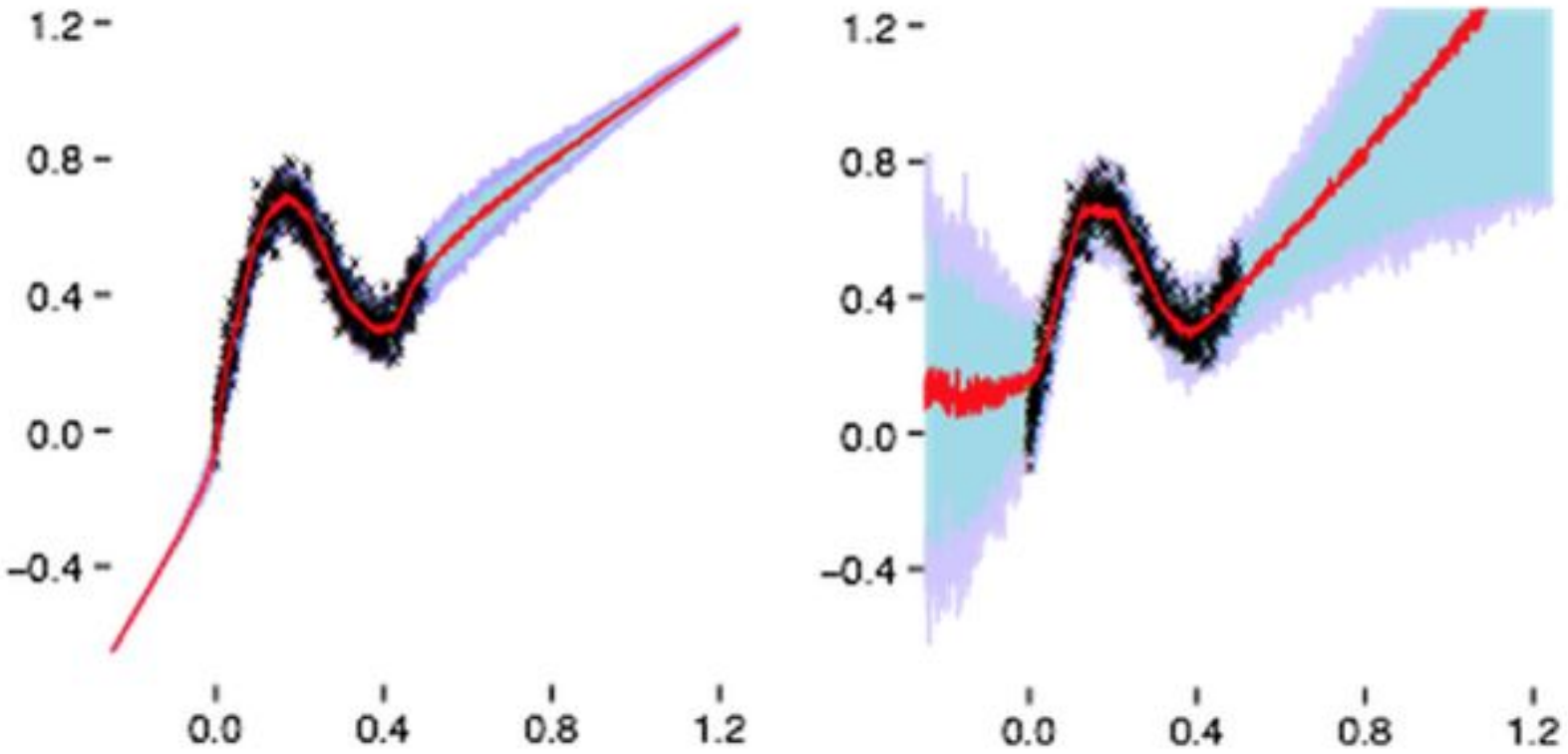
Approximate Inference and Generative Models

Danilo J. Rezende
Google DeepMind

Hammers & Nails - Machine Learning & HEP
July 19-28, 2017 | Weizmann Institute of Science, Israel

Making DNNs a little bit more Bayesian



Weight Uncertainty in Neural Networks https://arxiv.org/abs/1505.05424

Why Generative Models?

- Compact mechanism to express causal relations and dependencies
- Make predictions (with uncertainty)
- Plan sequences of decisions (planning as inference)
- Experiment design (planning + expected information gain)
- Data compression
- Hypothesis testing
- Handle missing data

Definitions

Observed variables

Unobserved or latent variables

$$\{x_i\} \quad x_i \in \Omega \,(\text{e.g. } \mathbb{R}^{d_x}, [0,1]^{d_x}, \{0,1\}^{d_x})$$

$$\{z_i\} \quad z_i \in \Lambda \,(\text{e.g. } \mathbb{R}^{d_z})$$

$$\theta \quad \in \mathbb{R}^{d_\theta}$$

$$p \quad : \Omega \otimes \Lambda \otimes \mathbb{R}^{d_\theta} \to \mathbb{R}^+ \quad s.t. \int_{x \in \Omega, z \in \Lambda, \theta \in \mathbb{R}^m} dx\, dz\, d\theta p(x,z,\theta) = 1$$

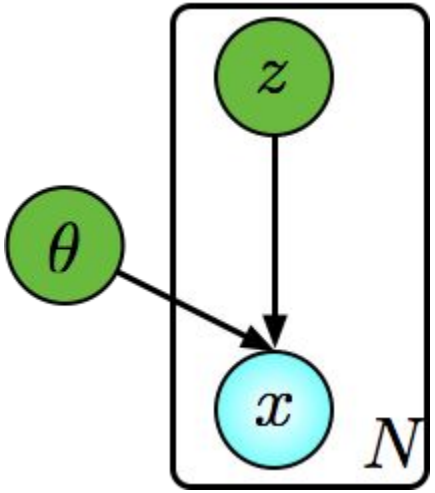
$$p \quad \in C^2$$

$$p(x,z,\theta) \quad = \frac{1}{\mathcal{Z}} e^{-U(x,z,\theta)}$$

$$\mathbb{E}_{x \sim p}[f(x)] = \int dx p(x) f(x)$$

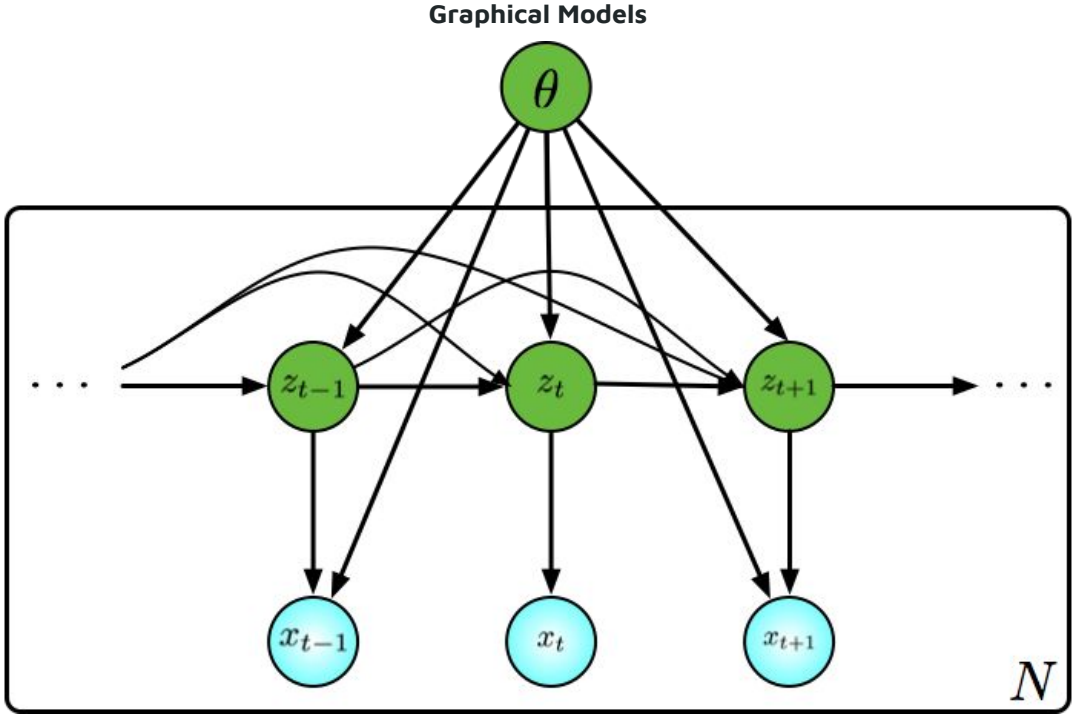
Building Generative Models

Graphical Models



$$p(x, z, \theta) = \rho(\theta) \prod_{i=1}^{N} p(x_i|z_i, \theta)\pi(z_i)$$

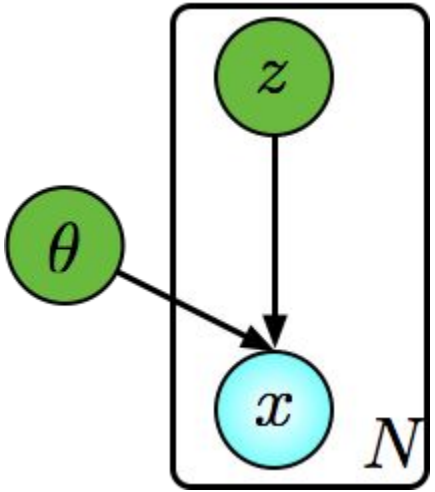
Building Generative Models



$$p(x, z, \theta) = \rho(\theta) \prod_{i=1}^{N} \prod_{t=0}^{T} p(x_t^i | z_{1,\ldots,t}^i, \theta) \pi(z_t^i | z_{1,\ldots,(t-1)}^i, \theta)$$

Building Generative Models

Graphical Models: a concrete example

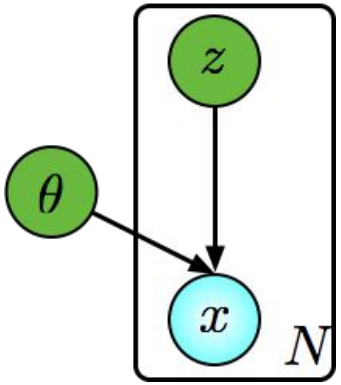


$$p(x, z, \theta) = \rho(\theta) \prod_{i=1}^{N} p(x_i|z_i, \theta)\pi(z_i)$$

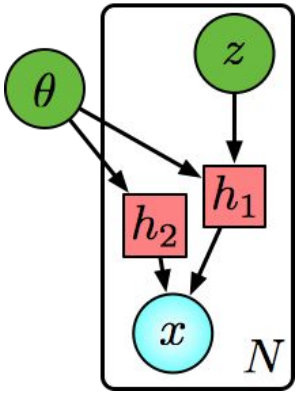
$$\begin{aligned}
\pi(z) &= \mathcal{N}(0, \mathbb{I}_{d_z}) \\
\rho(\theta) &= \mathcal{N}(0, \kappa^2 \mathbb{I}_{d_\theta}) \\
p(x|z, \theta) &= \mathcal{N}(\theta_0 + \theta_1 z, \exp(\theta_2)) \\
\theta &= \{\theta_0 \in \mathbb{R}^{d_x}, \theta_1 \in \mathbb{R}^{d_x \times d_z}, \theta_2 \in \mathbb{R}^{d_x}\}
\end{aligned}$$

Building Generative Models

Graphical Models + Computational Graphs (aka NNets)



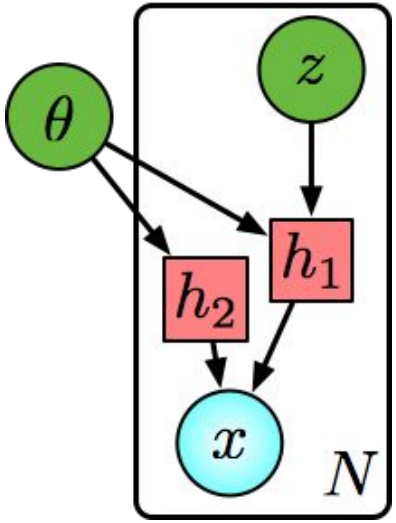
$$
\begin{aligned}
\pi(z) &= \mathcal{N}(0, \mathbb{I}_{d_z}) \\
\rho(\theta) &= \mathcal{N}(0, \kappa^2 \mathbb{I}_{d_\theta}) \\
p(x|z,\theta) &= \mathcal{N}(\theta_0 + \theta_1 z, \exp(\theta_2))
\end{aligned}
$$



$$
\begin{aligned}
\pi(z) &= \mathcal{N}(0, \mathbb{I}_{d_z}) \\
\rho(\theta) &= \mathcal{N}(0, \kappa^2 \mathbb{I}_{d_\theta}) \\
h_1 &= \theta_0 + \theta_1 z \\
h_2 &= \exp(\theta_2) \\
p(x|z,\theta) &= \mathcal{N}(h_1, h_2)
\end{aligned}
$$

Building Generative Models

Graphical Models + Computational Graphs (alternative notation)



$$
\begin{aligned}
\pi(z) &= \mathcal{N}(0, \mathbb{I}_{d_z}) \\
\rho(\theta) &= \mathcal{N}(0, \kappa^2 \mathbb{I}_{d_\theta}) \\
h_1 &= \theta_0 + \theta_1 z \\
h_2 &= \exp(\theta_2) \\
p(x|z,\theta) &= \mathcal{N}(h_1, h_2)
\end{aligned}
$$

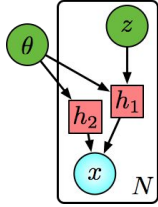
$\equiv$

$$
\begin{aligned}
\theta &\sim \mathcal{N}(0, \kappa^2 \mathbb{I}_{d_\theta}) \\
z_i &\sim \mathcal{N}(0, \mathbb{I}_{d_z}) \\
h_{1,i} &= \theta_0 + \theta_1 z_i \\
h_2 &= \exp(\theta_2) \\
x_i &\sim \mathcal{N}(h_{1,i}, h_2)
\end{aligned}
$$

What do we want to know about $p(x, z, \theta)$?

Goal	Quantity of Interest
Prediction	$p(x_{(t+1),\ldots,\infty} \vert x_{-\infty,\ldots,t})$
Planning	$J = \mathbb{E}_p \left[\int_0^\infty dt C(x_t) \middle\vert x_0, u \right]$
Parameter estimation	$p(\theta \vert x_{0,\ldots,N})$
Experiment Design	$\mathrm{EIG} = \mathrm{D}[p(f(x_{t,\ldots,\infty}) \vert u); p(f(x_{-\infty,\ldots,t}))]$
Hypothesis testing	$\frac{p(f(x_{-\infty,\ldots,t}) \vert H_0)}{p(f(x_{-\infty,\ldots,t}) \vert H_1)}$

Density estimation through conditioning



We observe N samples $\{x_{i=1,\dots,N}\}$ and we want to know $x_{N+1} \sim ?$

We start by what we know

$$p(x_{1,\dots,N+1}, z_{1,\dots,} \prod_{i=1}^{N} p(x_i|z_i,\theta)\pi(z_i)$$

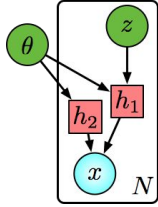
Conditioning is "just" solving integrals!

$$p(x_{N+1}|x_{1,\dots,N}) = \frac{1}{\mathcal{Z}} \int d\theta dz \rho(\theta) p(x_{N+1}|z_{N+1},\theta)\pi(z_{N+1}) \prod_{i=1}^{N} p(x_i|z_i,\theta)\pi(z_i)$$

$$\mathcal{Z} = p(x_{i=1,\dots,N})$$

Density estimation through conditioning

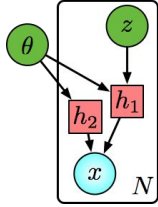
Let's approximate some integrals...



$$p(x_{N+1}|x_{1,\ldots,N}) = \frac{1}{\mathcal{Z}} \int d\theta d\, z \rho(\theta) p(x_{N+1}, z_{N+1}|\theta) \prod_{i=1}^{N} p(x_i|z_i,\theta)\pi(z_i)$$

$$= \frac{1}{\mathcal{Z}} \int d\theta d\, z_{N+1}\, p(x_{N+1}, z_{N+1}|\theta)\rho(\theta) \int d\, z \prod_{i=1}^{N} p(x_i, z_i|\theta)$$

Density estimation through conditioning



Let's approximate some integrals...

$$e^{-M(x,\theta)} = \int dz \prod_{i=1}^{N} p(x_i, z_i|\theta)$$

$$= \int dz e^{-u(x,z,\theta)}$$

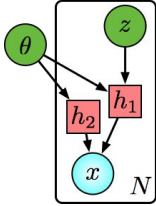
MLE estimator is given by

$$\theta^\star = \underset{\theta}{\operatorname{argmin}}\, M(x,\theta)$$

$$\mathcal{F}(x,z,\theta) = \ln q(z) + u(x,z,\theta)$$

$$u(x,z,\theta) = -\sum_{i=1}^{N} \ln p(x_i, z_i|\theta)$$

Approximate Inference: importance sampling

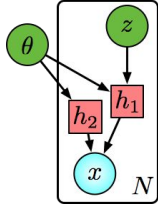


Let's approximate some integrals...

$$\{z_k\} \sim q(z)$$

$$M(x,\theta) \approx -\ln \sum_{k=1}^{K} e^{-\mathcal{F}(x,z_k,\theta)} + \ln K$$

Approximate Inference: Laplace/saddle point approx

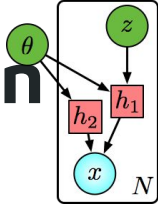


$$u(x,z,\theta) \approx u(x,\mu,\theta) + J^T(z-\mu) + \frac{1}{2}(z-\mu)^T H(z-\mu)$$

$$\begin{aligned}\int dz e^{-u(x,z,\theta)} &\approx e^{-u(x,\mu,\theta)} \int dz e^{-J^T(z-\mu)-\frac{1}{2}(z-\mu)^T H(z-\mu)} \\ &\approx e^{-u(x,\mu,\theta)+\frac{1}{2}J^T H J} \det(2\pi H)^{\frac{1}{2}}\end{aligned}$$

$$M(x,\theta) \approx u(x,\mu,\theta) - \frac{1}{2}J^T H J - \frac{1}{2}\ln\det(2\pi H)$$

Approximate Inference: Perturbative expansion

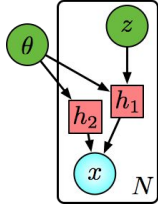


$$e^{-M(x,\theta)} = \mathbb{E}_{z\sim q}[e^{-\mathcal{F}(x,z,\theta)}]$$

$$e^{-\mathcal{F}(x,z,\theta)} = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!}\mathcal{F}(x,z,\theta)^k$$

$$e^{-M(x,\theta)} = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!}\mathbb{E}_{z\sim q}[\mathcal{F}(x,z,\theta)^k]$$

Approximate Inference: Variational approx



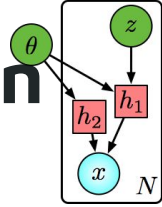
$$M(x,\theta) = -\ln \int dz q(z) e^{-\mathcal{F}(x,z,\theta)}$$

Exp is concave

$$\leqslant \int dz q(z) \mathcal{F}(x,z,\theta) \qquad \forall q \neq 0$$

$$M(x,\theta) \leqslant \underset{q}{\operatorname{argmin}} \int dz q(z) \mathcal{F}(x,z,\theta) \quad s.t. \int dz q(z) = 1$$

Approximate Inference: Relation to information theory



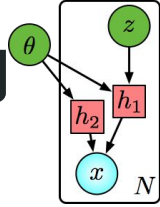
$$\mathbb{E}_{z\sim q}[\mathcal{F}(x,$$ $$) - \ln \pi(z_i)]$$

$$\mathrm{D}[q_i||\pi]]$$

gularizer

KLD(q||p) is the number of bits necessary to transmit the distribution q to a receiver that already knowns p

Approximate Inference: Variational mean-field approx



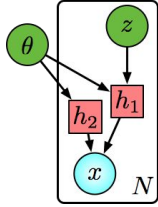
$$q(z) = \prod_{i=1}^{N} q_i(z_i) \approx \prod_{i=1}^{N} \prod_{j=1}^{d_z} q_{ij}(z_{ij})$$

$$M(x,\theta) \leqslant \operatorname*{argmin}_{q} \int dz q(z) \mathcal{F}(x,z,\theta)$$

Fixed-point equations

$$q^{\star}_{ij}(z_{ij}) \propto e^{-\mathbb{E}_{z_{/ij} \sim q^{\star}}[\mathcal{F}(x,z,\theta)]}$$

Variational approx: Amortized inference



$$q(z) = \prod_{i=1}^{N} \left(\right. \qquad z)\mathcal{F}(x, z, \theta)$$

Requires solving optimization for every new data-point x.

Solution:

Parameters phi are shared across all data-points => consistency, fast convergence

$$M(x,\theta) \leqslant \underset{q}{\operatorname{argmin}} \int dz q(z)\mathcal{F}(x,z,\theta) \Rightarrow M(x,\theta) \leqslant \underset{\phi}{\operatorname{argmin}} \int dz \prod_i q_\phi(z_i|x_i)\mathcal{F}(x,z,\theta)$$

Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082v3
Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114

Variational approx: Amortization & reparametrization

$$\underset{\phi}{\operatorname{argmin}} \int dz \prod_i q_\phi(z_i|x_i)\mathcal{F}(x,z,\theta) = \operatorname{argmin} \sum \mathbb{E}_{z_i \sim q_\phi}[\mathcal{F}(x_i, z_i, \theta)]$$

idea: find an invertible transformation g such that

$$\mathbb{E}_{z_i \sim q_\phi}[\mathcal{F}(x_i, z_i, \theta)] = \mathbb{E}_{\xi_i \sim \lambda}[\mathcal{F}(x_i, g(\xi_i, \phi), \theta)]$$

then

$$\nabla_\phi \mathbb{E}_{z_i \sim q_\phi}[\mathcal{F}(x_i, z_i, \theta)] = \nabla_\phi \mathbb{E}_{\xi_i \sim \lambda}[\mathcal{F}(x_i, g(\xi_i, \phi), \theta)] = \mathbb{E}_{\xi_i \sim \lambda}[\nabla_\phi \mathcal{F}(x_i, g(\xi_i, \phi), \theta)]$$

Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082v3
Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114

Variational approx: Amortization & reparametrization. Gaussian example:

$$q(z|x) = \mathcal{N}(z|\mu_\phi(x), \Sigma_\phi(x) = C_\phi(x)^T C_\phi(x))$$

then

$$\mathbb{E}_{z\sim q}[f(z)] = \mathbb{E}_{\xi\sim\mathcal{N}(0,\mathbb{I})}[f(\mu_\phi(x) + C_\phi(x)^T\xi)]$$

Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082v3
Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114

Variational approx: Amortization & reparametrization. Gaussian gradients example:

$$\mathbb{E}_{z\sim q}[f(z)] = \mathbb{E}_{\xi\sim\mathcal{N}(0,\mathbb{I})}[f(\mu_\phi(x) + C_\phi(x)^T\xi)]$$

Only using first order information

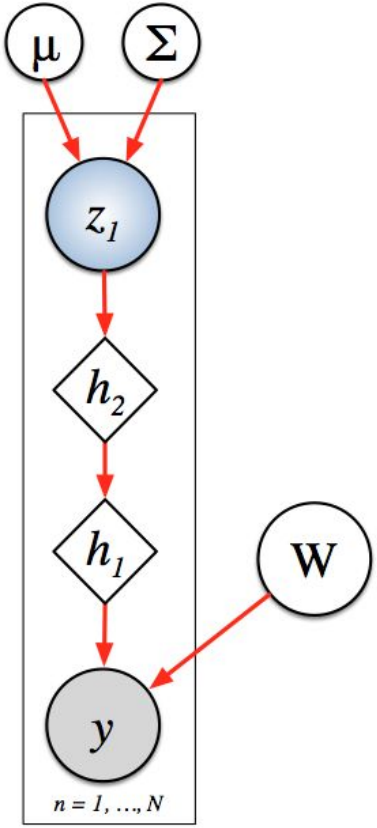
$$\nabla_\phi \mathbb{E}_{\xi\sim\mathcal{N}(0,\mathbb{I})}[f(\mu_\phi(x) + C_\phi(x)^T\xi)] = \mathbb{E}_{\xi\sim\mathcal{N}(0,\mathbb{I})}[J^T\{\nabla_\phi\mu_\phi(x) + \nabla_\phi C_\phi(x)^T\xi\}]$$

Using second order information

$$\mathbb{E}_{z\sim q}[J^T\nabla_\phi\mu_\phi(x) + \mathrm{Tr}[HC_\phi(x)\nabla_\phi C_\phi(x)]]$$

Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082v3
Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114

Deep Latent Generative Models / VAES



- Describes a causal process by which data ***y*** is generated.
- Layers of stochastic variables **z**
- Conditional probabilities using nc
  on deep neural networks.
- Statistical problem of density esti

Latent Variables (Stochastic layers)

$$\mathbf{z}_l \sim \mathcal{N}(\mathbf{z}_l | f_l(\mathbf{z}_{l+1}), \boldsymbol{\Sigma}_l)$$

$$f_l(\mathbf{z}) = \sigma(\mathbf{W} h(\mathbf{z}) + \mathbf{b})$$

Deterministic layers

$$h_i(\mathbf{x}) = \sigma(\mathbf{A}\mathbf{x} + \mathbf{c})$$

Observation Model

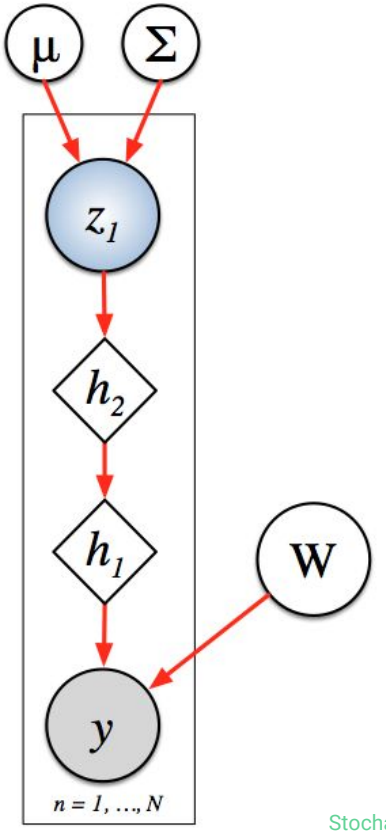
$$\boldsymbol{\eta} = \mathbf{W}\mathbf{h}_\mathbf{1} + \mathbf{b}$$

$$\mathbf{y} \sim \text{Expon}(\mathbf{y} | \boldsymbol{\eta})$$

Can also use non-exponential family.

Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082v3
Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114

Two processes to understand



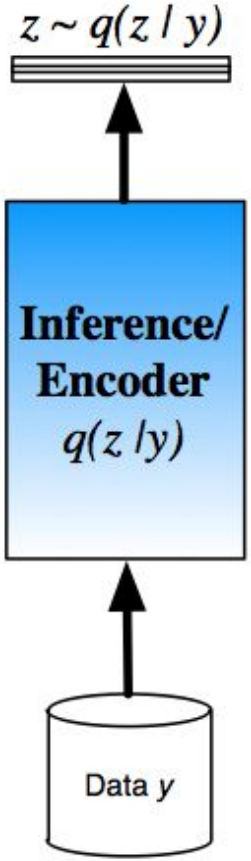
1. Generate new data

- Sample new data given underlying causes.

2. Explain observed data

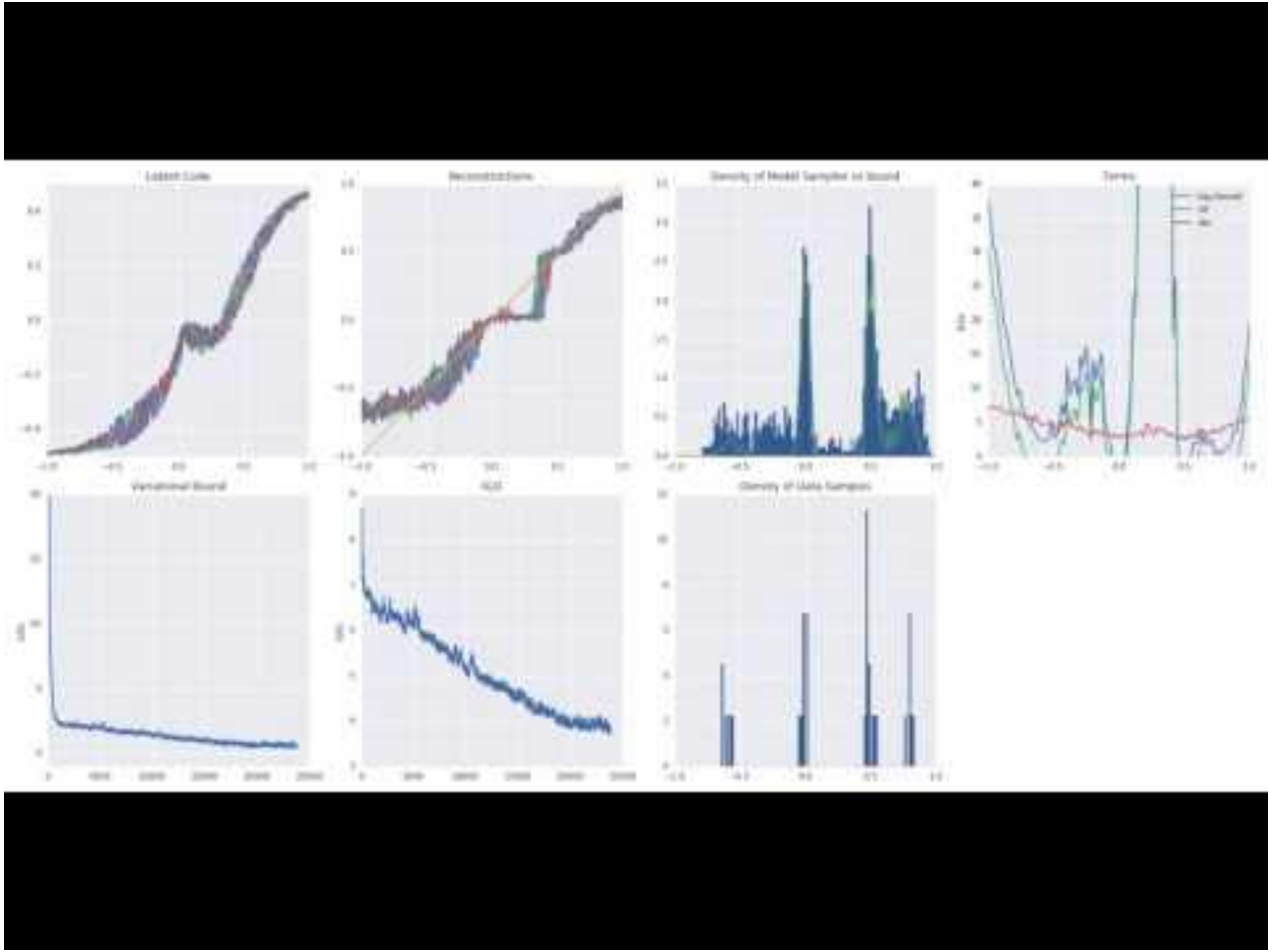
- Recognition or inference of observed data to obtain the posterior distribution.

Generative model and posterior representation are connected by variational inference.



Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082v3
Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114

Demo



Improving Inference

Desired properties of an inference model

Flexible:

- Can increase complexity on demand

Fast at training time:

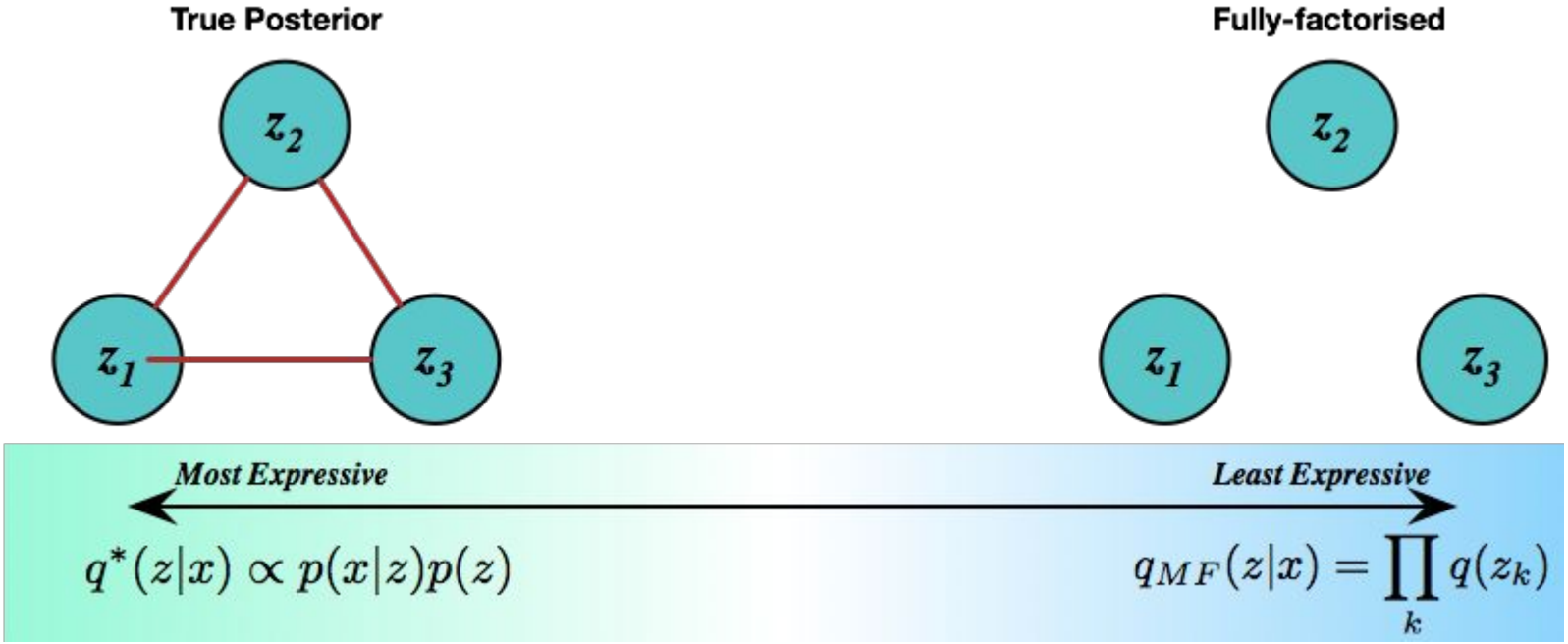
- Cheap to sample from
- Unbiased approximation to entropy derivatives known

Fast at test time:

- Should not rely on the generative model’s likelihoods or derivatives at test time

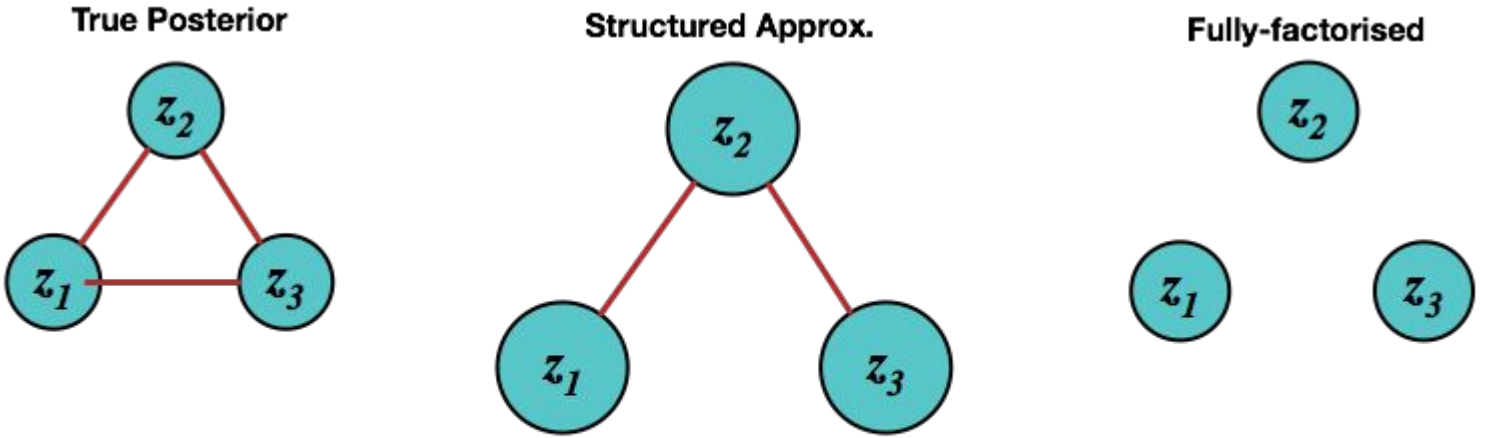
Richer Families of Posteriors

Two high-level goals: □ Build richer approximate posterior distributions. □ Maintain computational efficiency and scalability

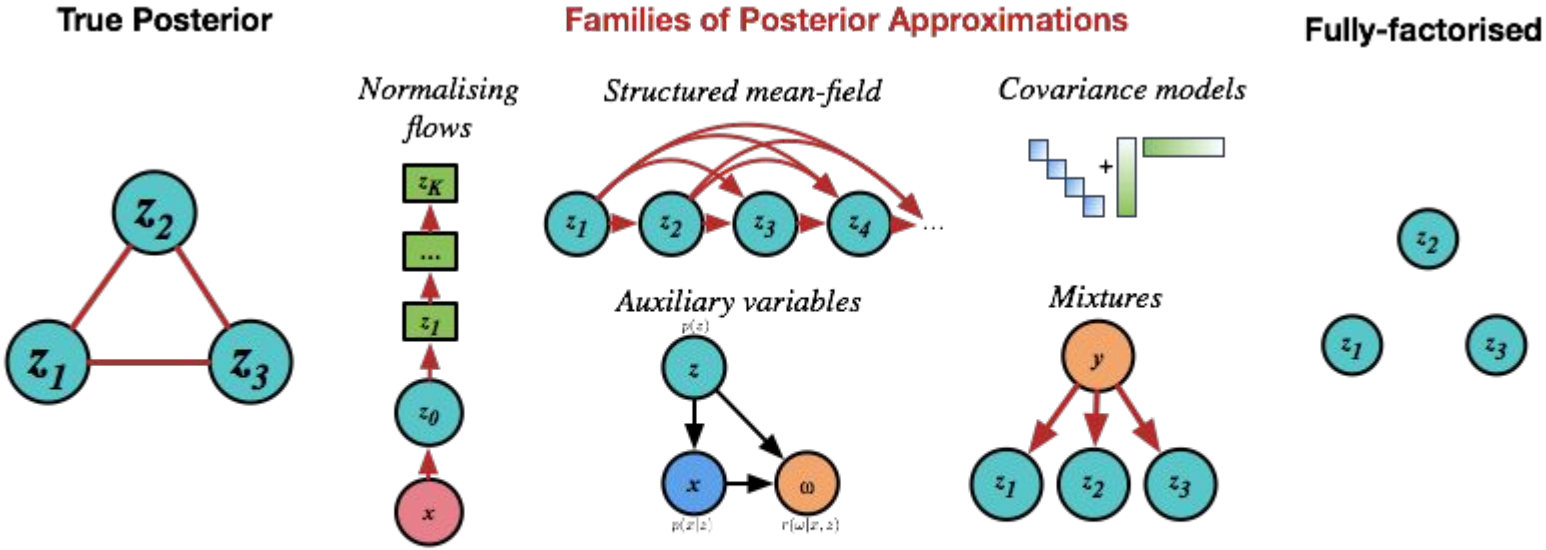


Same as the problem of specifying a model of the data itself

Richer Families of Posteriors

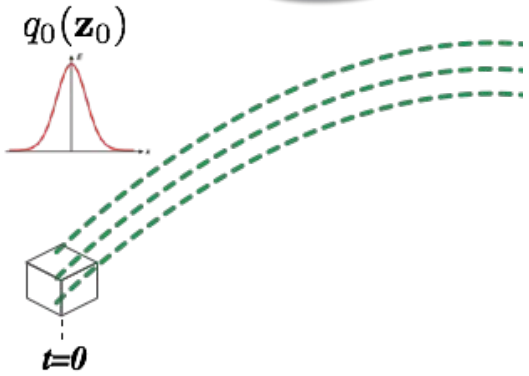


Richer Families of Posteriors



Amortized inference with Normalizing Flows

$$q(\mathbf{z}') = q(\mathbf{z}) \left| \det \frac{\partial f}{\partial \mathbf{z}} \right|^{-1}$$



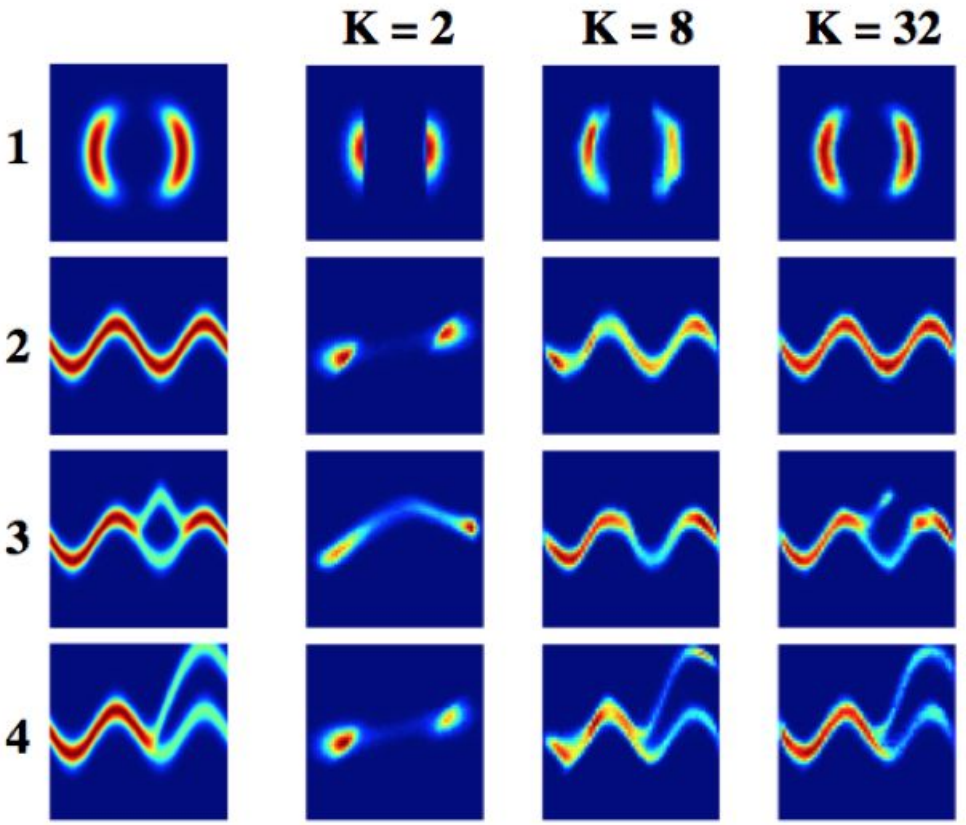
Variational Inference with Normalizing Flows https://arxiv.org/abs/1505.05770

Example: Planar Flow in $\mathbb{R}^2$

Planar Flow

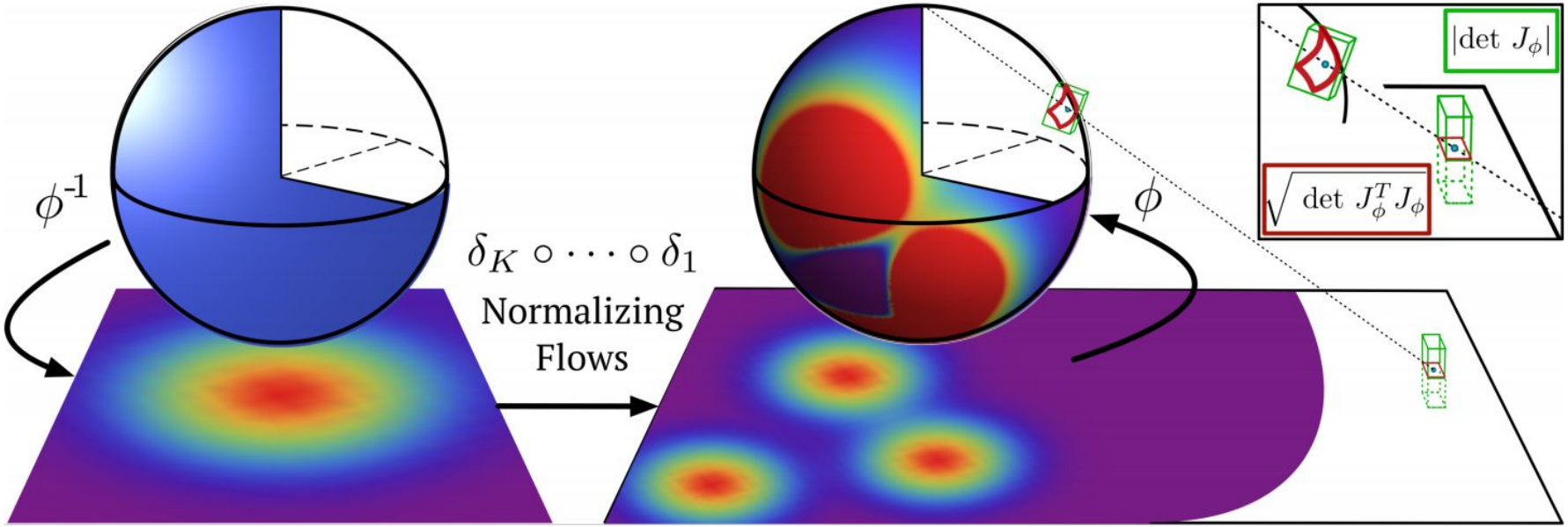
$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^\top \mathbf{z} + b)$$

$$\det \left| \frac{\partial f}{\partial \mathbf{z}} \right| = |\det(\mathbf{I} + \mathbf{u}\psi(\mathbf{z})^\top)| = |1 + \mathbf{u}^\top \psi(\mathbf{z})|$$



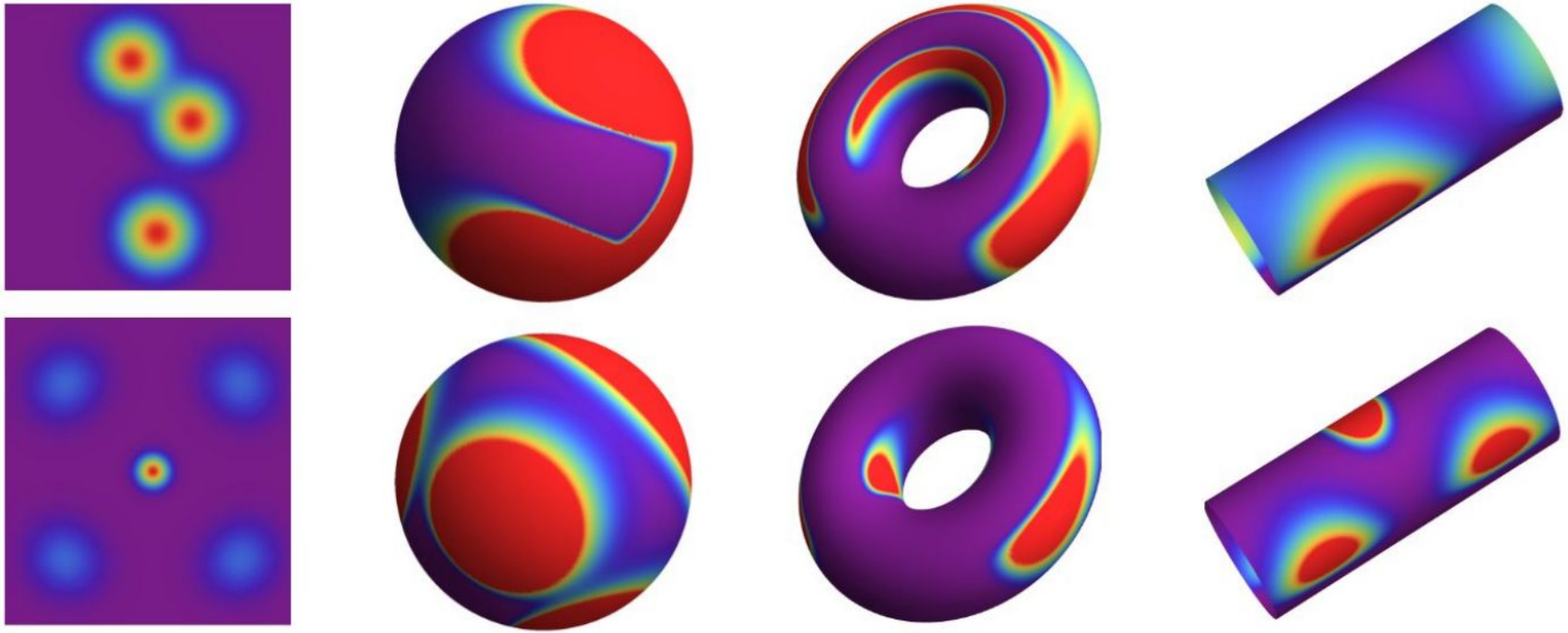
Variational Inference with Normalizing Flows https://arxiv.org/abs/1505.05770

Normalizing Flows on non-Euclidean Manifolds



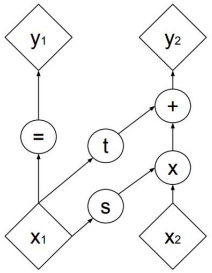
Normalizing Flows on Riemannian Manifolds https://arxiv.org/abs/1611.02304

Normalizing Flows on non-Euclidean Manifolds

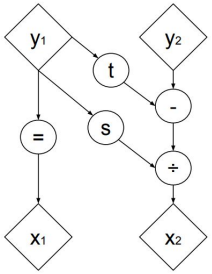


Normalizing Flows on Riemannian Manifolds https://arxiv.org/abs/1611.02304

Improved Normalizing Flows



(a) Forward propagation



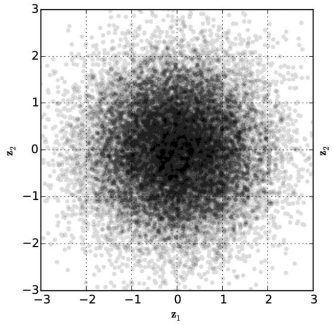
(b) Inverse propagation

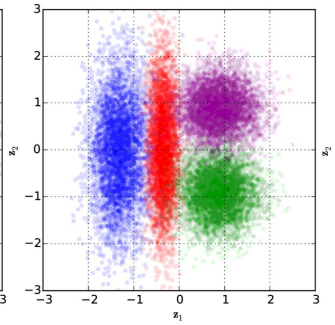
$$[\mathbf{m}_t, \mathbf{s}_t] \leftarrow \texttt{AutoregressiveNN}[t](\mathbf{z}_t, \mathbf{h}; \boldsymbol{\theta})$$

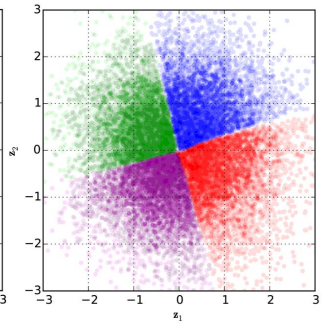
$$\boldsymbol{\sigma}_t = \text{sigmoid}(\mathbf{s}_t)$$
$$\mathbf{z}_t = \boldsymbol{\sigma}_t \odot \mathbf{z}_{t-1} + (1 - \boldsymbol{\sigma}_t) \odot \mathbf{m}_t$$

Dataset	PixelRNN [46]	Real NVP	Conv DRAW [22]	IAF-VAE [34]
CIFAR-10	3.00	3.49	< 3.59	< 3.28
Imagenet (32×32)	3.86 (3.83)	4.28 (4.26)	< 4.40 (4.35)	
Imagenet (64×64)	3.63 (3.57)	3.98 (3.75)	< 4.10 (4.04)	
LSUN (bedroom)		2.72 (2.70)		
LSUN (tower)		2.81 (2.78)		
LSUN (church outdoor)		3.08 (2.94)		
CelebA		3.02 (2.97)		

Table 1: Bits/dim results for CIFAR-10, Imagenet, LSUN datasets and CelebA. Test results for CIFAR-10 and validation results for Imagenet, LSUN and CelebA (with training results in parenthesis for reference).







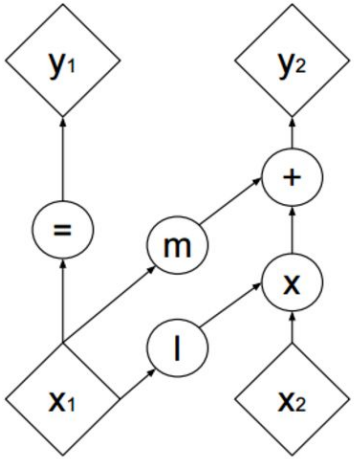
(a) Prior distribution (b) Posteriors in standard VAE (c) Posteriors in VAE with IAF

Density estimation using Real NVP https://arxiv.org/abs/1605.08803

Improving Variational Inference with Inverse Autoregressive Flow https://arxiv.org/abs/1606.0493

Real NVP: core idea

(real valued non-volume preserving transformations)



$$y_{1:d} = x_{1:d}$$

$$d)) + t(x_{1:d}),$$

A complex, invertible, transformation with triangular Jacobian

$$\frac{\partial y}{\partial x^T} = \begin{bmatrix} \mathbb{I}_d & 0 \\ \frac{\partial y_{d+1:D}}{\partial x_{1:d}^T} & \operatorname{diag}\left(\exp\left[s\left(x_{1:d}\right)\right]\right) \end{bmatrix}$$

Density estimation using Real NVP https://arxiv.org/abs/1605.08803

Real NVP: key property

(real valued non-volume preserving transformations)

$$\begin{cases} y_{1:d} & = x_{1:d} \\ y & \end{cases}$$

$$\ldots)) + t(x_{1:d})$$

The transformation is easily and analytically invertible!

$$\Leftrightarrow \begin{cases} x_{1:d} & = y_{1:d} \\ x_{d+1:D} & = \left(y_{d+1:D} - t(y_{1:d})\right) \odot \exp\left(-s(y_{1:d})\right), \end{cases}$$

Density estimation using Real NVP https://arxiv.org/abs/1605.08803

Real NVP: results

(real valued non-volume preserving transformations)

Dataset	PixelRNN [46]	Real NVP	Conv DRAW [22]	IAF-VAE [34]
CIFAR-10	3.00	3.49	< 3.59	< 3.28
Imagenet (32×32)	3.86 (3.83)	4.28 (4.26)	< 4.40 (4.35)	
Imagenet (64×64)	3.63 (3.57)	3.98 (3.75)	< 4.10 (4.04)	
LSUN (bedroom)		2.72 (2.70)		
LSUN (tower)		2.81 (2.78)		
LSUN (church outdoor)		3.08 (2.94)		
CelebA		3.02 (2.97)		

Density estimation using Real NVP https://arxiv.org/abs/1605.08803

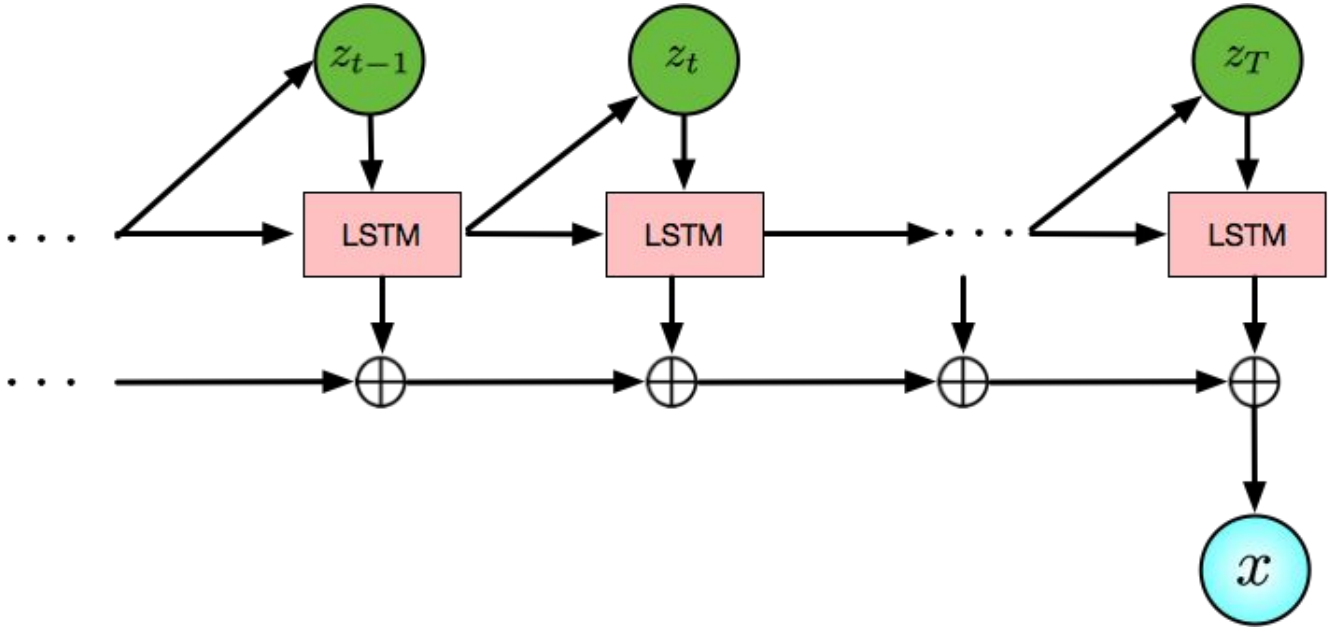
Real NVP: results

(real valued non-volume preserving transformations)

Density estimation using Real NVP https://arxiv.org/abs/1605.08803

Recurrent VAE: Iteratively Generating Data

Forward Model



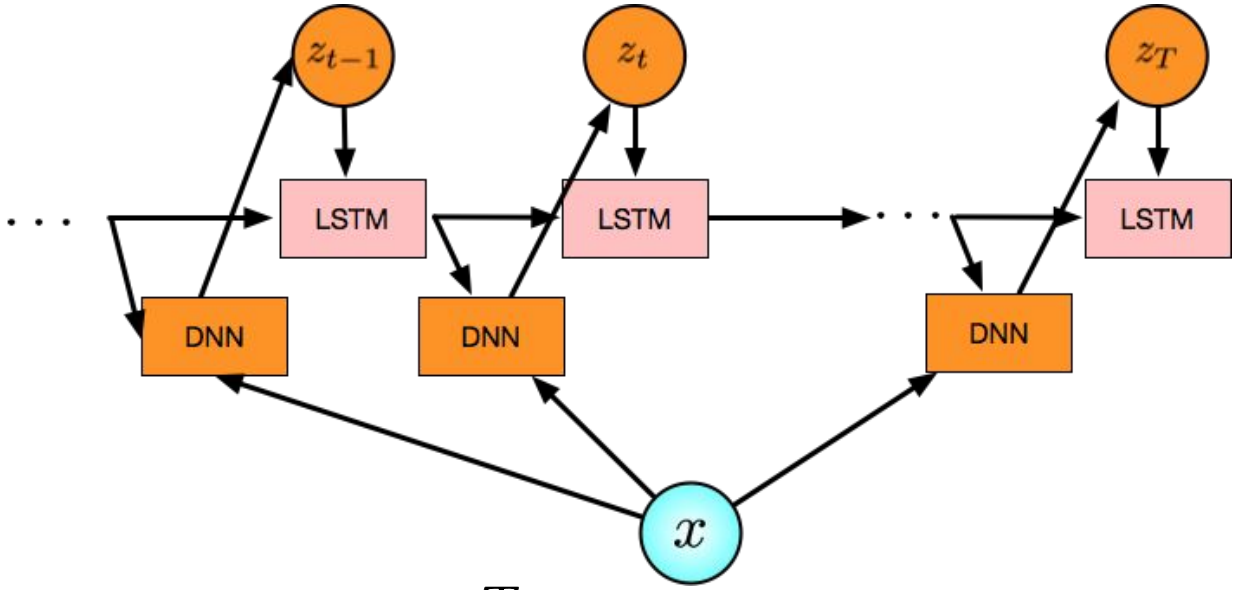
$$p(X, Z) = p(x_T|z_{0,\ldots,T}) \prod_{t=0}^{T} \pi(z_t|z_{0,\ldots,(t-1)})$$

Towards Conceptual Compression https://arxiv.org/abs/1604.08772

DRAW: A Recurrent Neural Network For Image Generation https://arxiv.org/abs/1502.04623

Recurrent VAE: Iteratively Generating Data

Inference Model



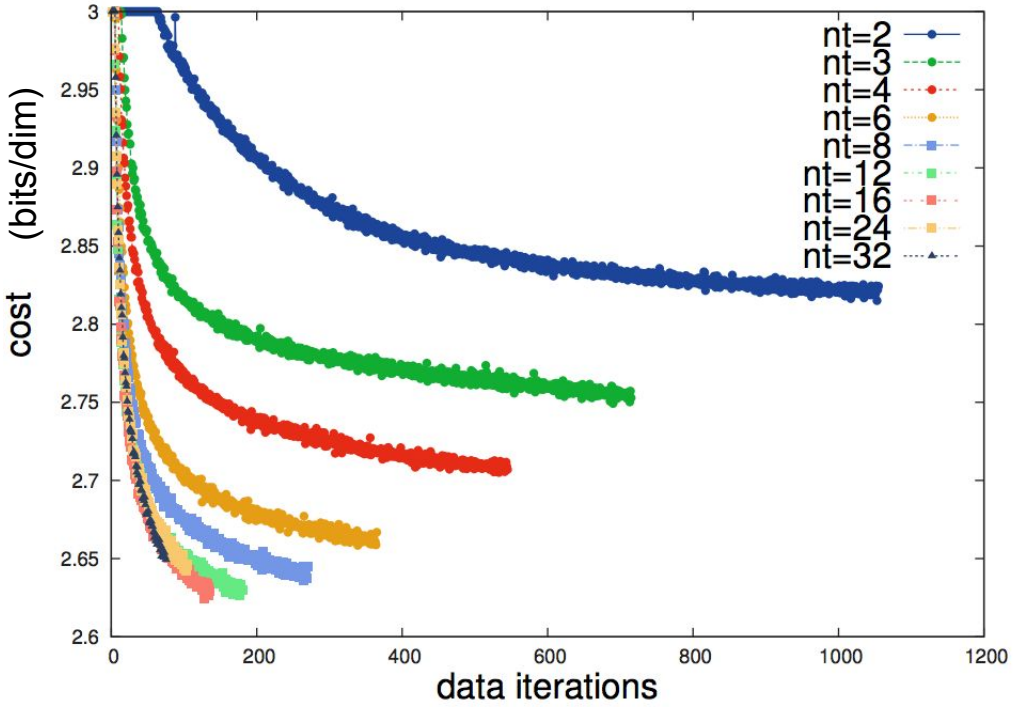
$$q(Z|x) = \prod_{t=0}^{T} q(z_t|z_{0,\ldots,(t-1)}, x)$$

Towards Conceptual Compression https://arxiv.org/abs/1604.08772

DRAW: A Recurrent Neural Network For Image Generation https://arxiv.org/abs/1502.04623

Recurrent VAE: Iteratively Generating Imagenet

- Extend the process of generation and recognition to be sequential using recurrent neural networks.
- Inference and generation are convolutional maps.



Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Recurrent VAE: Iteratively Generating Imagenet

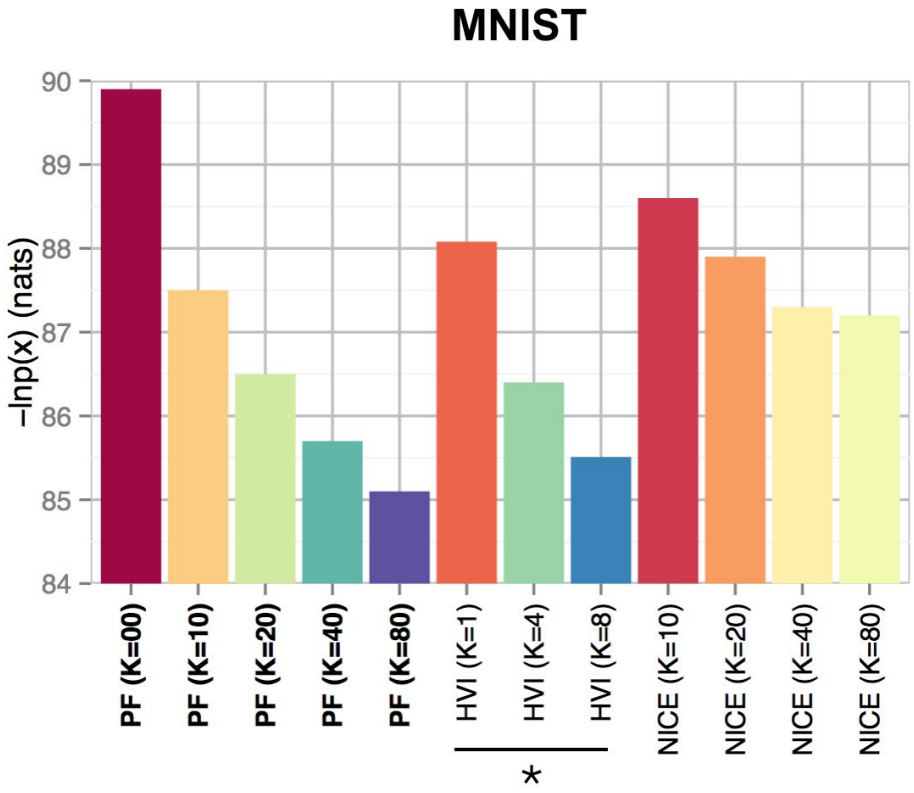
Recurrent VAE

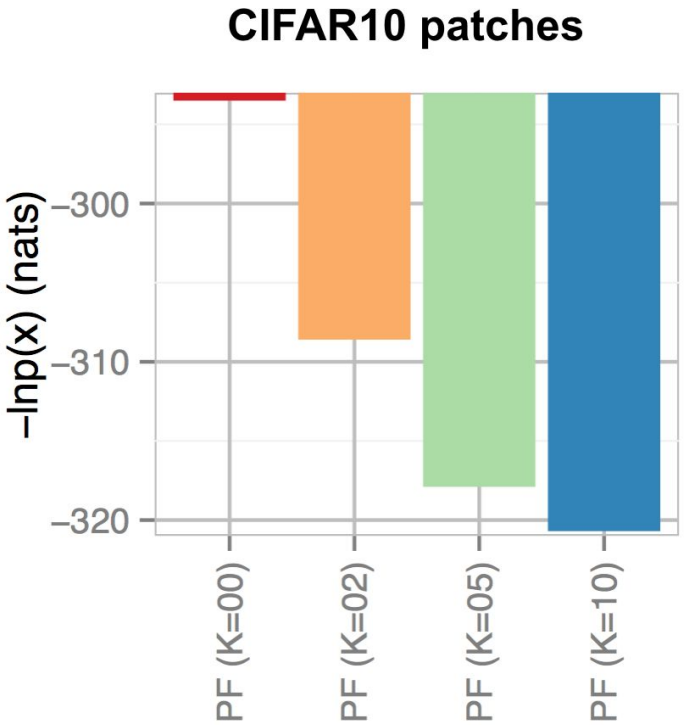
Pixel RNN

ompression I

Applications of VAEs

Amortized inference with Normalizing Flows





Variational Inference with Normalizing Flows https://arxiv.org/abs/1505.05770

Making DNNs a little bit more Bayesian

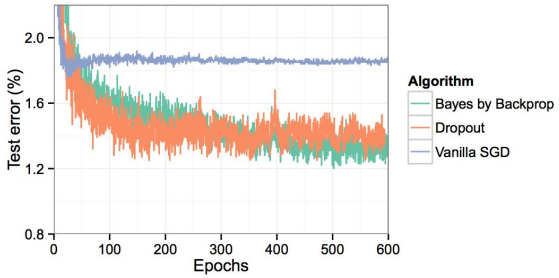


Figure 2. Test error on MNIST as training progresses.

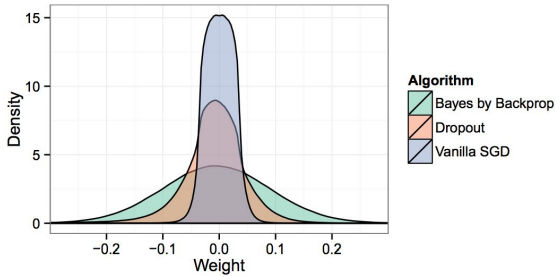
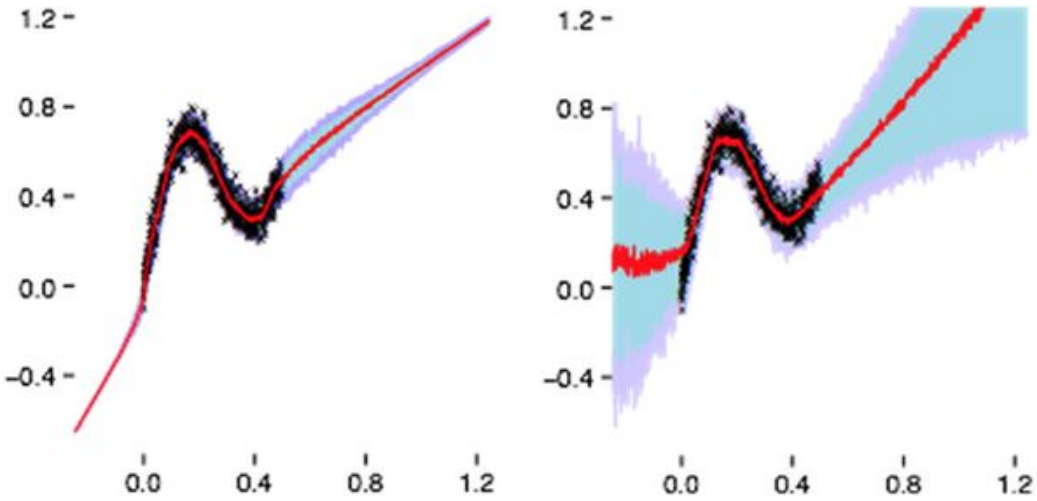


Figure 3. Histogram of the trained weights of the neural network, for Dropout, plain SGD, and samples from Bayes by Backprop.



Weight Uncertainty in Neural Networks https://arxiv.org/abs/1505.05424

Making DNNs a little bit more Bayesian

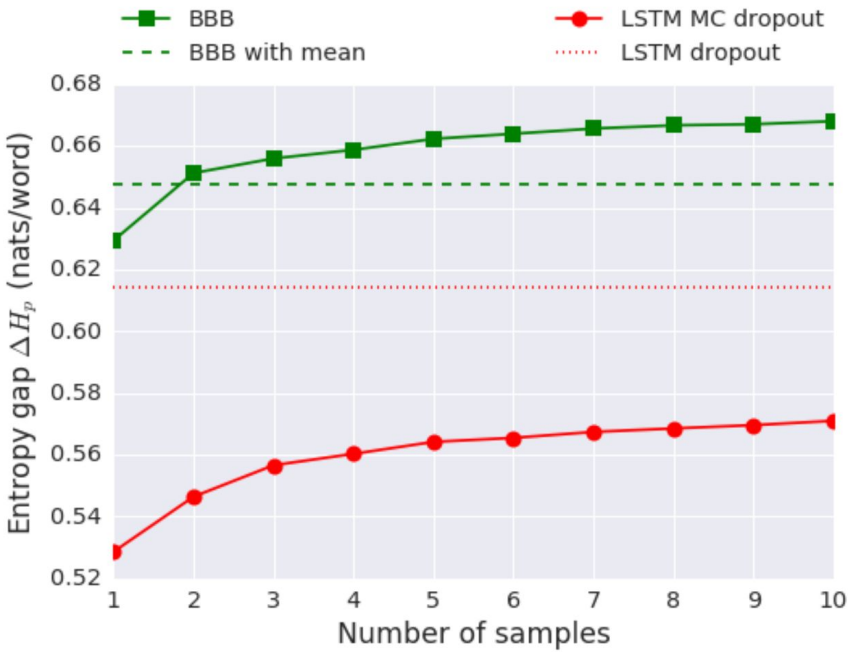


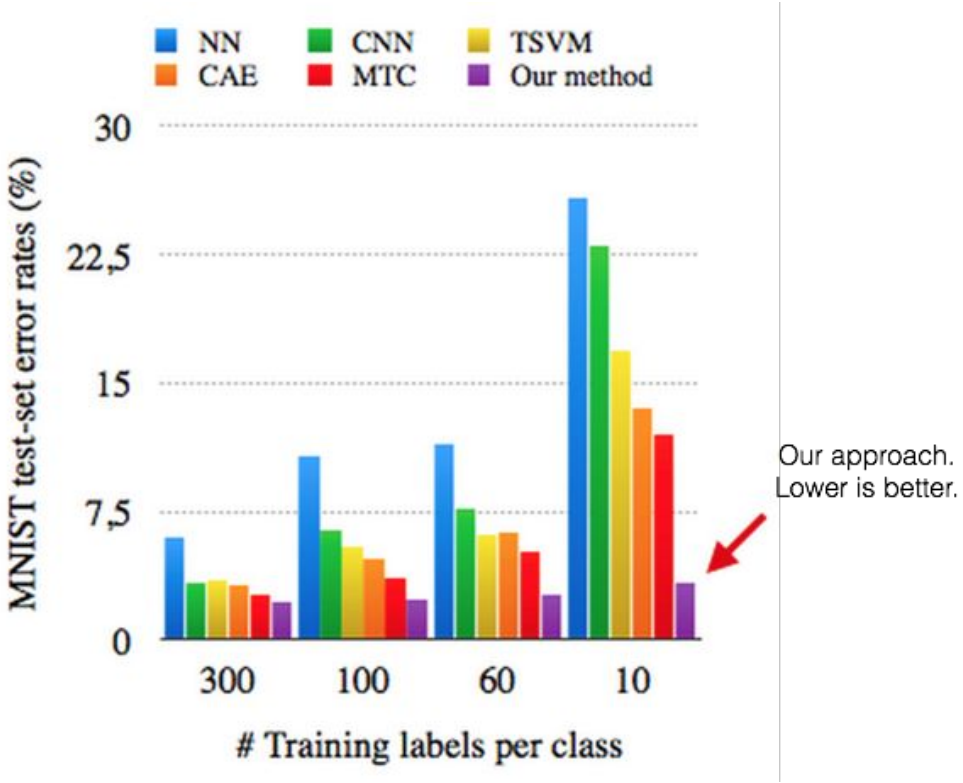
Figure 2. Entropy gap ΔH_p (eq. (12)) between reversed and regular Penn Treebank test sets $\times$ number of samples.

Bayesian Recurrent Neural Networks https://arxiv.org/abs/1704.02798

Improving semi-supervised with unsupervised features

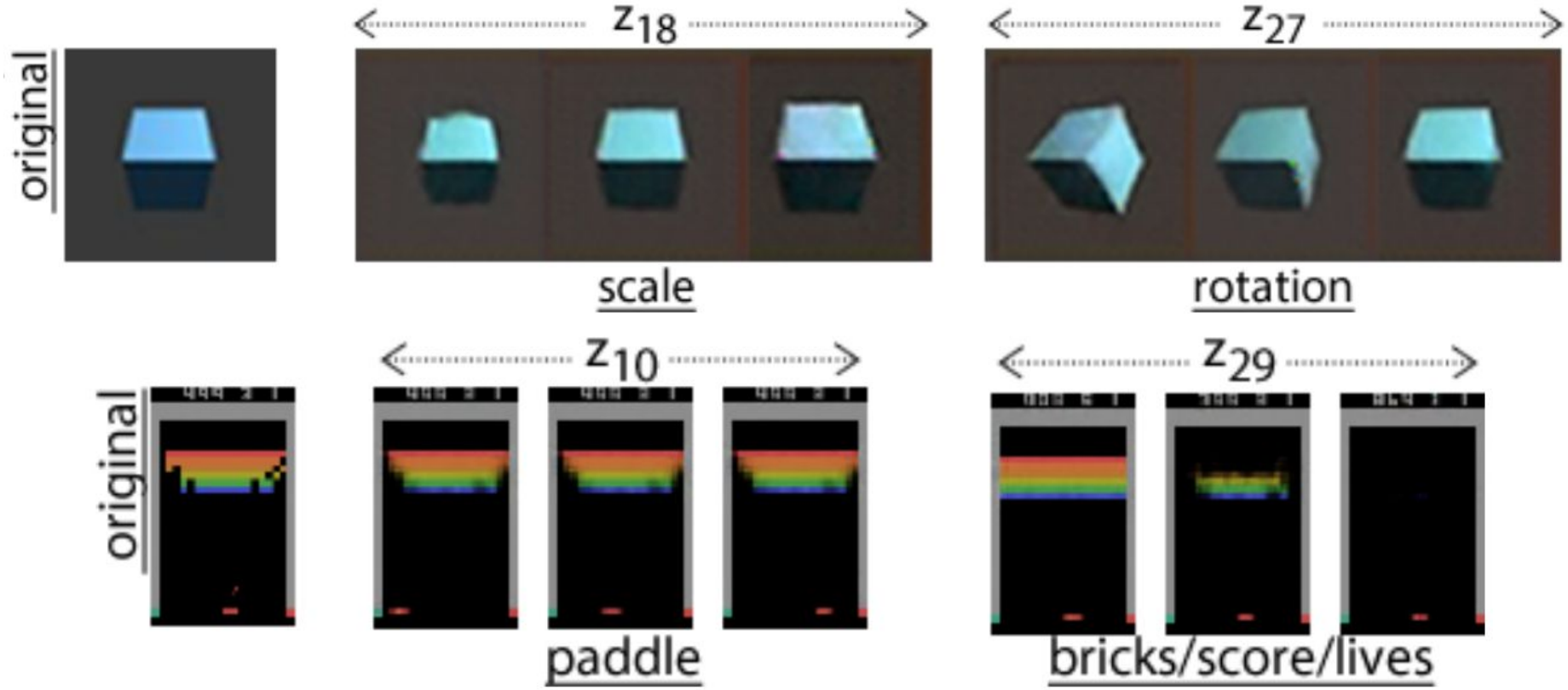
Maintaining uncertainty allows us to:

- Speed up learning,
- Use less data, and
- Obtain better predictions.



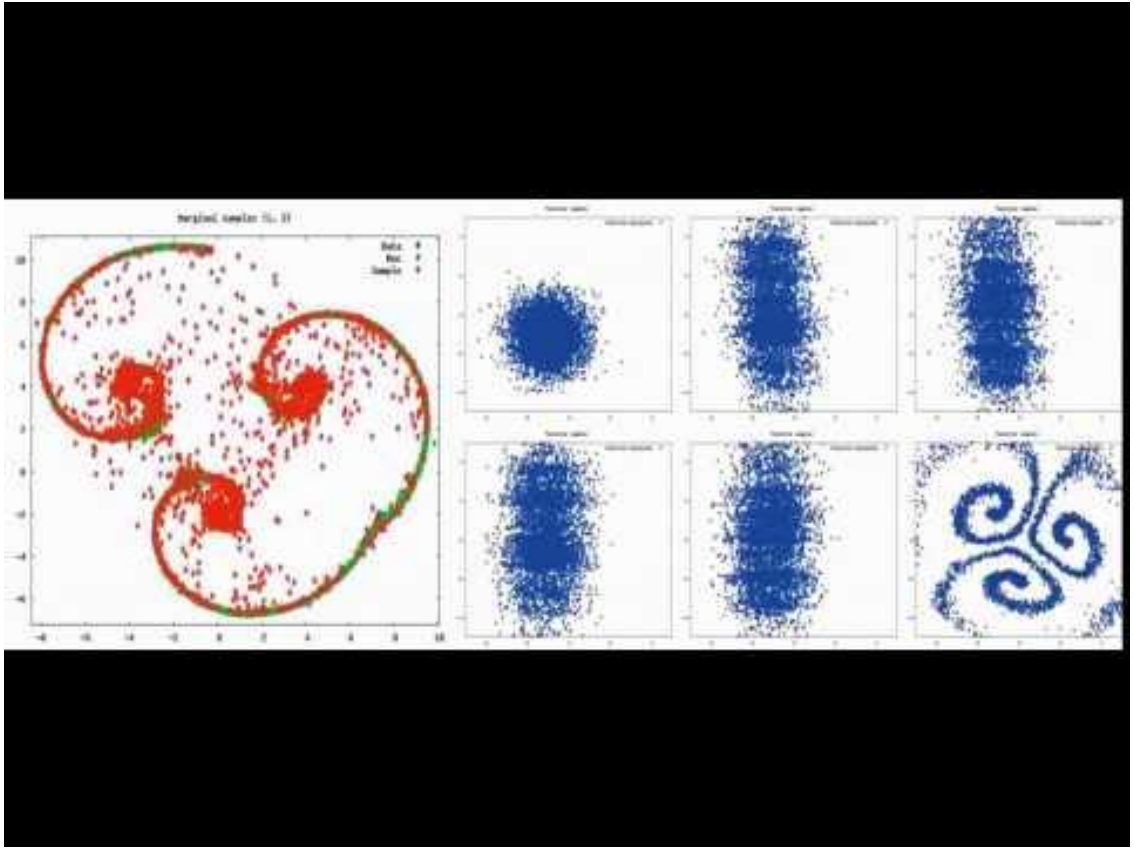
Semi-Supervised Learning with Deep Generative Models http://arxiv.org/abs/1406.5298

Extracting relevant factors of variation



Early Visual Concept Learning with Unsupervised Deep Learning https://arxiv.org/abs/1606.05579

Discovering the dimensionality of the data



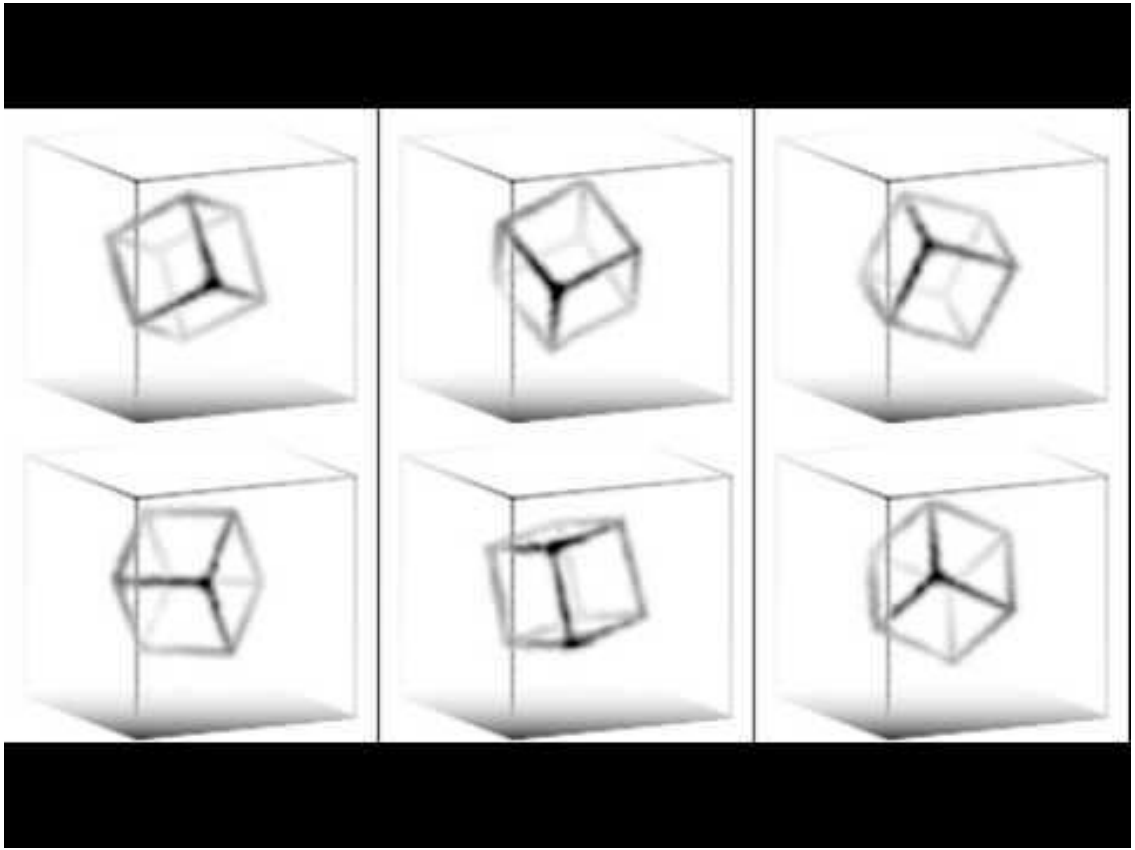
Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/pdf/1401.4082v3.pdf

Missing data imputation for 2D images

Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/pdf/1401.4082v3.pdf

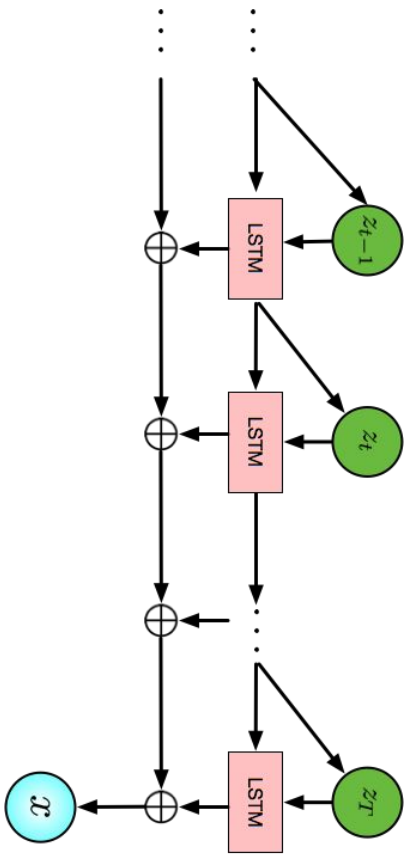
Applications of Recurrent VAEs

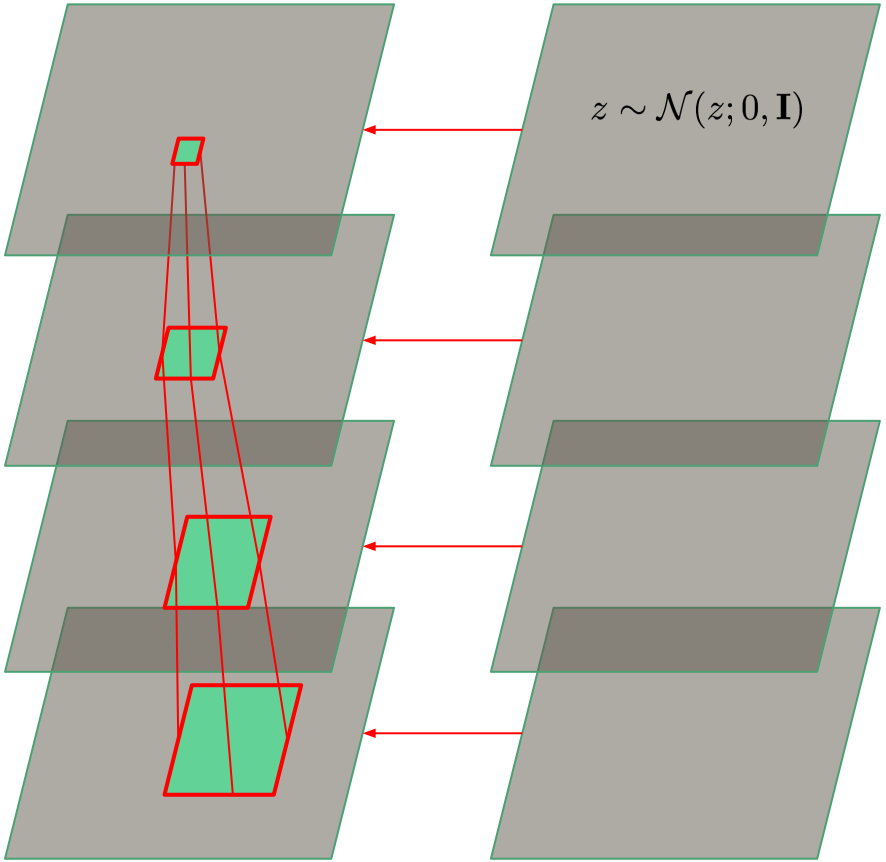
Missing data imputation for 3D images



Unsupervised Learning of 3D Structure from Images https://arxiv.org/abs/1607.00662

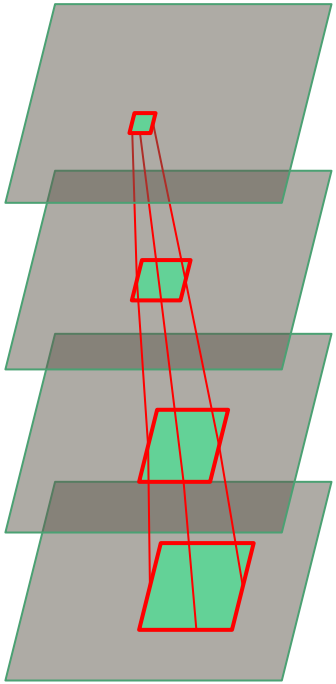
Understanding Images at multiple scales

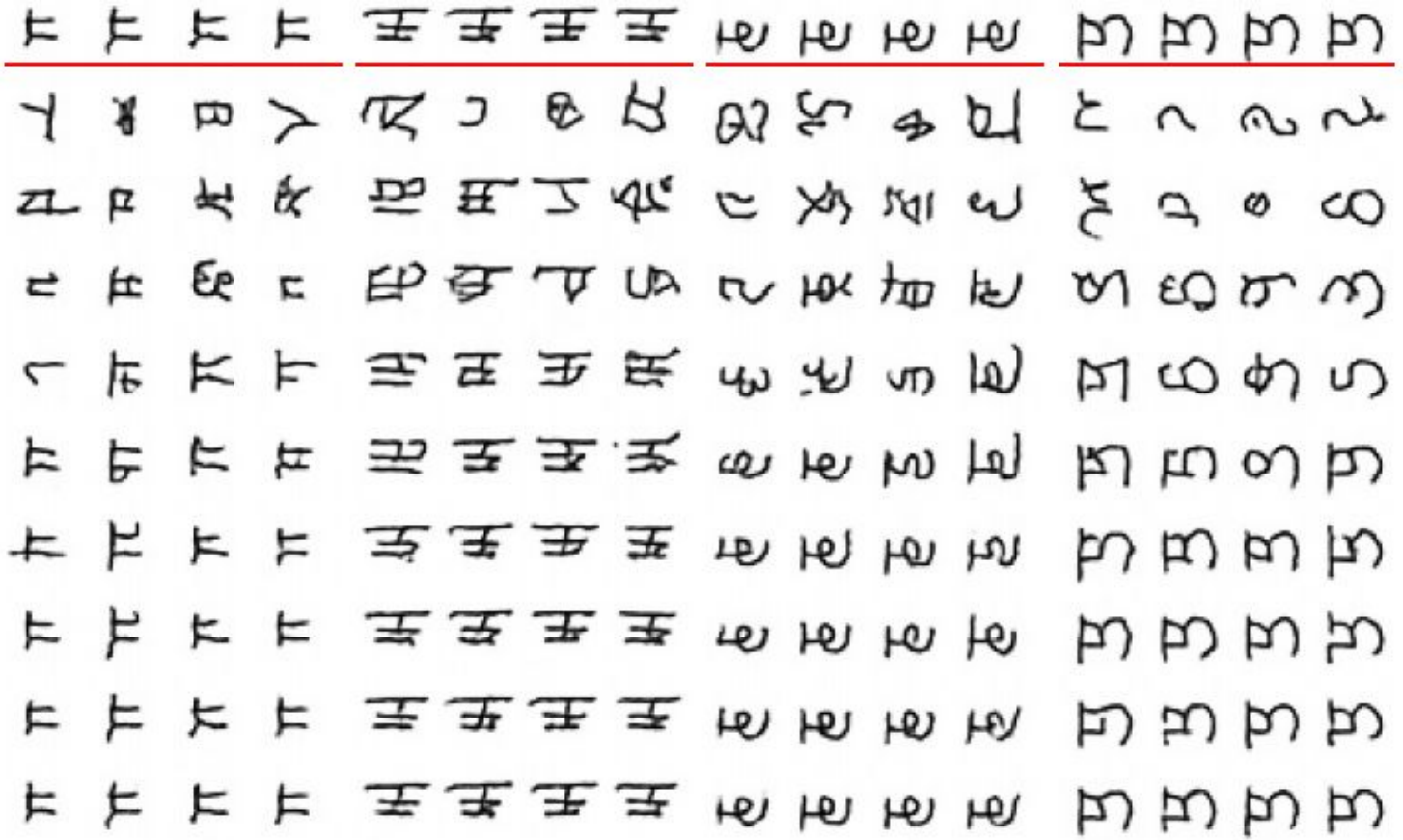




Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Understanding Images at multiple scales





Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Lossy Compression

- Assuming we have access to the joint density

$$p(x, z) = p(x|z)\pi(z)$$

- A new sample **x** can be compressed by encoding it with the posterior density

$$p(z|x)$$

- The number of bits necessary to communicate this density the KL-divergence

$$\frac{\mathrm{KL}(p(z|x); \pi(z))}{\ln(2)}$$

Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Compression Rate

- Given the model $p(x, z) = p(x|z)\pi(z)$

- Where $x \in \mathbb{R}^{d_x}$

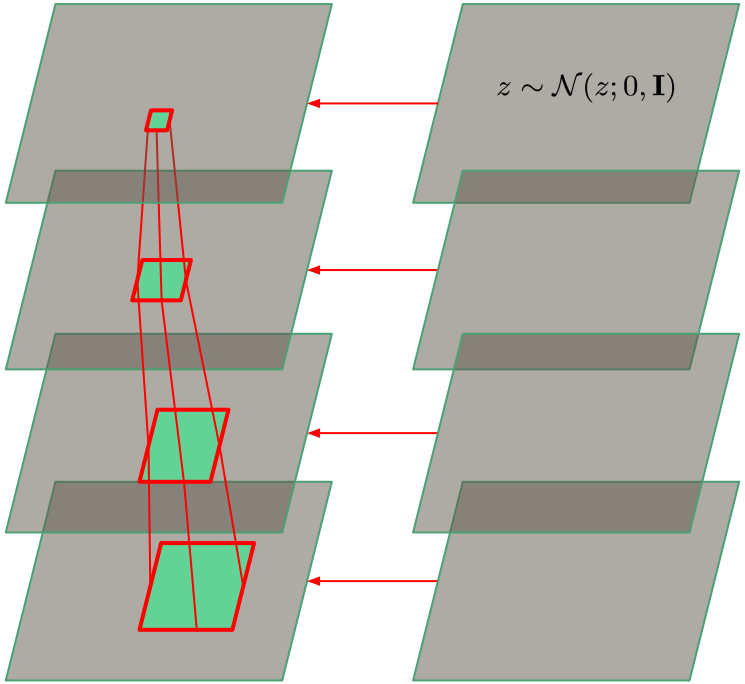
- We define compression rate as

$$r = \frac{1}{d}\mathbb{E}_{p(x)}\left[\frac{\mathrm{KL}(p(z|x); \pi(z))}{\ln(2)}\right]$$

Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Understanding Images at multiple scales: How many bits are stored at each level?

$$\mathcal{F}(x) = \mathbf{E_q}[-\ln \mathbf{p(x|z)} + \sum_{\mathbf{t=1}}^{\mathbf{L}} \mathrm{KL}(\mathbf{q_t(z_t|z_{<t}, x)}; \mathbf{p(z_t)})]$$



$$r_1 = \frac{1}{d}\mathbb{E}_{p(x)}\left[\mathrm{KL}(q_1(z_1|x)); p(z_1)\right]$$

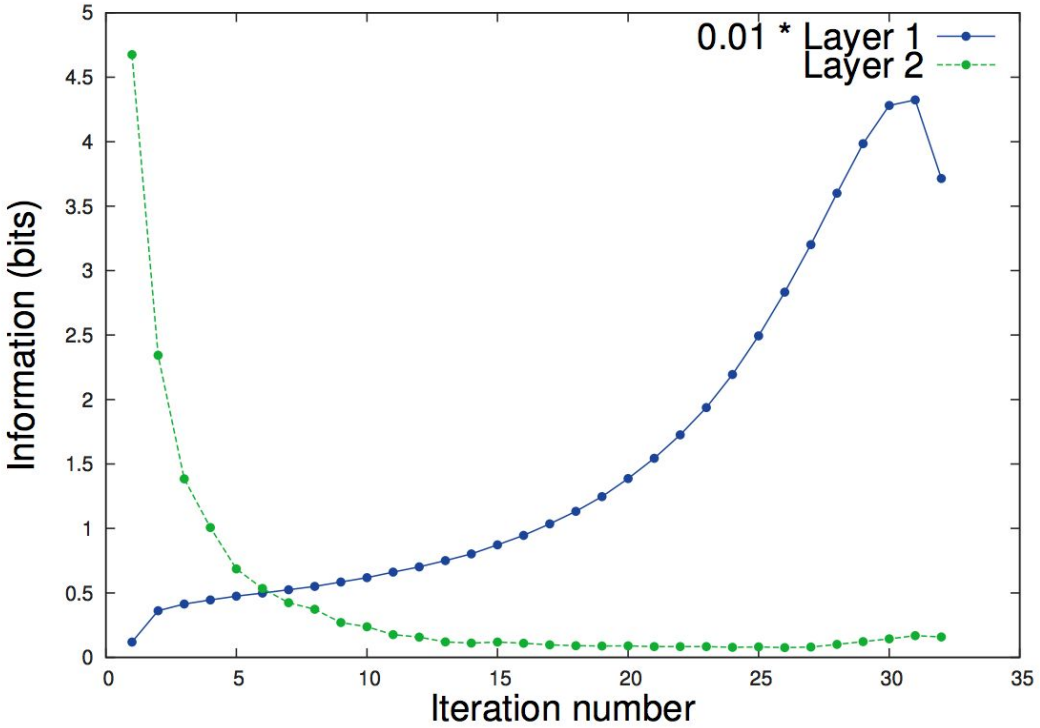
$$r_2 = \frac{1}{d}\mathbb{E}_{p(x)}\left[\mathrm{KL}(q_1(z_1|x)); p(z_1)\right] + \frac{1}{d}\mathbb{E}_{p(x)}\left[\mathrm{KL}(q_2(z_2|z_{<2}, x)); p(z_2)\right]$$

$$\vdots$$

$$r_k = \frac{1}{d}\sum_k \mathbb{E}_{p(x)}\left[\mathrm{KL}(q_k(z_k|z_{<k}, x)); p(z_k)\right]$$

Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Understanding Images at multiple scales: How many bits are stored at each level?



Towards Conceptual Compression https://arxiv.org/abs/1604.08772

Understanding Images at multiple scales

Original images

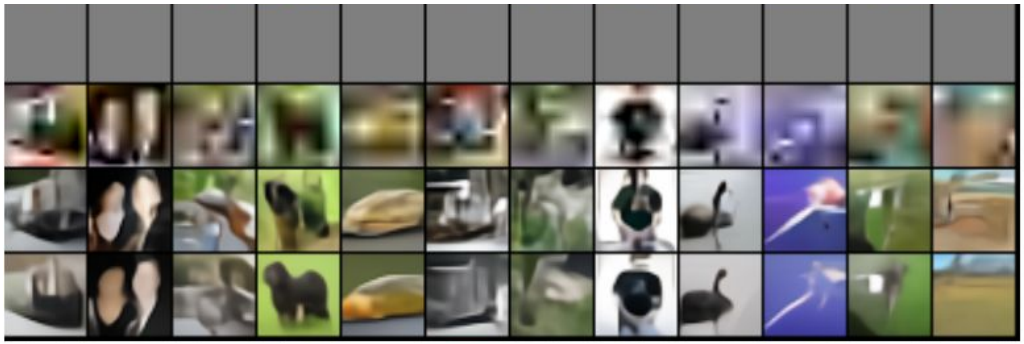
Compression rate: 0.05bits/dimension

JPEG

JPEG-2000

RVAE v1

RVAE v2



Understanding Images at multiple scales

Original images

Compression rate: 0.15bits/dimension

JPEG

JPEG-2000

RVAE v1

RVAE v2

Understanding Images at multiple scales

Original images

Compression rate: 0.2bits/dimension

JPEG

JPEG-2000

RVAE v1

RVAE v2

Summary

- Variational methods in combination with deep learning are very powerful tools for many problems in density estimation, information theory and control theory
- We have come a long way in scaling these ideas
- Several challenges remain:
  - Lower-variance gradient estimators
  - Several practical issues in training conditional models
  - Discrete variables
  - Applications to RL

A few references

- Stochastic Backpropagation and Approximate Inference in Deep Generative Models http://arxiv.org/abs/1401.4082
- Variational Inference with Normalizing Flows https://arxiv.org/abs/1505.05770
- Auto-Encoding Variational Bayes https://arxiv.org/abs/1312.6114
- Semi-Supervised Learning with Deep Generative Models http://arxiv.org/abs/1406.5298
- DRAW: A Recurrent Neural Network For Image Generation https://arxiv.org/abs/1502.04623
- Towards Conceptual Compression https://arxiv.org/abs/1604.08772
- Unsupervised Learning of 3D Structure from Images https://arxiv.org/abs/1607.00662
- One-Shot Generalization in Deep Generative Models http://arxiv.org/abs/1603.05106
- Normalizing Flows on Riemannian Manifolds https://arxiv.org/abs/1611.02304
- Improving Variational Inference with Inverse Autoregressive Flow https://arxiv.org/abs/1606.0493
- Weight Uncertainty in Neural Networks https://arxiv.org/abs/1505.05424
- Early Visual Concept Learning with Unsupervised Deep Learning https://arxiv.org/abs/1606.05579
- Bayesian Recurrent Neural Networks https://arxiv.org/abs/1704.02798
- Density estimation using Real NVP https://arxiv.org/abs/1605.08803

UAI 2017 Tutorial on

Deep Generative Models

Shakir Mohamed and Danilo Rezende

Conference on Uncertainty in Artificial Intelligence
Sydney, Australia
August 11-15, 2017

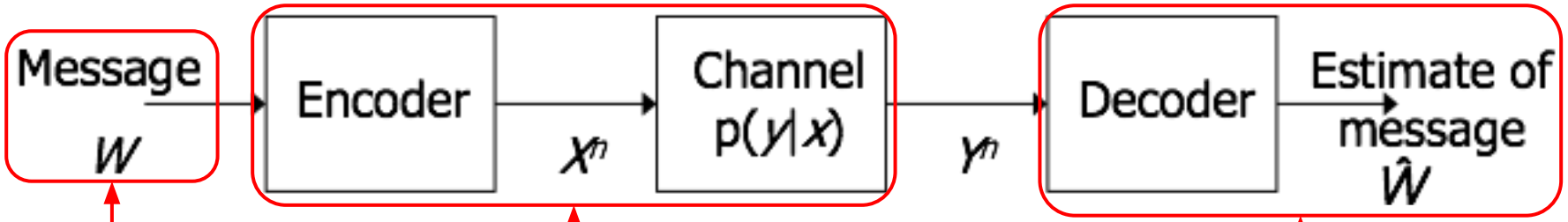
uai2017

DeepMind

Backup

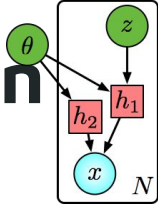
Variational Methods for Channel Capacity Estimation

$$\mathcal{E}(\mathbf{s}) \quad = \quad \max_{\omega} \mathcal{I}^{\omega}(\mathbf{a}, \mathbf{s}'|\mathbf{s}) = \max_{\omega} \mathbb{E}_{p(s'|a,s)\omega(a|s)} \left[\log \left(\frac{p(\mathbf{a}, \mathbf{s}'|\mathbf{s})}{\omega(\mathbf{a}|\mathbf{s}) p(\mathbf{s}'|\mathbf{s})} \right) \right]$$



$$\mathcal{I}^{\omega}(\mathbf{s}) = H(\mathbf{a}|\mathbf{s}) - H(\mathbf{a}|\mathbf{s}', \mathbf{s}) \geq H(\mathbf{a}) + \mathbb{E}_{p(s'|a,s)\omega_\theta(a|s)} [\log q_\xi(\mathbf{a}|\mathbf{s}', \mathbf{s})] = \mathcal{I}^{\omega,q}(\mathbf{s})$$

Approximate Inference: Perturbative expansion compatible with variational bound



$$h_n(x) = e^x - \sum_{k=0}^{2n-1} \frac{x^k}{k!}$$

h is convex

$$\Delta = -$$

Both perturbative and variational!

$$e^{-M(x,\theta)} = e^{-\mathbb{E}[} \left[\sum_{k=0}^{} \frac{}{k!} \right] z,\theta)] \mathbb{E}\left[\sum_{k=0}^{2n-1} \frac{\Delta^k}{k!} \right]$$

$$M(x,\theta) \leqslant \mathbb{E}[\mathcal{F}(x,z,\theta)] - \ln \mathbb{E}\left[\sum_{k=0}^{2n-1} \frac{\Delta^k}{k!} \right]$$