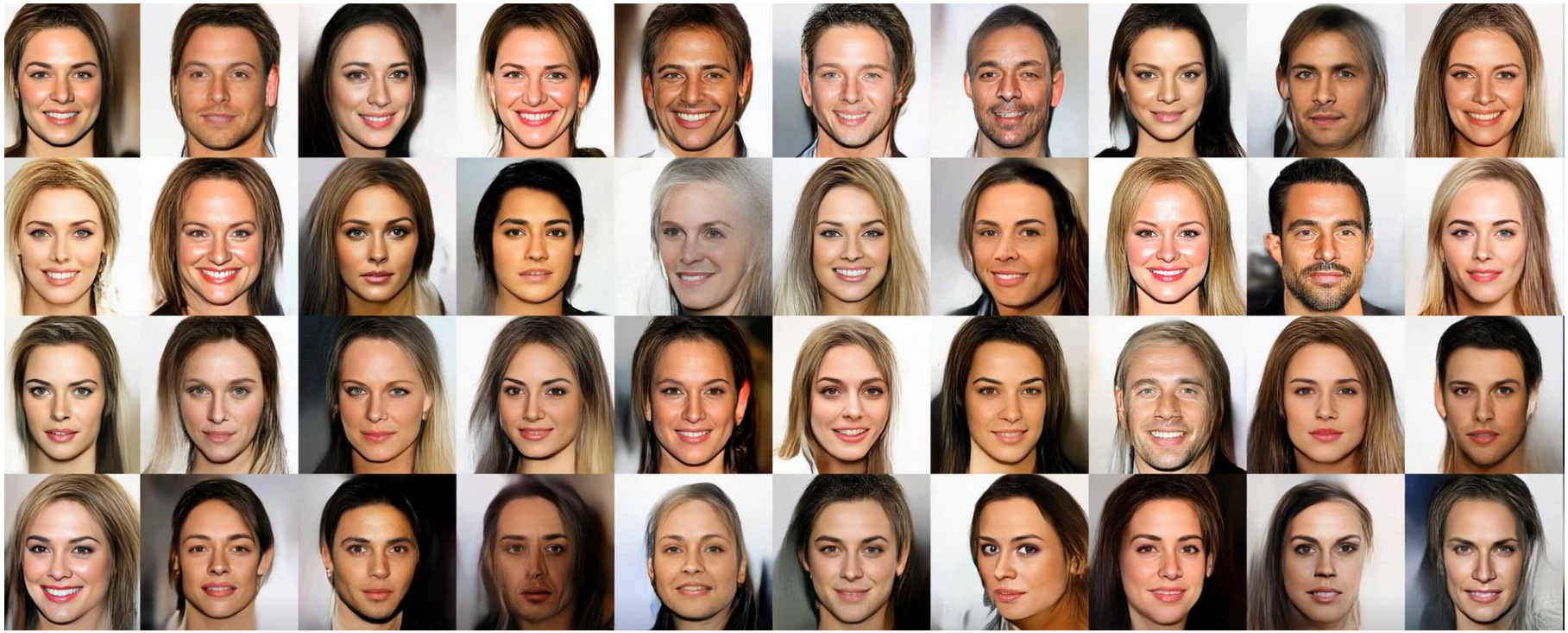


$$\log\left|\det\frac{\partial f_\theta^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right| = -\sum_{i=1}^{D-d} s_\theta(\boldsymbol{z}_a)_i$$

Density Estimation using Real NVP. L Dinh, et al. ICLR 2017

Normalizing Flows

- Recent advances make them comparable to AR models
  - Lagging behind AR on unconditional generation when trained on ImageNet



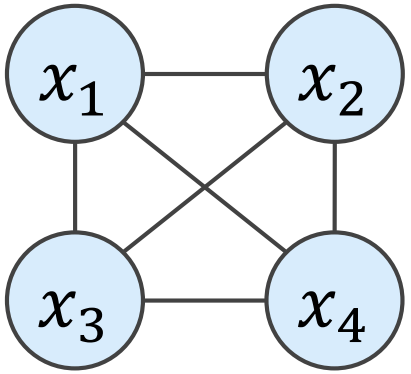
Glow: Generative Flow with Invertible 1x1 Convolutions. DP Kingma and P Dhariwal. NeurIPS 2018
Flow++: Improving Flow-Based Generative Models with Variational Dequantization and Architecture Design. Jonathan Ho, et al. ICML 2019

Energy-Based Models

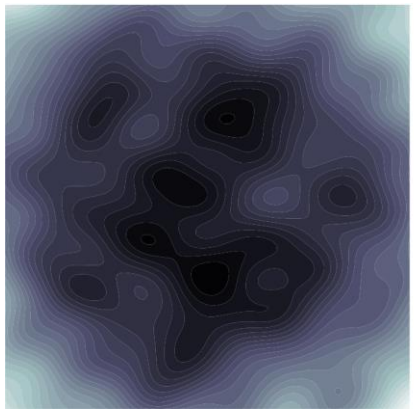
- Parametrize any scalar ***energy*** function $E_\theta(\boldsymbol{x})$
  - $-E_\theta(\boldsymbol{x})$ can be interpreted as a ***score*** function
    - i.e., it gives the "goodness of configurations"
- Intractable likelihood, hard to train
  - Because the partition function Z_θ is intractable
- Sampling is usually hard and slow (need MCMC)
- Highly expressive and flexible
  - Also some nice properties, e.g., compositionality

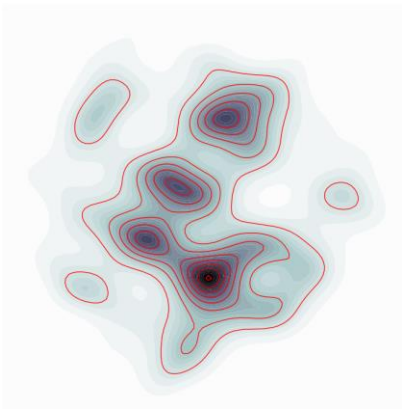
$$p_\theta(\boldsymbol{x}) = \frac{e^{-E_\theta(\boldsymbol{x})}}{Z_\theta}$$

$$Z_\theta = \int e^{-E_\theta(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x}$$

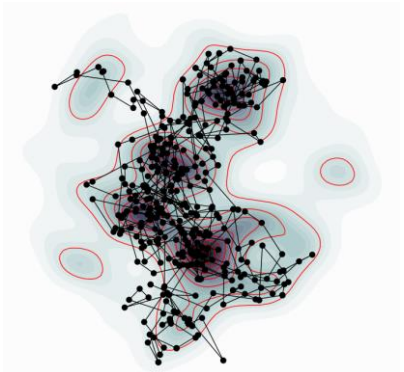


$-E_\theta(\boldsymbol{x})$





$p_\theta(\boldsymbol{x})$



A Tutorial on Deep Probabilistic Generative Models. Ryan P. Adams. Machine Learning Summer School 2018

Energy-Based Models

- There is renewed interest in them; but still have a long way to go

https://openai.com/blog/energy-based-models/
Maximum Entropy Generators for Energy-Based Models. Rithesh Kumar, et al. Arxiv 2019
On the Anatomy of MCMC-based Maximum Likelihood Learning of Energy-Based Models. Erik Nijkamp, et al. CVPR 2019

Quick Summary

- Where are (your favorite) **GAN**s? Wait a minute …

- In generative modeling, our goal is …

  to make the model $p_\theta(\boldsymbol{x})$ **mimic** the true data distribution $p^*(\boldsymbol{x})$

  - By minimizing some **divergence** measure: $\theta^* = \underset{\theta}{\operatorname{argmin}}\, D(p^*, p_\theta)$

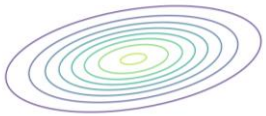
- So far, AR/Flow/LVM/EBM are usually trained via MLE

  - Why MLE? It was derived by choosing D to be **Kullback–Leibler divergence**

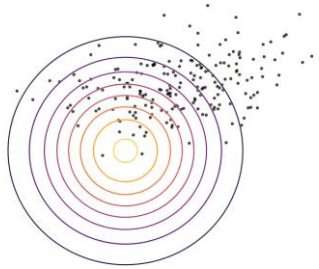
$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathbf{KL}\left(p^* \| p_\theta\right) = \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{\boldsymbol{x} \sim p^*}\left[\log \frac{p^*(\boldsymbol{x})}{p_\theta(\boldsymbol{x})}\right]$$

$$= \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{\boldsymbol{x} \sim p^*}[\log p^*(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x} \sim p^*}[\log p_\theta(\boldsymbol{x})]$$

$$= \underset{\theta}{\operatorname{argmax}}\, \mathbb{E}_{\boldsymbol{x} \sim p^*}[\log p_\theta(\boldsymbol{x})] \approx \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \log p_\theta\left(\boldsymbol{x}^{(i)}\right)$$



$p^*(\boldsymbol{x})$



$p_\theta(\boldsymbol{x})$

https://colinraffel.com/blog/gans-and-divergence-minimization.html

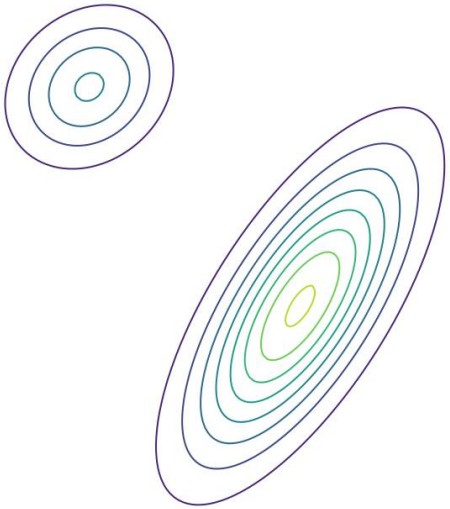
Quick Summary

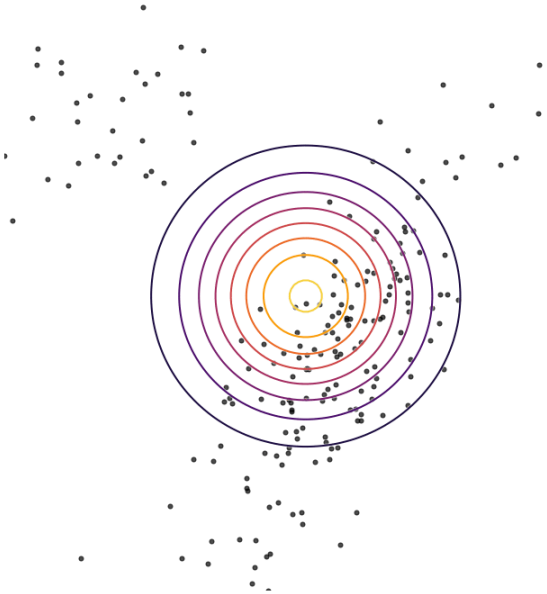
- Some generated images are not so realistic… Why?

$$\theta^* = \arg\min_\theta \mathbf{KL}\left(p^* \| p_\theta\right) \approx \arg\max_\theta \sum_{i=1}^{N} \log p_\theta\left(\boldsymbol{x}^{(i)}\right)$$

- For real data, most models are mis-specified: $p^* \notin \{p_\theta | \theta \in \Theta\}$

$p^*(\boldsymbol{x})$



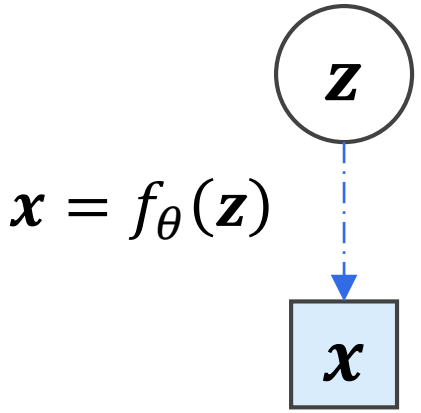


$p_\theta(\boldsymbol{x})$

https://colinraffel.com/blog/gans-and-divergence-minimization.html

Implicit Models

- A.k.a. ***likelihood-free*** models or ***simulator*** models
  - Also called ***neural samplers*** when $f_\theta(\cdot)$ is NN



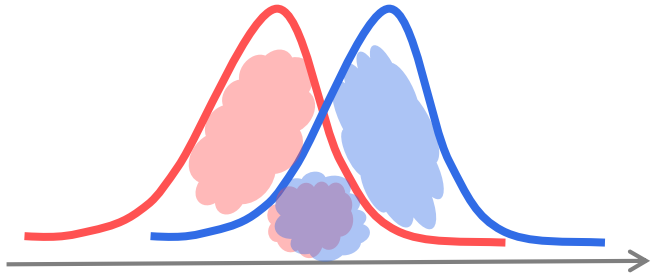
- No explicit likelihood, so MLE doesn't work
  - Because $f_\theta(\cdot)$ may be non-surjective and non-injective

$$p_\theta(x) = ?$$

- Can only sample from the model
  - Sampling is fast and differentiable w.r.t. θ

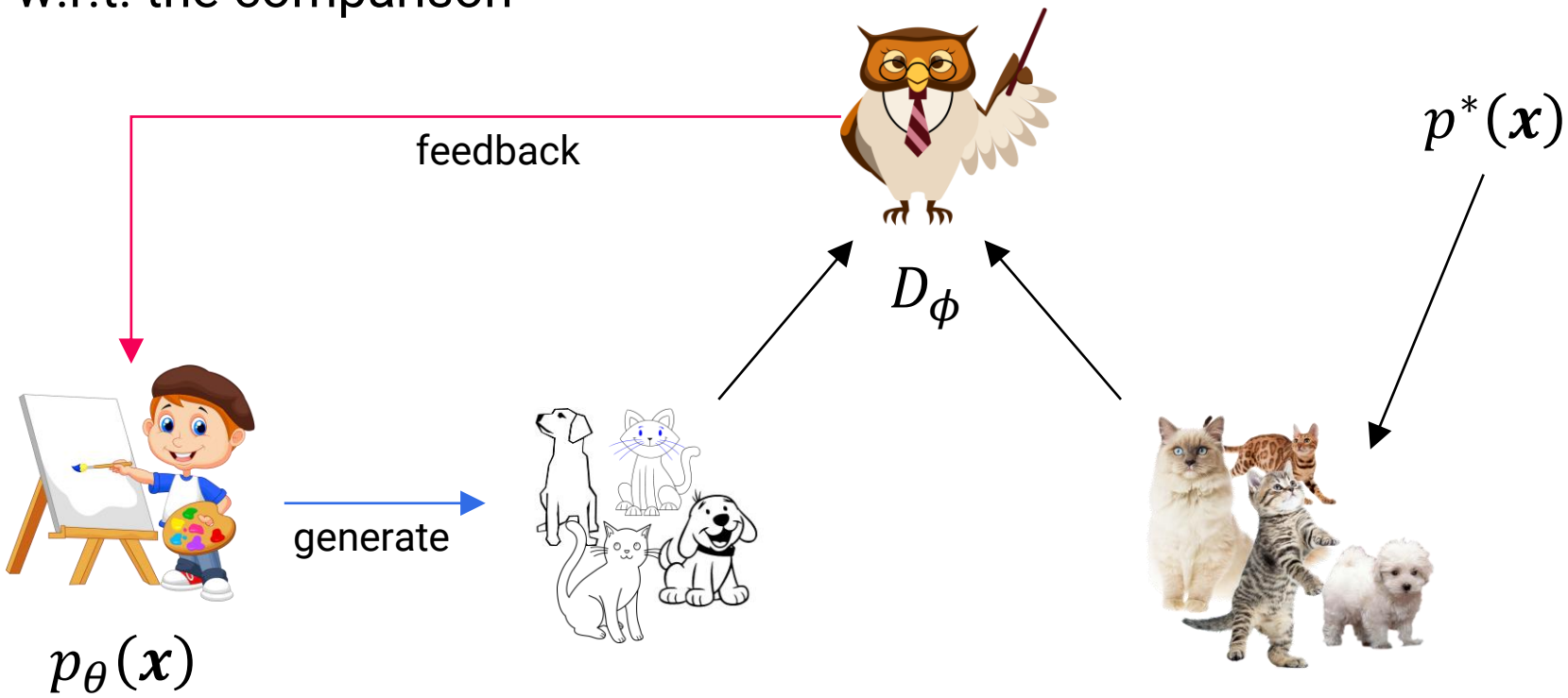
- I just want fantasy samples. How to learn the model?

- Idea: comparing p_θ and p^* using **samples**



Implicit Models: Learning by Comparison

- Use an auxiliary model f_ϕ (a.k.a. discriminator/critic/teacher) to **test** how a set of simulated samples differs from observed data
- Adjust the generative model p_θ to better match the data distribution w.r.t. the comparison



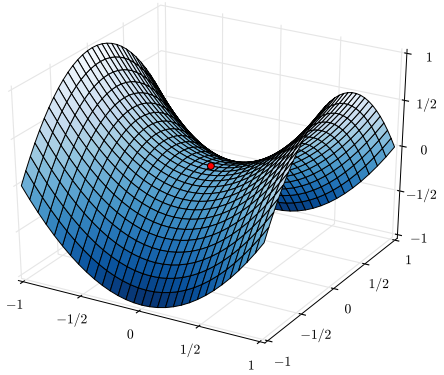
The Maximum Mean Discrepancy for Training Generative Adversarial Networks. Arthur Gretton. 2018

Implicit Models: GAN

- The teacher should be strict
- The student should challenge the teacher
- Let them play the following two-player minimax game:

$$\min_{\theta} \max_{\phi} \mathbb{E}_{\boldsymbol{x} \sim p^*}\left[\log D_{\phi}(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{x} \sim p_{\theta}}\left[\log\left(1 - D_{\phi}(\boldsymbol{x})\right)\right]$$

- Implement this game using ***alternating optimization***
  - Alternate between (1 step of) updating θ and (k step of) updating ϕ
  - One player minimizes the objective function maximized by the other
  - Seek a ***saddle point*** / an ***equilibrium***

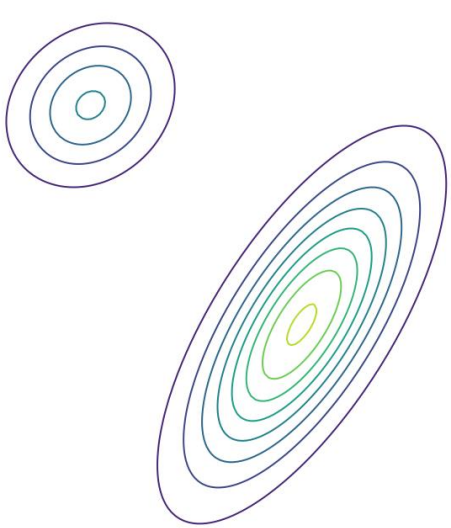


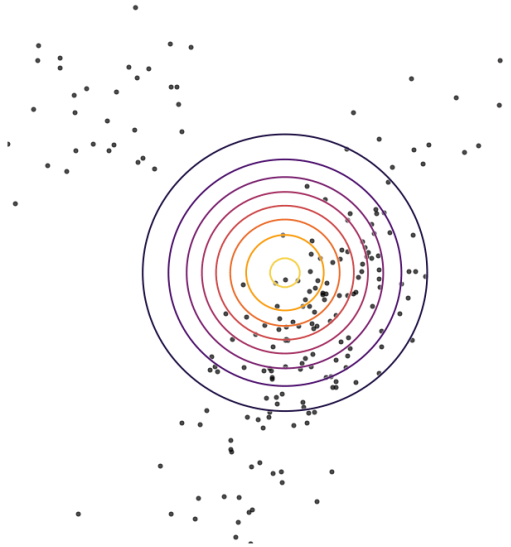
GAN: Mode Collapse

- GAN is shown to be minimizing the **J**ensen–**S**hannon divergence or **reverse K**ullback–**L**eibler divergence between p^* and p_θ

$$\min_\theta \mathbf{KL}\left(p_\theta \| p^*\right) = \min_\theta \mathbb{E}_{\boldsymbol{x} \sim p_\theta}\left[\log \frac{p_\theta(\boldsymbol{x})}{p^*(\boldsymbol{x})}\right]$$

$p^*(\boldsymbol{x})$





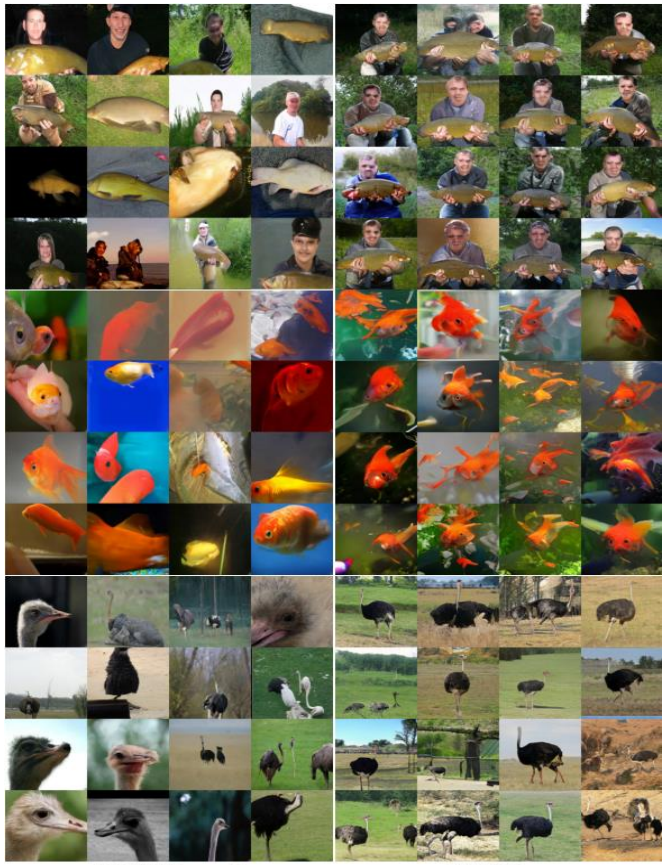
$p_\theta(\boldsymbol{x})$

https://colinraffel.com/blog/gans-and-divergence-minimization.html

Implicit Models

- SOTA GANs can generate fantasy samples
  - Trade-off between ***quality*** & ***diversity***

Training Set	Resolution	Top-5 Accuracy	Top-1 Accuracy	Inception Score	FID-50K
Real	128×128	88.79%	68.82%	165.38 ± 2.84	1.61
BigGAN-deep	128×128	64.44%	40.64%	71.31 ± 1.57	4.22
Hierarchical Autoregressive	128×128	**77.33%**	**54.05%**	17.02 ± 0.79	46.05
Real	256×256	91.47%	73.09%	331.83 ± 5.00	2.47
BigGAN-deep	256×256	65.92%	42.65%	109.39 ± 1.56	11.78
High-Res VQ-VAE	256×256	**77.59%**	**54.83%**	43.44 ± 0.87	38.05



VQ-VAE (Proposed) BigGAN deep

Large Scale GAN Training for High Fidelity Natural Image Synthesis. Andrew Brock, et al. ICLR 2019
Classification Accuracy Score for Conditional Generative Models. Suman Ravuri, Oriol Vinyals. Arxiv 2019
Generating Diverse High-Fidelity Images with VQ-VAE-2. Ali Razavi, et al. Arxiv 2019

Quick Summary

	Sampling	Likelihood	Training	Latent	SOTA Perf.
AR	Slow	Exact	MLE	No*	1*
Flow	Fast*	Exact	MLE	Yes	2
LVM	Fast	Approx.	Approx. MLE	Yes	3
EBM	Very slow	Intractable*	Approx. MLE	No	4
Implicit	Fast	Intractable	Adversarial	No*	1*

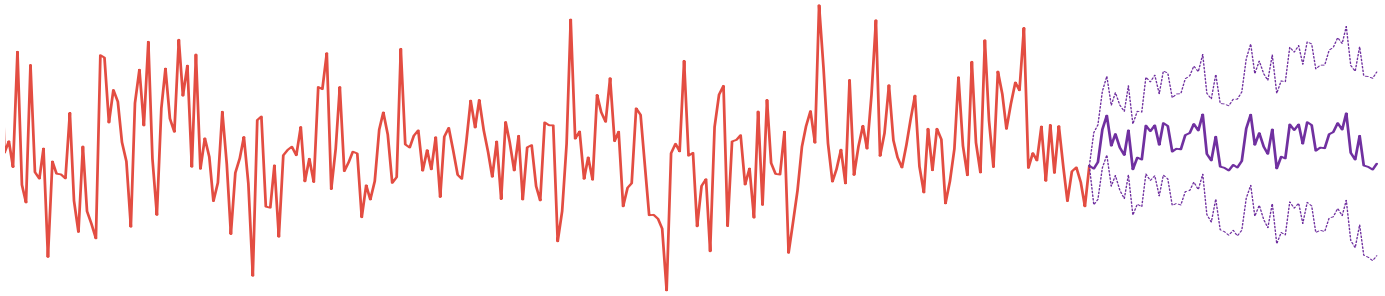
Outline

- **Deep Generative Models**
  - What; Why; Landscape of DGMs; **Sequential DGMs**

DGMs for Sequential Data

- For sequential data, we may:
  - Simply use AR models, as they perfectly match the order of data
  - Simply plug RNNs/CNNs into VAEs/GANs and play

- Besides, we can further exploit the sequential structure to design more flexible/expressive sequential (D)GMs



A Spectrum of Time Series Models

- Interestingly, most time series models are generative models

Limited capacity
Data efficient
Fast learning/inference
Easy to understand

Highly flexible
(Data intensive)
Expensive learning
(Harder to interpret)

ARIMA

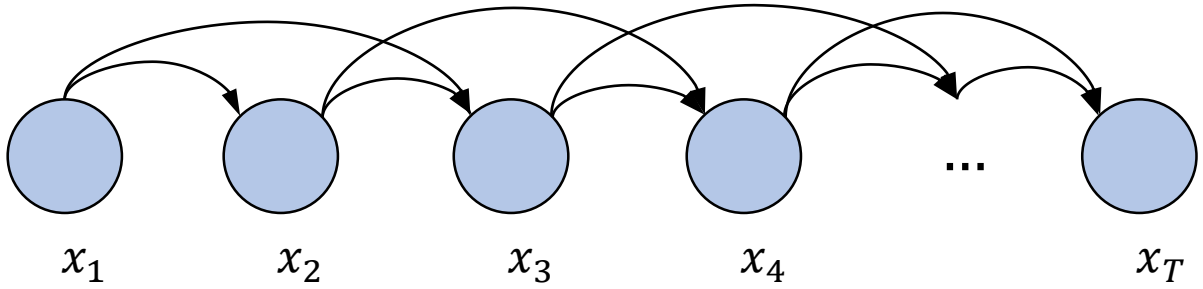
- Example 1: **A**uto-**R**egressive **I**ntegrated **M**oving **A**verage

$$\text{ARIMA}(p,0,0): \quad x_t = c + \epsilon_t + \sum_{i=1}^{p} \theta_i x_{t-i}\,,\ \epsilon_t \sim \mathcal{N}(0,\sigma^2)$$

$$\Leftrightarrow$$

$$x_t \sim \mathcal{N}\left(c + \sum_{i=1}^{p} \theta_i x_{t-i}\,, \sigma^2\right)$$

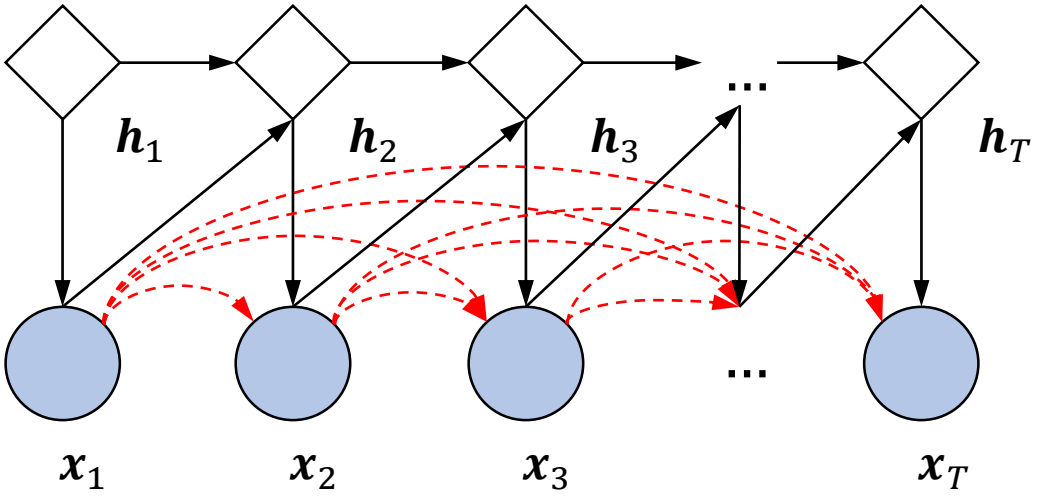
$$p_\theta(x_{1:T}) = p_\theta(x_{1:p}) \prod_{t=p+1}^{T} p_\theta\left(x_t \middle| x_{t-p:t-1}\right)$$



RNN

- Example 2: **R**ecurrent **N**eural **N**etworks (LSTM, GRU, …)
- RNNs are also ***autoregressive*** models
  - Compared to ARIMA, RNNs enable longer and more complex historical dependencies

$$p_\theta(\boldsymbol{x}_{1:T}) = \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_t|\boldsymbol{x}_{<t}) = \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_t|\boldsymbol{h}_t)$$



$$\boldsymbol{x}_1 \sim g_\theta(\cdot\,|\boldsymbol{h}_1)$$

$$\boldsymbol{h}_t = \mathrm{LSTM}_\theta(\boldsymbol{h}_{t-1}, \boldsymbol{x}_{t-1})$$

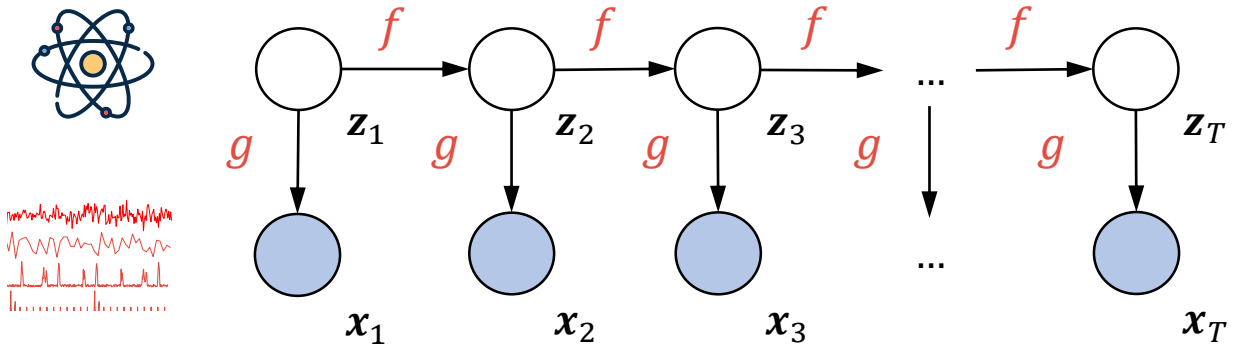
$$\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1} \sim g_\theta(\cdot\,|\boldsymbol{h}_t)$$

$$\text{where } t = 2 \ldots T$$

the actual conditional dependences are shown in red dashed lines

State-Space Models

- SSMs are a general class of probabilistic sequential models
  - Unlike AR models, SSMs introduce latent variables to enhance their capability
- Basic elements in SSMs:
  - parameters $\boldsymbol{\theta}$, prior density π_θ, transition density f_θ, measurement density g_θ
  - latent states $\boldsymbol{z}_t$, observations $\boldsymbol{x}_t$



$$\boldsymbol{z}_1 \sim \pi_\theta(\cdot)$$

$$\boldsymbol{z}_t|\boldsymbol{z}_{t-1} \sim f_\theta(\cdot\,|\boldsymbol{z}_{t-1})$$

$$\boldsymbol{x}_t|\boldsymbol{z}_t \sim g_\theta(\cdot\,|\boldsymbol{z}_t)$$

where $t = 2 \ldots T$

$$p_\theta(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T}) = \pi_\theta(\boldsymbol{z}_1) g_\theta(\boldsymbol{x}_1|\boldsymbol{z}_1) \prod_{t=2}^{T} f_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) g_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t)$$

Example: Gaussian Linear Dynamical Systems

latent dynamics

$$\vec{z}_{t+1} \sim \mathcal{N}\left(\boldsymbol{A}\vec{z}_t + \vec{b}, \boldsymbol{Q}\right)$$

observation model

$$\vec{x}_{t+1} \sim \mathcal{N}\left(\boldsymbol{C}\vec{z}_{t+1} + \vec{d}, \boldsymbol{R}\right)$$

Example: Deep SSMs

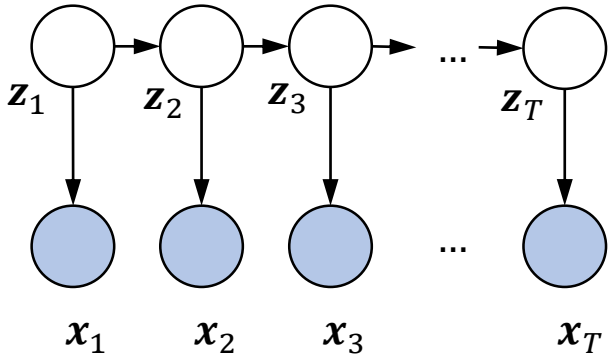
- Deep Markov Model

$$\boldsymbol{z}_1 \sim \pi_\theta(\cdot)$$

$$\boldsymbol{z}_t|\boldsymbol{z}_{t-1} \sim \mathcal{N}\left(\mu_\theta(\boldsymbol{z}_{t-1}), \sigma_\theta^2(\boldsymbol{z}_{t-1})\right)$$

$$\boldsymbol{x}_t|\boldsymbol{z}_t \sim g_\theta(\cdot|\boldsymbol{z}_t)$$

where $\mu_\theta(\cdot)$ and $\sigma_\theta^2(\cdot)$ are (gated) neural networks



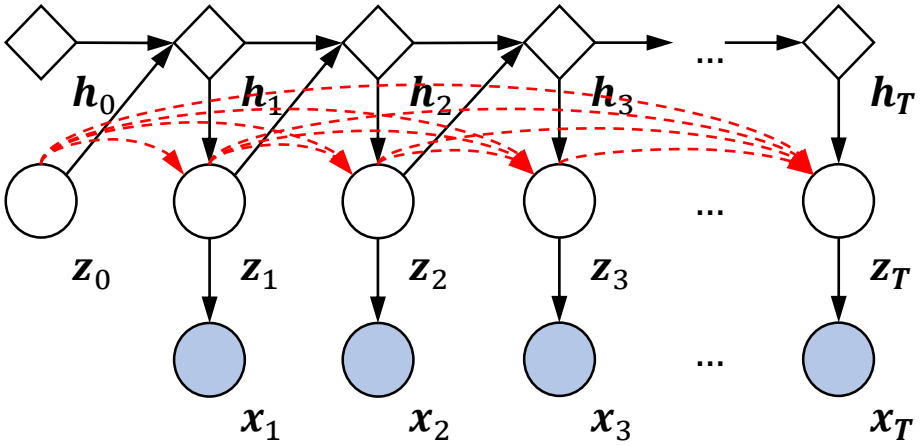
- LSTM State Space Model

$$\boldsymbol{z}_0 \sim \pi_\theta(\cdot)$$

$$\boldsymbol{h}_t = \mathrm{LSTM}_\theta(\boldsymbol{h}_{t-1}, \boldsymbol{z}_{t-1})$$

$$\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1} \sim f_\theta(\cdot|\boldsymbol{h}_t)$$

$$\boldsymbol{x}_t|\boldsymbol{z}_t \sim g_\theta(\cdot|\boldsymbol{z}_t)$$

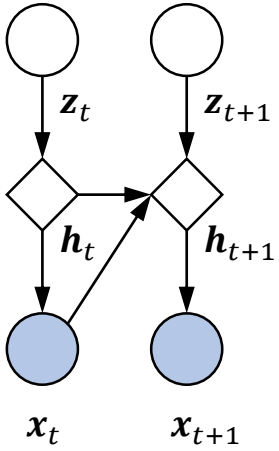


Structured Inference Networks for Nonlinear State Space Models. Krishnan, et al. AAAI 2017
State Space LSTM Models with Particle MCMC Inference. Zheng, et al. Arxiv 2017

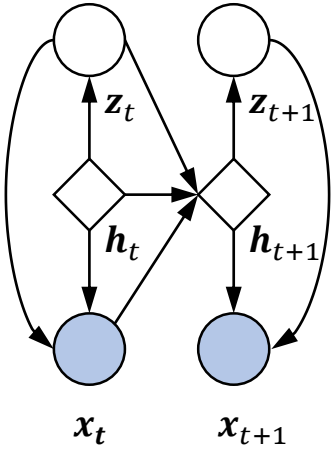
Example: Stochastic RNNs

- Injecting random variables into vanilla RNNs

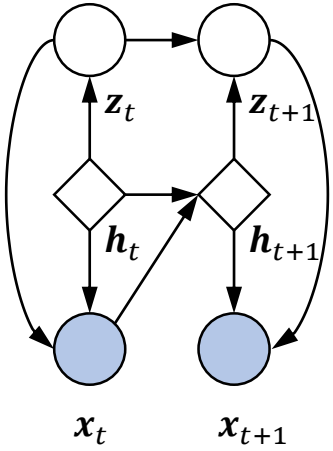
STORN [Arxiv 2014]
Noisin [ICML 2018]



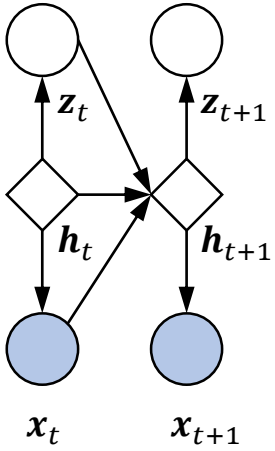
VRNN [NIPS 2015]



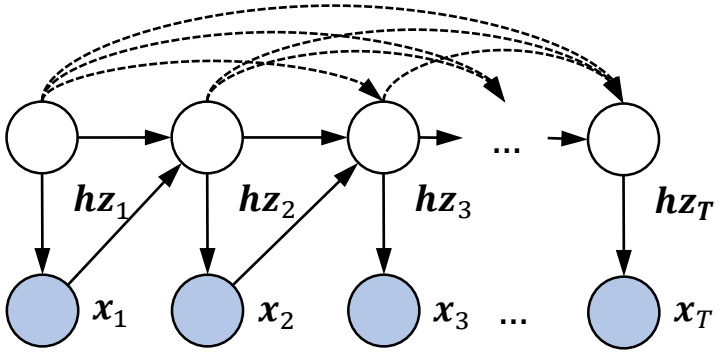
SRNN [NIPS 2016]



Z-Forcing [NIPS 2017]



After merging
the deterministic & stochastic state
at each time step,
almost all of them resemble this:

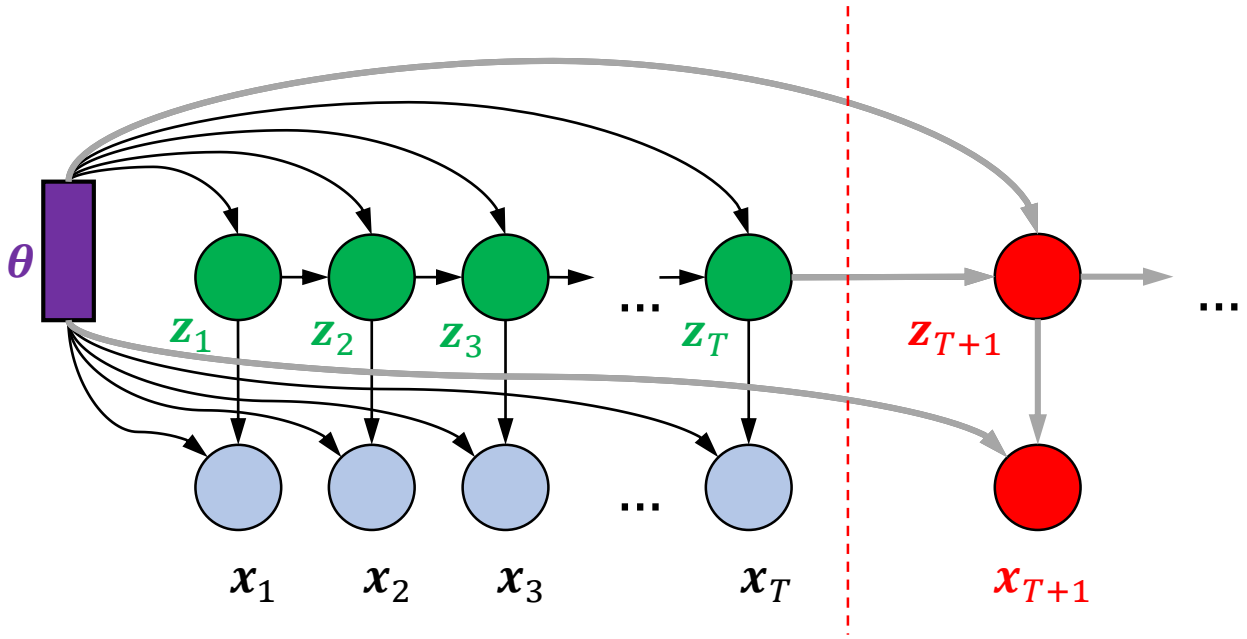


Playing with Probabilistic Sequential Models

- Given these models, our basic tasks are:
  - **Model learning**: MLE for the model parameters $\boldsymbol{\theta}$: $\boldsymbol{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\text{argmax}}\ p_{\boldsymbol{\theta}}(\boldsymbol{x})$
  - **Bayesian inference**: posterior distribution of latent variables $\boldsymbol{z}$: $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$
  - **Prediction**: conditional distribution of future states: $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{T+1:T+L}|\boldsymbol{x}_{1:T})$
    - And future observations: $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{T+1:T+L}|\boldsymbol{x}_{1:T})$

Many learning algorithms available:

- **E**xpectation **M**aximization
- **V**ariational **I**nference
- **S**equential **M**onte **C**arlo
- Particle MCMC
- Likelihood-free methods



On Particle Methods for Parameter Estimation in State-Space Models. N Kantas, et al. Statistical Science 2015