Bayesian Reasoning and Deep Learning

Shakir Mohamed

Google DeepMind

shakirm.com @shakir_za

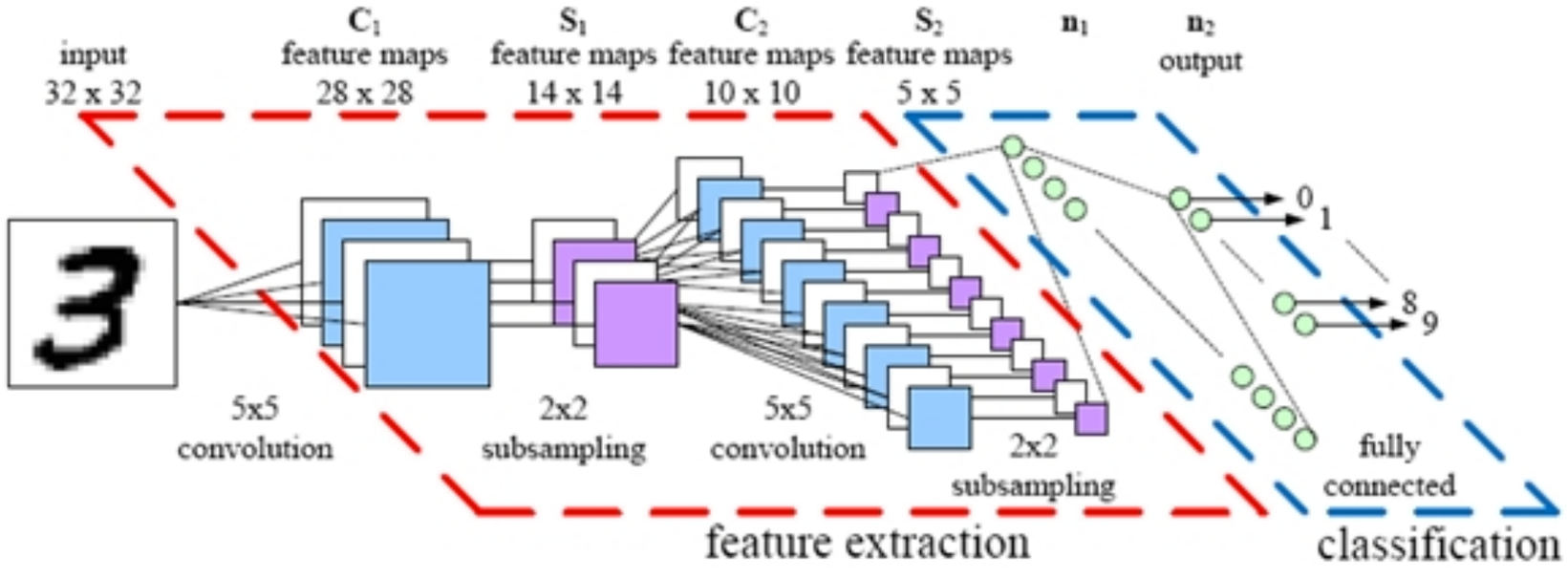
9 October 2015

Abstract

Deep learning and Bayesian machine learning are currently two of the most active areas of machine learning research. *Deep learning* provides a powerful class of models and an easy framework for learning that now provides state-of-the-art methods for applications ranging from image classification to speech recognition. *Bayesian reasoning* provides a powerful approach for information integration, inference and decision making that has established it as the key tool for data-efficient learning, uncertainty quantification and robust model composition that is widely used in applications ranging from information retrieval to large-scale ranking. Each of these research areas has shortcomings that can be effectively addressed by the other, pointing towards a needed convergence of these two areas of machine learning; the complementary aspects of these two research areas is the focus of this talk. Using the tools of auto-encoders and latent variable models, we shall discuss some of the ways in which our machine learning practice is enhanced by combining deep learning with Bayesian reasoning. This is an essential, and ongoing, convergence that will only continue to accelerate and provides some of the most exciting prospects, some of which we shall discuss, for contemporary machine learning research.


Deep Learning
Bayesian Reasoning
Better ML

Deep Learning



A framework for constructing flexible models

+ Rich non-linear models for classification and sequence prediction.

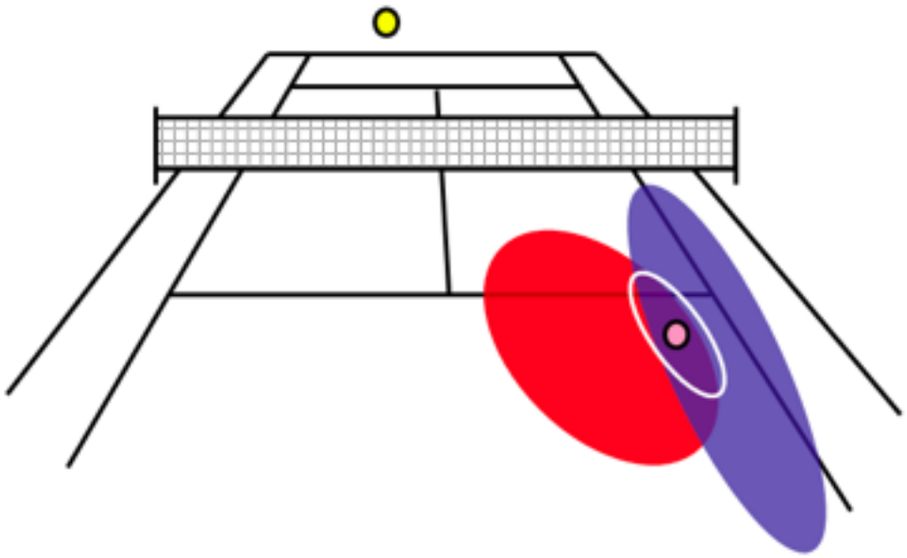
+ Scalable learning using stochastic approximations and conceptually simple.

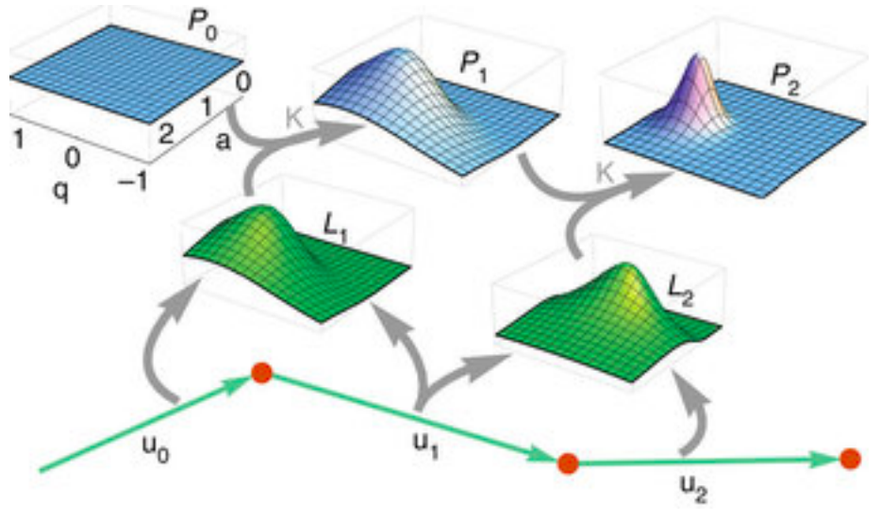
+ Easily composable with other gradient-based methods

- Only point estimates

- Hard to score models, do model selection and complexity penalisation.

Bayesian Reasoning





A framework for inference and decision making

+ Unified framework for model building, inference, prediction and decision making

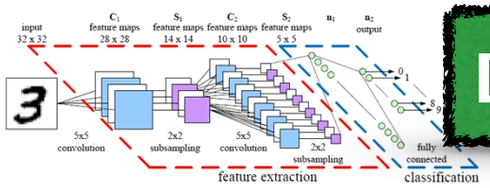
+ Explicit accounting for uncertainty and variability of outcomes

+ Robust to overfitting; tools for model selection and composition.

- Mainly conjugate and linear models

- Potentially intractable inference leading to expensive computation or long simulation times.

Two Streams of Machine Learning



Deep Learning

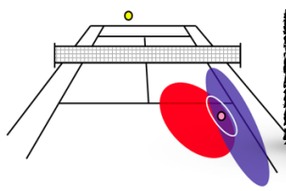
+ Rich non-linear models for classification and sequence prediction.

+ 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.



Bayesian Reasoning

- Mainly conjugate and linear models

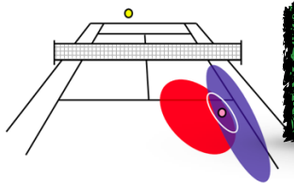
- Potentially intractable inference, computationally expensive or long simulation time.

+ Unified framework for model building, inference, prediction and decision making

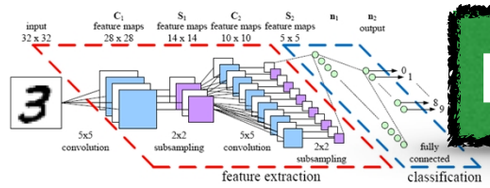
+ Explicit accounting for uncertainty and variability of outcomes

+ Robust to overfitting; tools for model selection and composition.

Outline



Bayesian Reasoning + **Deep Learning**



Complementary strengths that we should expect to be successfully combined.

1. **Why is this a good idea?**
   - ✤ Review of deep learning
   - ✤ Limitations of maximum likelihood and MAP estimation

2. **How can we achieve this convergence?**
   - ✤ Case study using auto-encoders and latent variable models
   - ✤ Approximate Bayesian inference

3. **What else can we do?**
   - ✤ Semi-supervised learning, classification, better inference and more.

A (Statistical) Review of Deep Learning

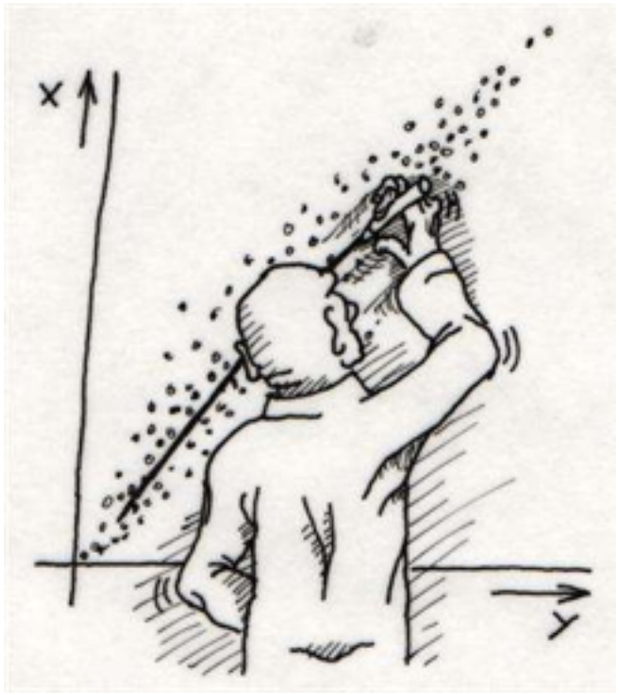
Generalised Linear Regression

$$\eta = \mathbf{w}^\top \mathbf{x} + b$$

$$p(y|\mathbf{x}) = p(y|g(\eta);\theta)$$

- The basic function can be any linear function, e.g., affine, convolution.
- $g(.)$ is an ***inverse link function*** that we'll refer to as an activation function.

Target	Regression	Link	Inv link	Activation
Real	Linear	Identity	Identity	
Binary	Logistic	Logit $\log \frac{\mu}{1-\mu}$	Sigmoid $\frac{1}{1+\exp(-\eta)}$	Sigmoid
Binary	Probit	Inv Gauss CDF $\Phi^{-1}(\mu)$	Gauss CDF $\Phi(\eta)$	Probit
Binary	Gumbel	Compl. log-log $\log(-\log(\mu))$	Gumbel CDF $e^{-e^{-x}}$	
Binary	Logistic		Hyperbolic Tangent $\tanh(\eta)$	Tanh
Categorical	Multinomial		Multin. Logit $\frac{\eta_i}{\sum_j \eta_j}$	Softmax
Counts	Poisson	$\log(\mu)$	$\exp(\nu)$	
Counts	Poisson	$\sqrt{(}\mu)$	ν^2	
Non-neg.	Gamma	Reciprocal $\frac{1}{\mu}$	$\frac{1}{\nu}$	
Sparse	Tobit		max $\max(0;\nu)$	ReLU
Ordered	Ordinal		Cum. Logit $\sigma(\phi_k - \eta)$	

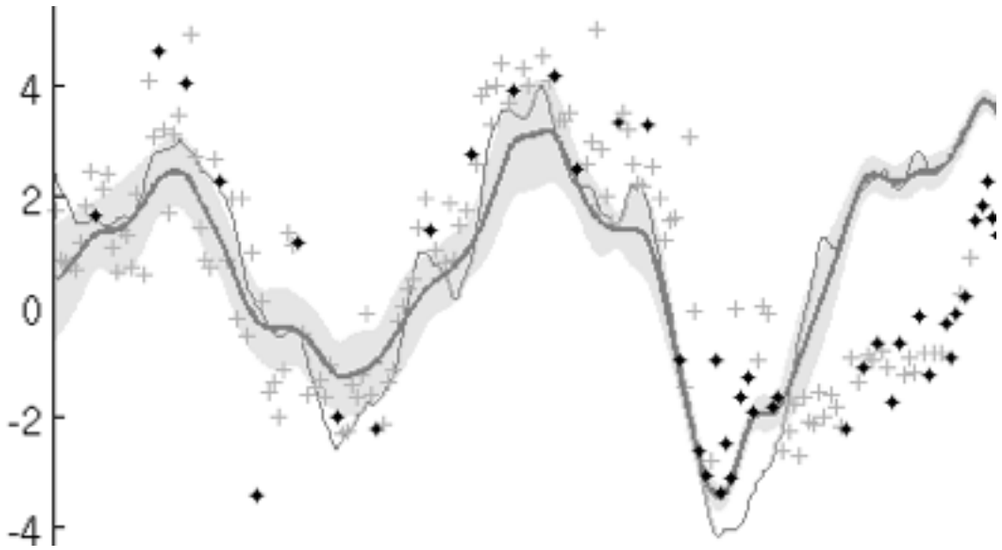


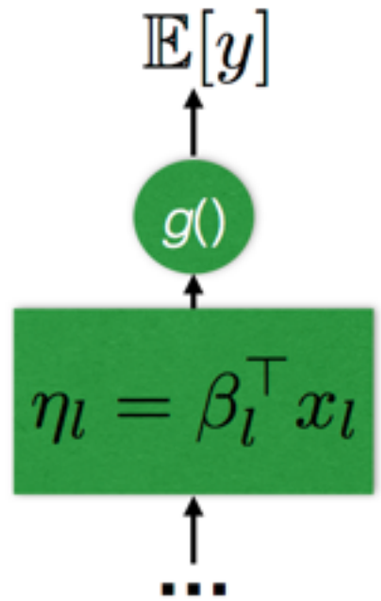
Maximum likelihood estimation

Optimise the negative log-likelihood

$$\mathcal{L} = -\log p(y|g(\eta);\theta)$$

A (Statistical) Review of Deep Learning

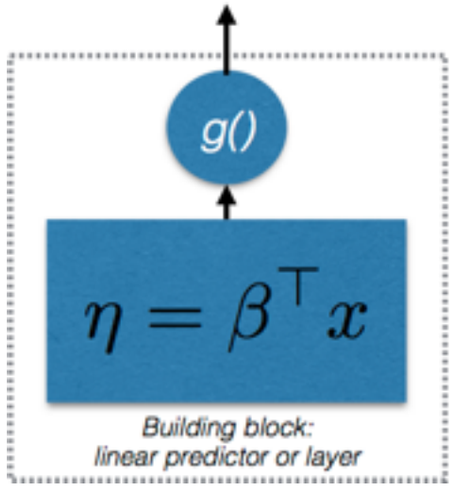




Recursive Generalised Linear Regression

- ✦ Recursively compose the basic linear functions.
- ✦ Gives a deep neural network.

$$\mathbb{E}[y] = h_L \circ \ldots \circ h_l \circ h_0(\mathbf{x})$$



A general framework for building non-linear, parametric models

Problem: Overfitting of MLE leading to limited generalisation.

A (Statistical) Review of Deep Learning

Regularisation Strategies for Deep Networks

- ✦ Regularisation is essential to overcome the limitations of maximum likelihood estimation.
- ✦ Regularisation, penalised regression, shrinkage.
- ✦ A wide range of available regularisation techniques:

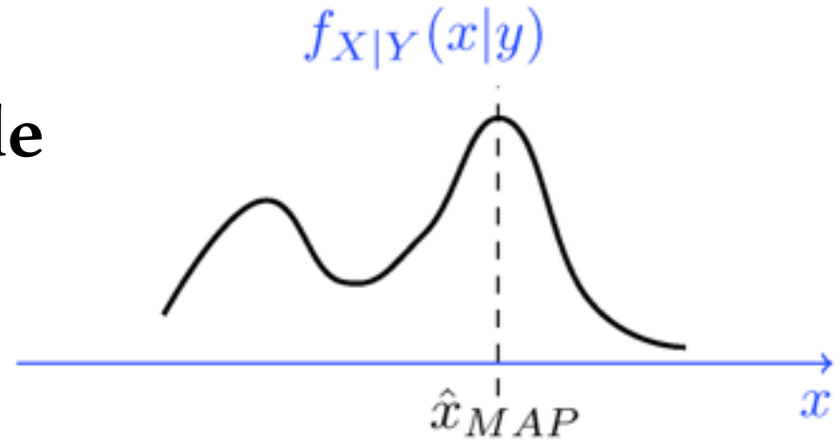
  - Large data sets
  - Input noise/jittering and data augmentation/expansion.
  - L2 /L1 regularisation (Weight decay, Gaussian prior)
  - Binary or Gaussian Dropout
  - Batch normalisation

More robust loss function using MAP estimation instead.

More Robust Learning

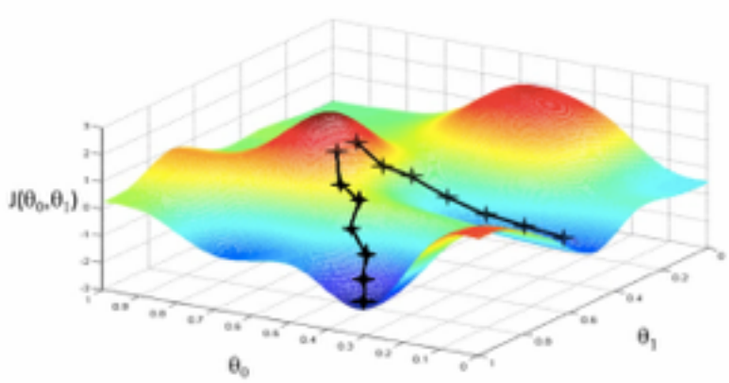
MAP estimators and limitations

- ✦ Power of MAP estimators is that they provide some robustness to overfitting.
- ✦ Creates sensitivities to parameterisation.

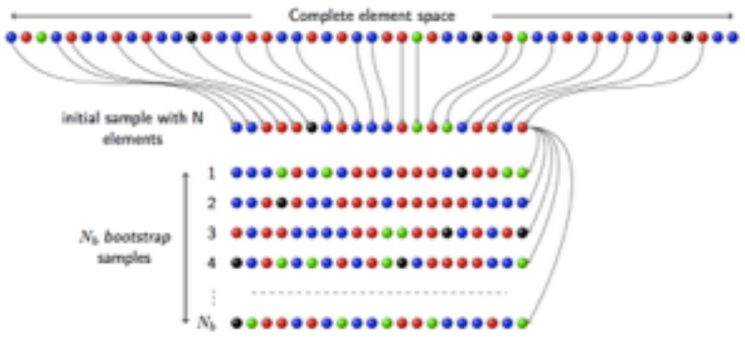


1. Sensitivities affect gradients and can make learning hard

Invariant MAP estimators and exploiting natural gradients, trust region methods and other improved optimisation.



2. Still no way to measure confidence of our model.



Can generate frequentist confidence intervals and bootstrap estimates.

Towards Bayesian Reasoning

Proposed solutions have not fully dealt with the underlying issues.

Issues arise as a consequence of:

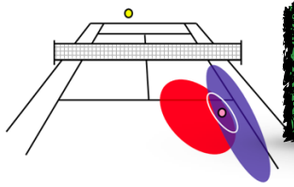
- **Reasoning only about the most likely solution and**
- **Not maintaining knowledge of the underlying variability (and averaging over this).**

Given this powerful model class and invaluable tools for regularisation and optimisation, let us develop a

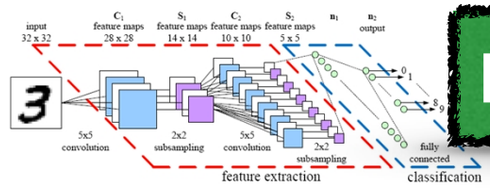
Pragmatic Bayesian Approach for
Probabilistic Reasoning in Deep Networks.

Bayesian reasoning over some, but not all parts of our models (yet).

Outline



Bayesian Reasoning + **Deep Learning**



Complementary strengths that we should expect to be successfully combined.

1. Why is this a good idea?
   - ✤ Review of deep learning
   - ✤ Limitations of maximum likelihood and MAP estimation

2. **How can we achieve this convergence?**
   - ✤ Case study using auto-encoders and latent variable models
   - ✤ Approximate Bayesian inference

3. What else can we do?
   - ✤ Semi-supervised learning, classification, better inference and more.

Dimensionality Reduction and Auto-encoders

Unsupervised learning and auto-encoders

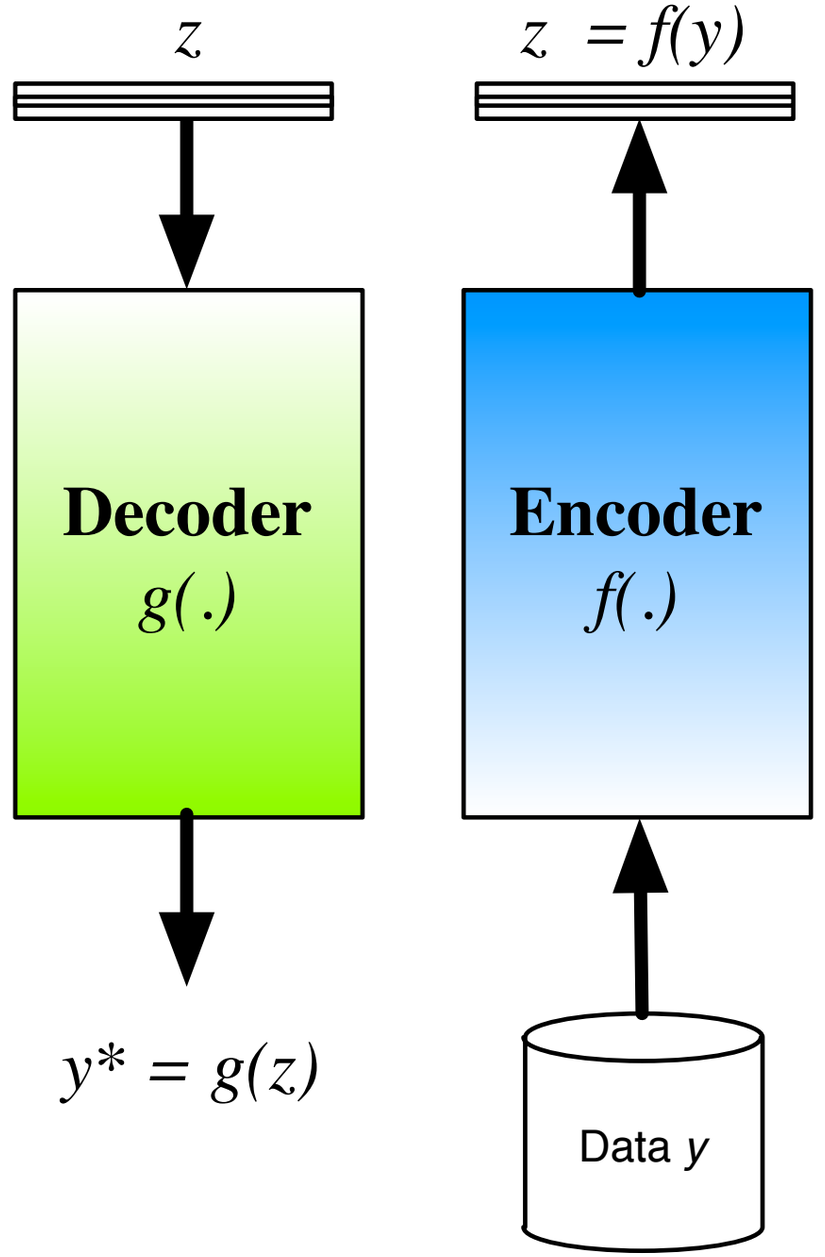
- A generic tool for dimensionality reduction and feature extraction.
- Minimise reconstruction error using an encoder and a decoder.

+ Non-linear dimensionality reduction using deep networks for encoder and decoder.

+ Easy to implement as a single computational graph and train using SGD

- No natural handling of missing data

- No representation of variability of the representation space.



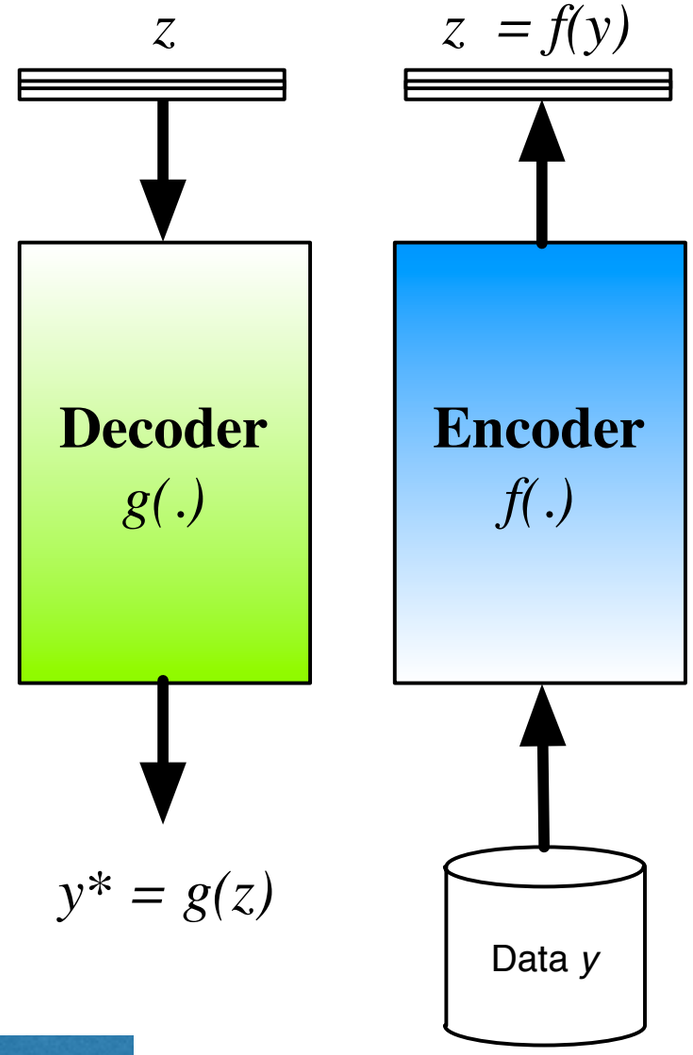
$$\mathcal{L} = -\log p(y|g(z))$$

$$\mathcal{L} = \|y - g(f(y))\|_2^2$$

Dimensionality Reduction and Auto-encoders

Some questions about auto-encoders:

- What is the model we are interested in?
- Why use an encoder?
- How do we regularise?



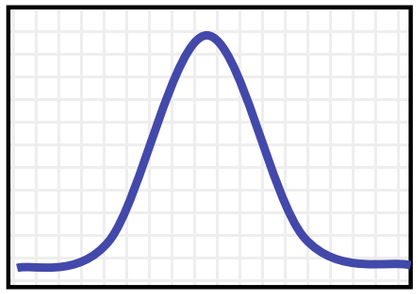
Best to be explicit about the:

- Probabilistic **model** of interest and
- Mechanism we use for **inference**.

Density Estimation and Latent Variable Models

Latent variable models:

- Generic and flexible model class for density estimation.
- Specifies a generative process that gives rise to the data.



Latent Gaussian Models:

- Probabilistic PCA, Factor analysis (FA), Bayesian Exponential Family PCA (BXPCA).

BXPCA

Latent Variable

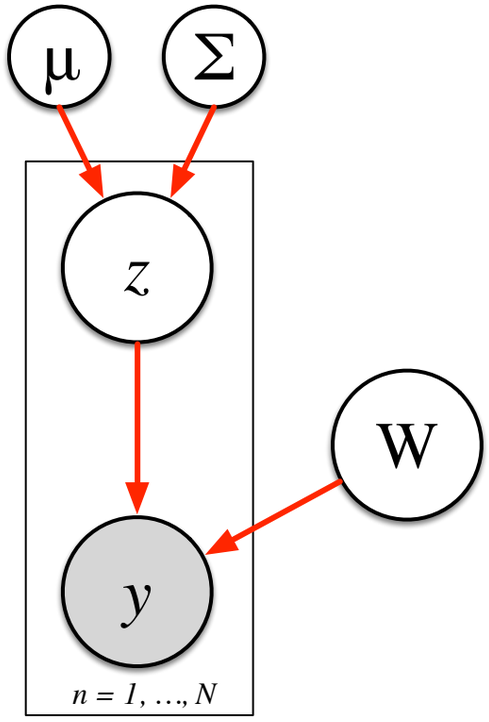
$$\mathbf{z} \sim \mathcal{N}(\mathbf{z}|\mu, \boldsymbol{\Sigma})$$

Observation Model

$$\boldsymbol{\eta} = \mathbf{W}\mathbf{z} + \mathbf{b}$$

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

Exponential fam natural parameters η.



Use our knowledge of deep learning to design even richer models.

Deep Generative Models

Rich extension of previous model using deep neural networks.

E.g., non-linear factor analysis, non-linear Gaussian belief networks, deep latent Gaussian models (DLGM).

DLGM

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

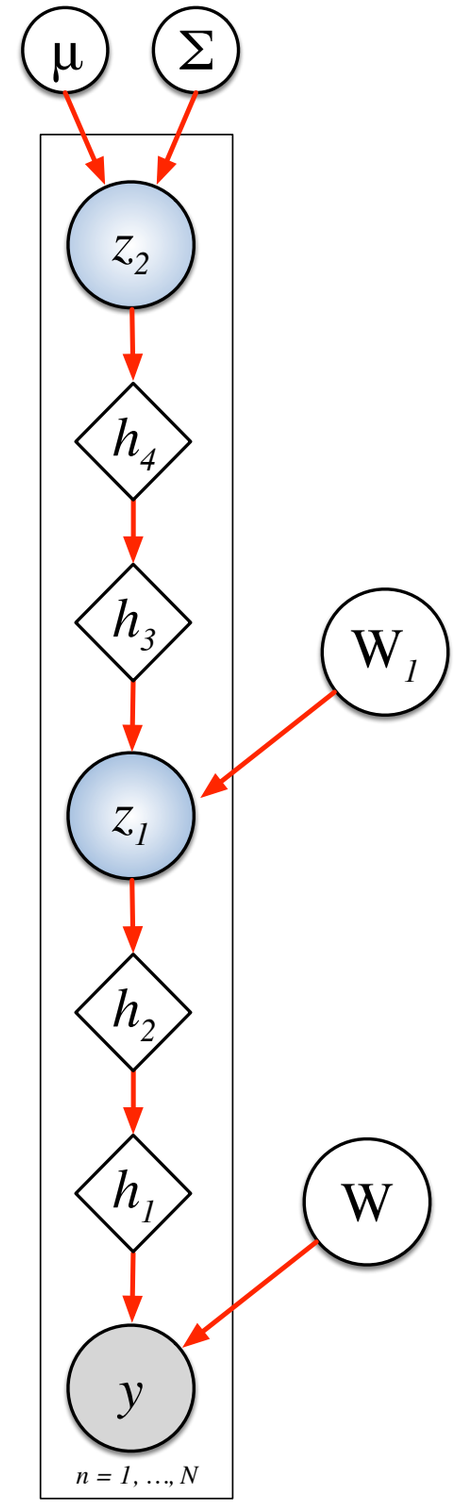
$$h_i(\mathbf{x}) = \sigma(\mathbf{A}\mathbf{x} + \mathbf{c})$$

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.



Deep Latent Gaussian Models

Our inferential tasks are:

1. Explain this data

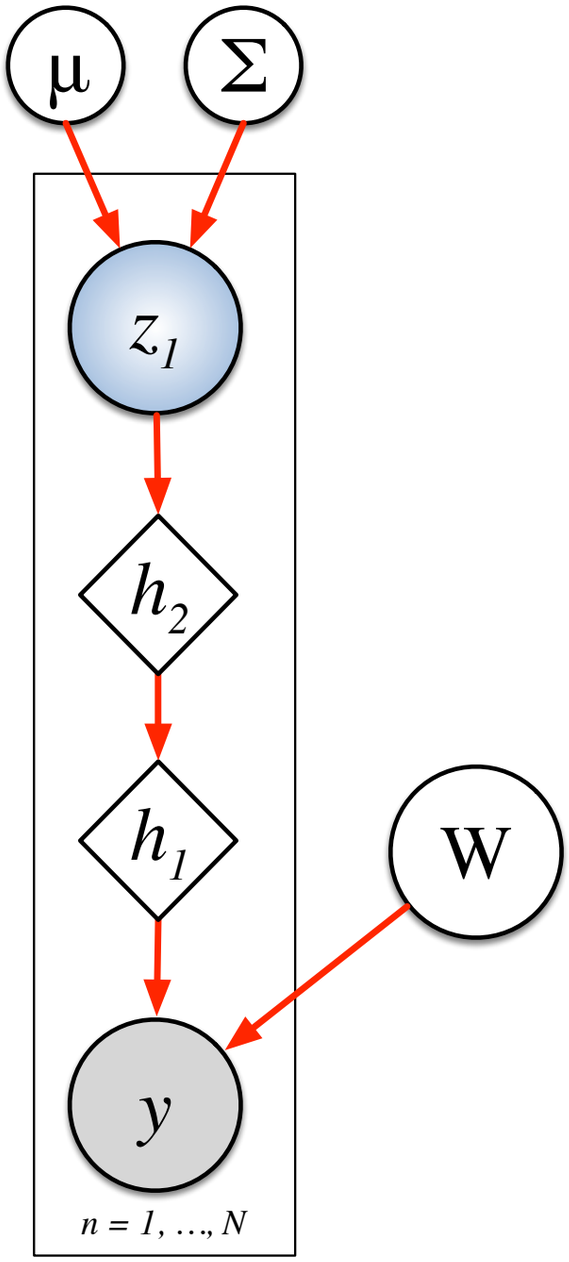
$$p(\mathbf{z}|\mathbf{y}, \mathbf{W}) \propto p(\mathbf{y}|\mathbf{z}, \mathbf{W})p(\mathbf{z})$$

2. Make predictions:

$$p(\mathbf{y}^*|\mathbf{y}) = \int p(\mathbf{y}^*|\mathbf{z}, \mathbf{W})p(\mathbf{z}|\mathbf{y}, \mathbf{W})d\mathbf{z}$$

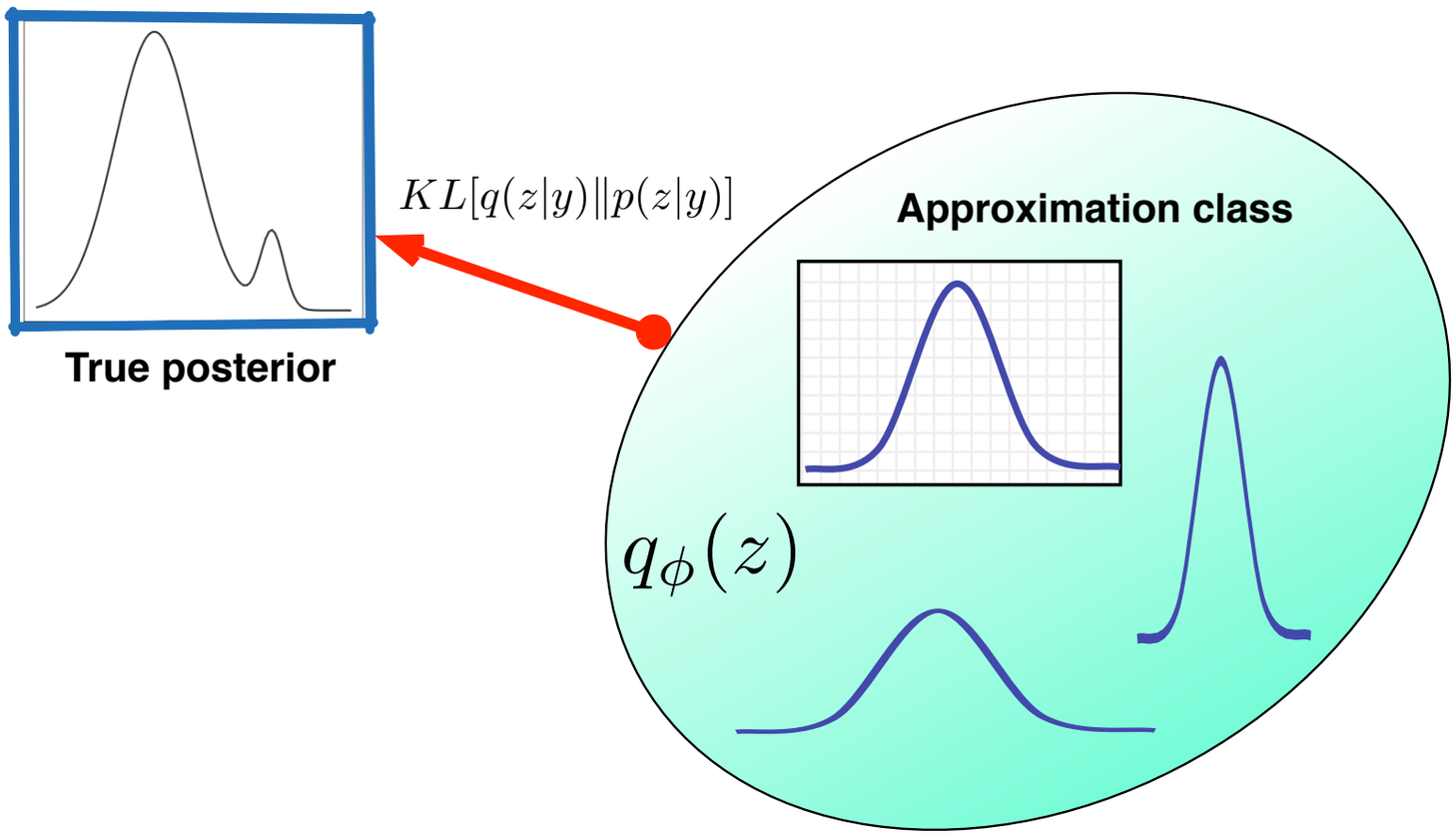
3. Choose the best model

$$p(\mathbf{y}|\mathbf{W}) = \int p(\mathbf{y}|\mathbf{z}, \mathbf{W})p(\mathbf{z})d\mathbf{z}$$



Variational Inference

Use tools from approximate inference to handle intractable integrals.



- **Reconstruction cost**: Expected log-likelihood measures how well samples from $q(z)$ are able to explain the data y.
- **Penalty:** Explanation of the data $q(z)$ doesn't deviate too far from your beliefs $p(z)$ - Okham's razor.

Reconstruction

Penalty

$$\mathcal{F}(y,q) = \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)\|p(z)]$$

Penalty is derived from your model and does not need to be designed.

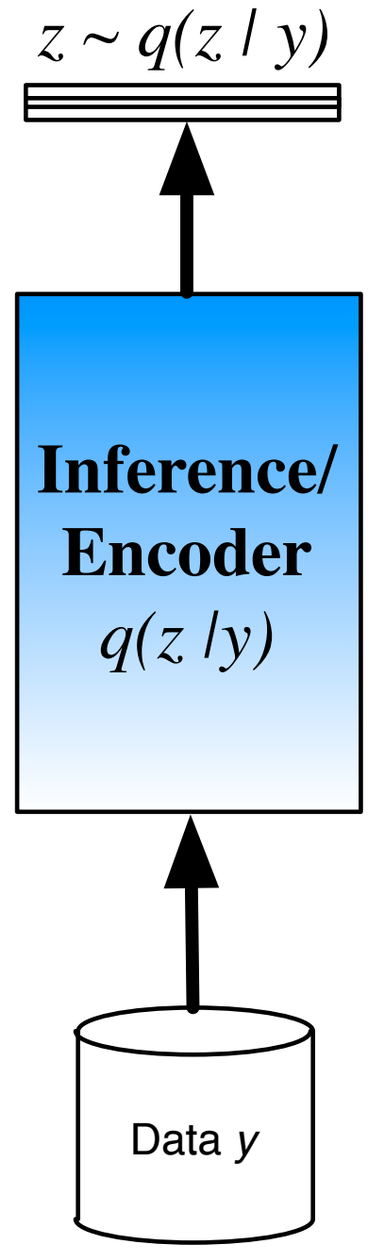
Amortised Variational Inference

$$\mathcal{F}(y,q) = \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)\|p(z)]$$

Approx. Posterior

Reconstruction

Penalty



Approximate posterior distribution *q(z):* Best match to true posterior *p(z|y)*, one of the unknown inferential quantities of interest to us.

Inference network: q is an encoder or inverse model.

Parameters of q are now a set of global parameters used for inference of all data points - test and train.

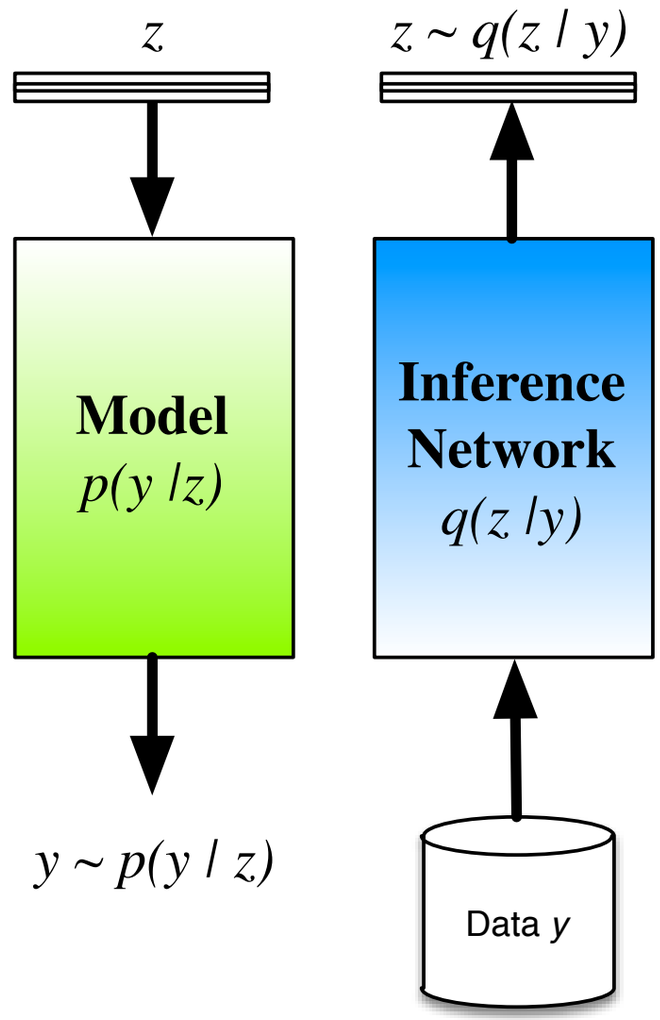
Amortise (spread) the cost of inference over all data.

Encoders provide an efficient mechanism for **amortised posterior inference**

Auto-encoders and Inference in DGMs

$$\mathcal{F}(y,q) = \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)\|p(z)]$$

Approx. Posterior | Reconstruction | Penalty



- **Model (Decoder):** likelihood $p(y|z)$.
- **Inference (Encoder):** variational distribution $q(z|y)$

Stochastic encoder-decoder systems implement variational inference.

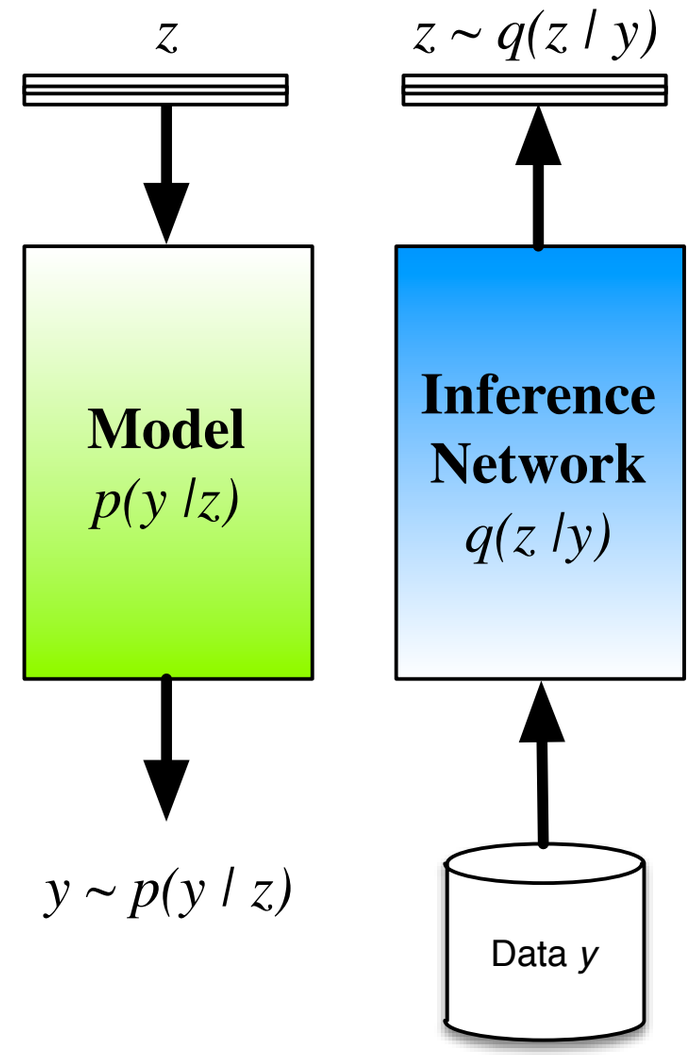
Specific combination of **variational inference** in **latent variable models** using **inference networks**
Variational Auto-encoder

But don't forget what your model is, and what inference you use.

What Have we Gained

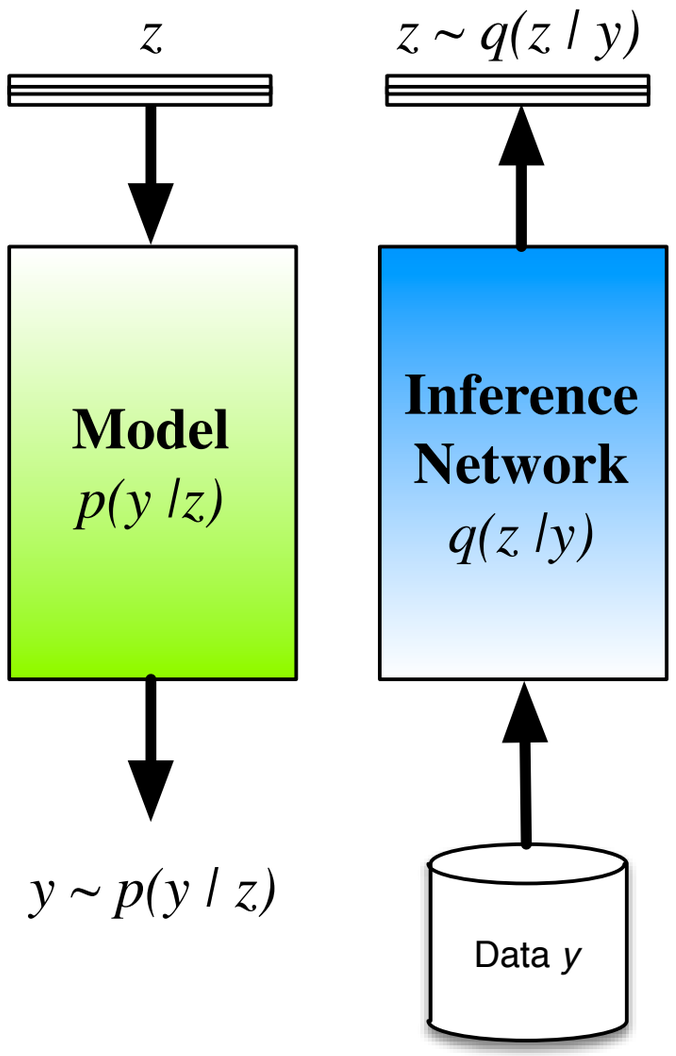
- Transformed an auto-encoders into more interesting **deep generative models.**
- Rich new class of density estimators built with **non-linear models.**
- Used a **principled approach** for deriving loss functions that automatically include appropriate penalty functions.
- Explained **how** an encoder enters into our models and **why** this is a good idea.
- Able to answer all our desired **inferential questions.**
- **Knowledge of the uncertainty** associated with our latent variables.

$$\mathcal{F}(y,q) = \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)||p(z)]$$



What Have we Gained

$$\mathcal{F}(y, q) = \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)\|p(z)]$$

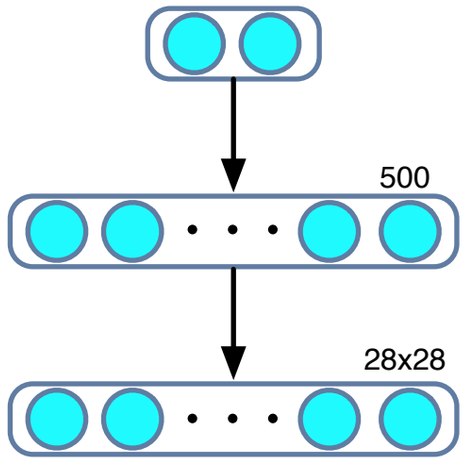


- Able to **score our models** and do model selection using the free energy.
- Can **impute missing data** under any missingness assumption
- Can still combine with natural gradient and improved optimisation tools.
- Easy implementation - have a single computational graph and simple Monte Carlo gradient estimators.
- Computational complexity the same as any large-scale deep learning system.

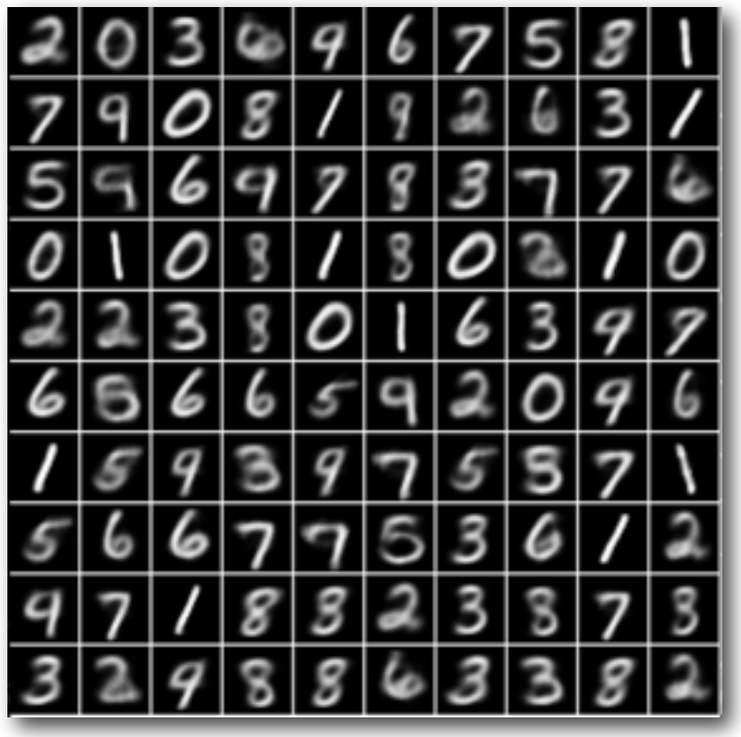
A true marriage of Bayesian Reasoning and Deep Learning

Data Visualisation

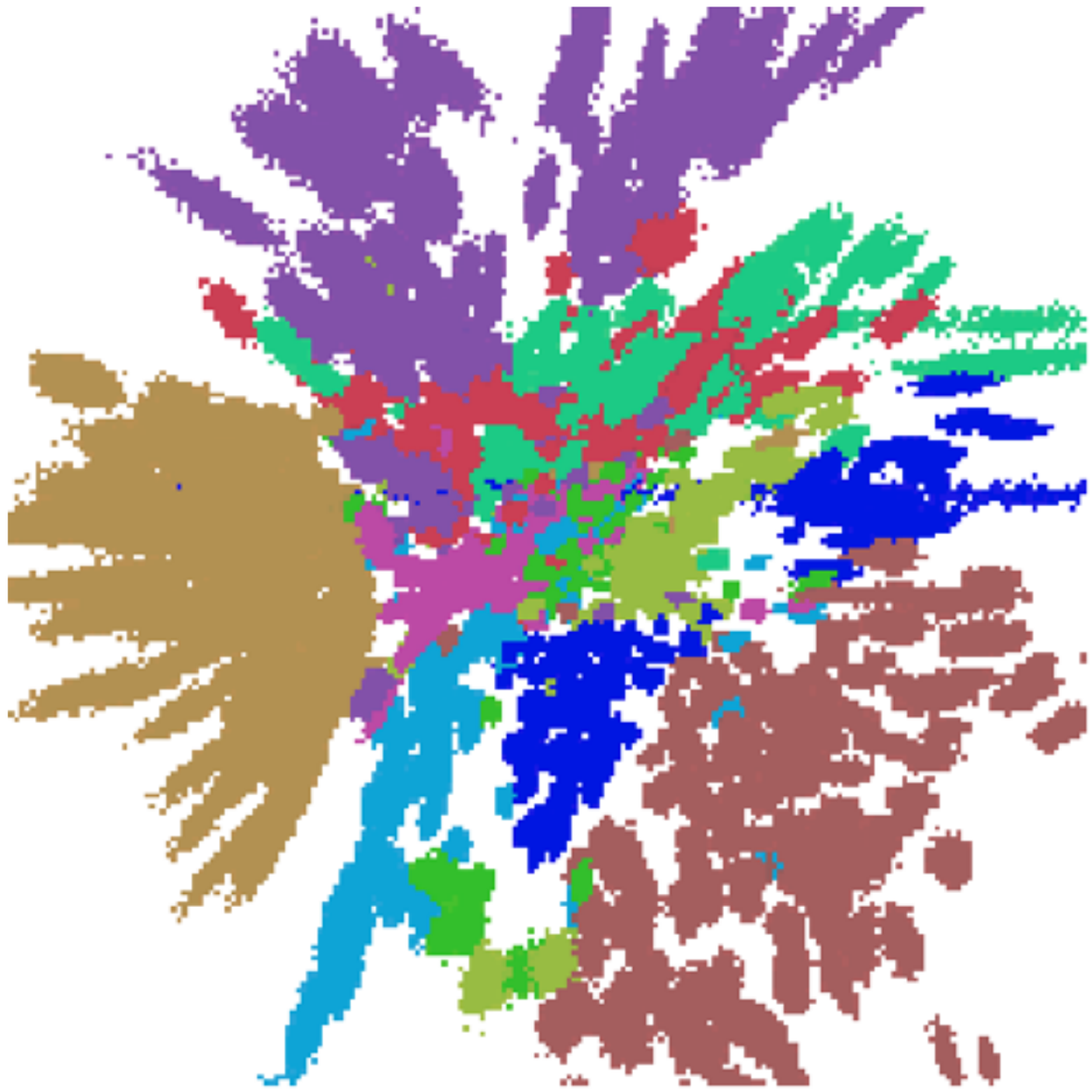
MNIST Handwritten digits



DLGM



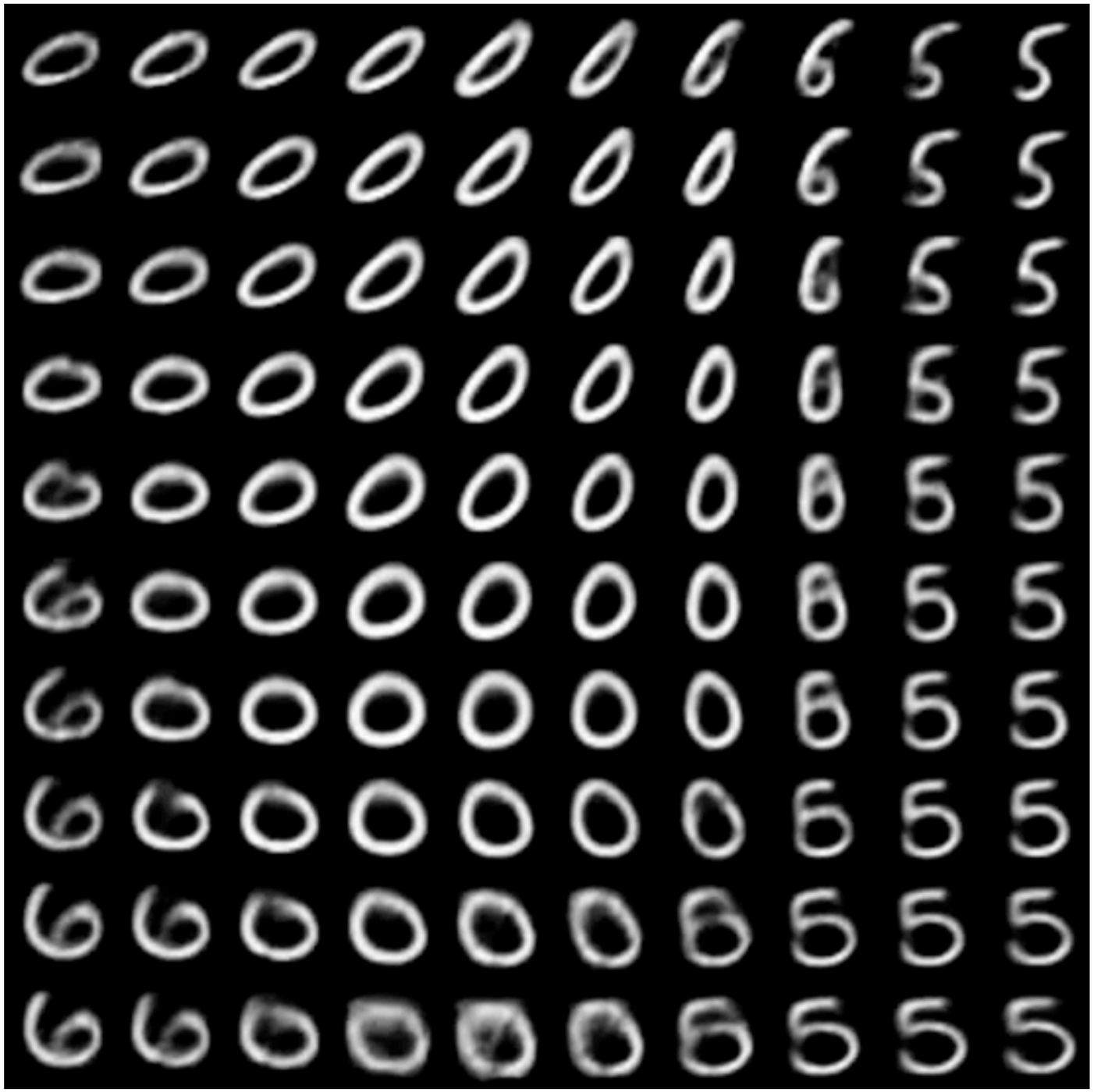
Samples from 2D latent model



Labels in 2D latent space

Visualising MNIST in 3D

DLGM



Data Simulation

DLGM

Data

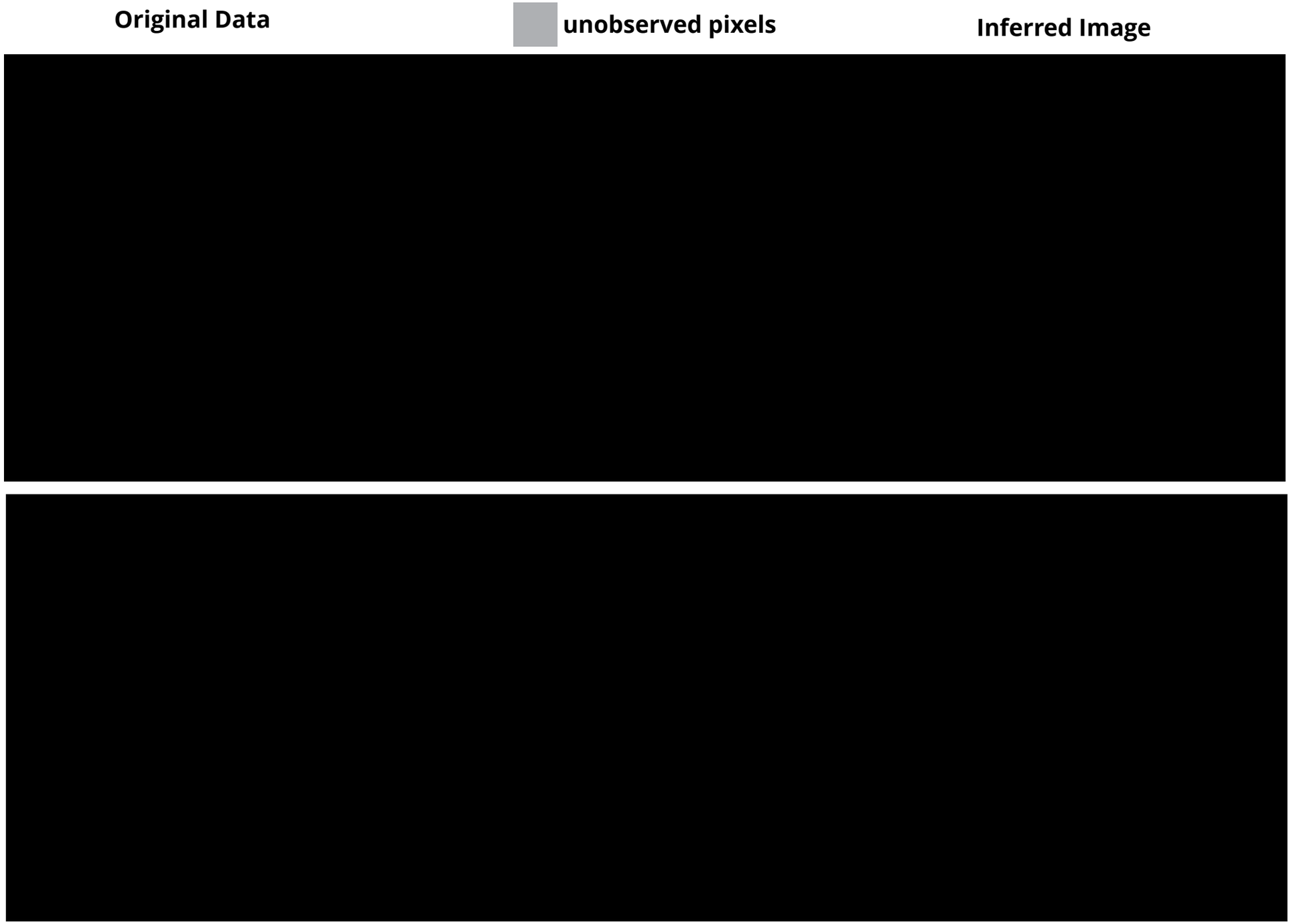
Samples

Missing Data Imputation

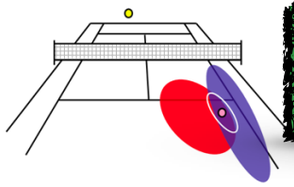
DLGM

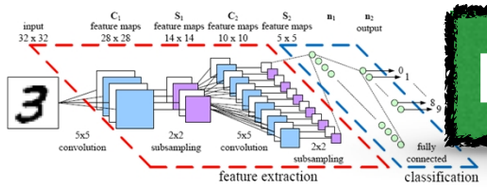
10% observed

50% observed



Outline



Bayesian Reasoning + 

Deep Learning

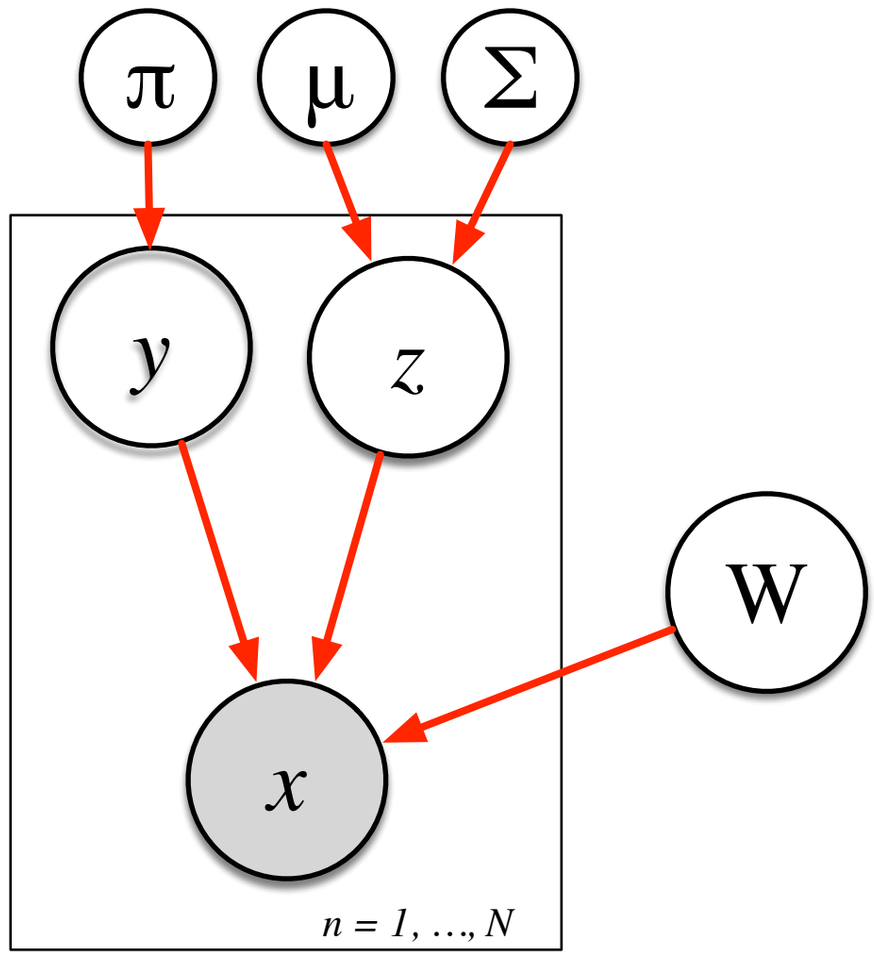
Complementary strengths that we should expect to be successfully combined.

1. Why is this a good idea?
   - ✤ Review of deep learning
   - ✤ Limitations of maximum likelihood and MAP estimation
2. How can we achieve this convergence?
   - ✤ Auto-encoders and latent variable models
   - ✤ Approximate and variational inference
3. **What else can we do?**
   - ✤ Semi-supervised learning, recurrent networks, classification, better inference and more.

Semi-supervised Learning

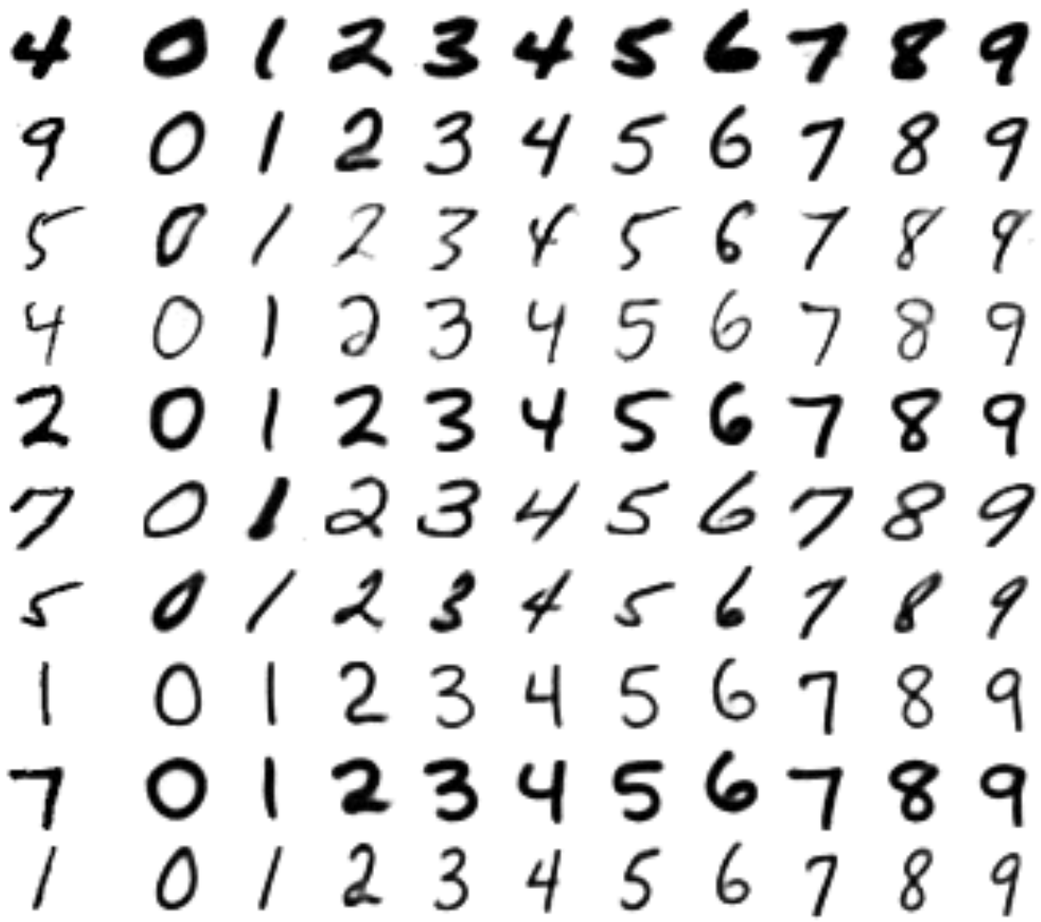
Can extend the marriage of Bayesian reasoning and deep learning to the problem of semi-supervised classification.

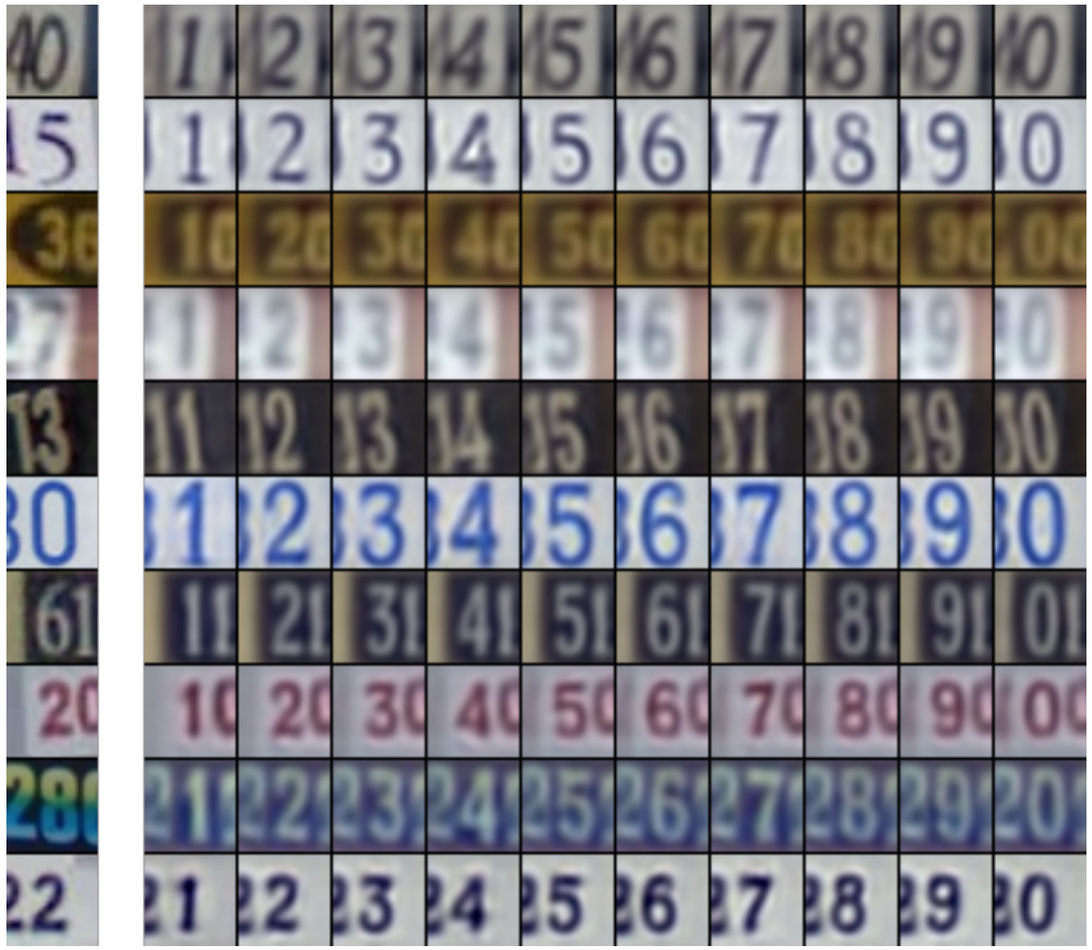
Semi-supervised DLGM



Analogical Reasoning

Semi-supervised DLGM

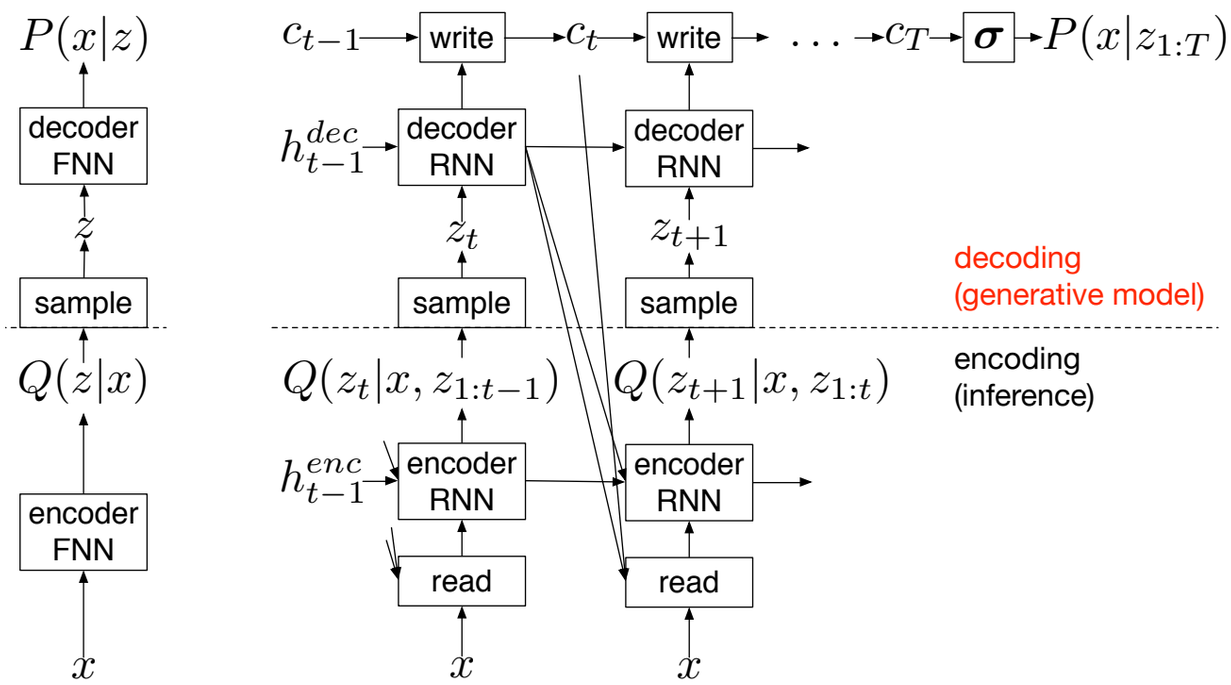




Generative Models with Attention

We can also combine other tools from deep learning to design even more powerful generative models: recurrent networks and attention.

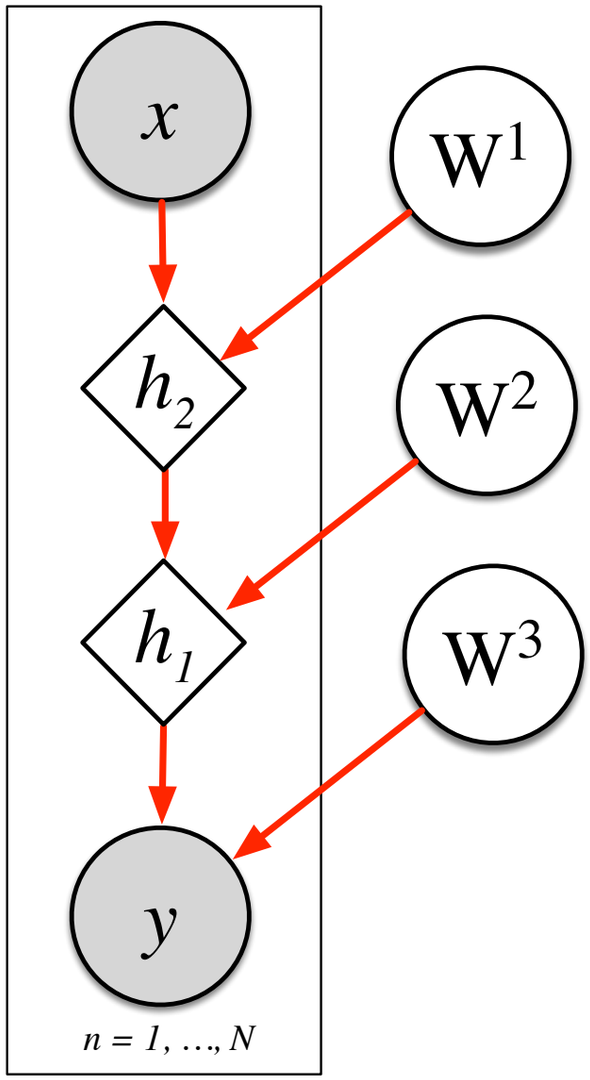
DRAW

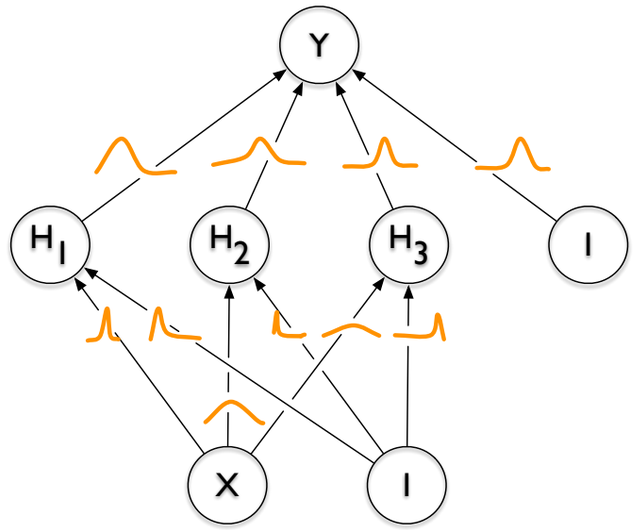


Uncertainty on Model Parameters

We can also combine other tools from deep learning to design even more powerful generative models: recurrent networks and attention.

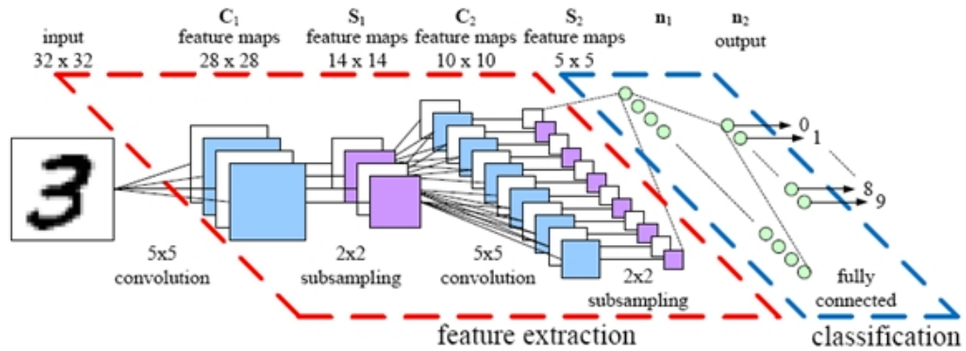
Bayesian Neural Networks

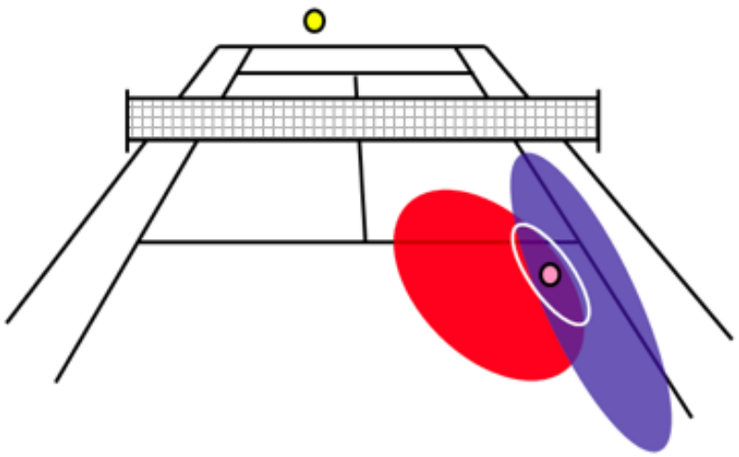




In Review …

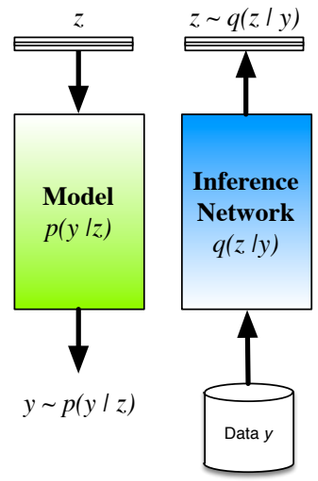
Deep learning as a ***framework for building highly flexible non-linear parametric models***, but regularisation and accounting for uncertainty and lack of knowledge is still needed.





Bayesian reasoning as a general ***framework for inference that allows us to account for uncertainty*** and a principled approach for regularisation and model scoring.

Combined Bayesian reasoning with auto-encoders and showed just how much can be gained by a ***marriage of these two streams*** of machine learning research.



Thanks to many people:

Danilo Rezende, Ivo Danihelka, Karol Gregor, Charles Blundell, Theophane Weber, Andriy Mnih, Daan Wierstra (*Google DeepMind*), Durk Kingma, Max Welling (*U. Amsterdam*)

Thank You.

Some References

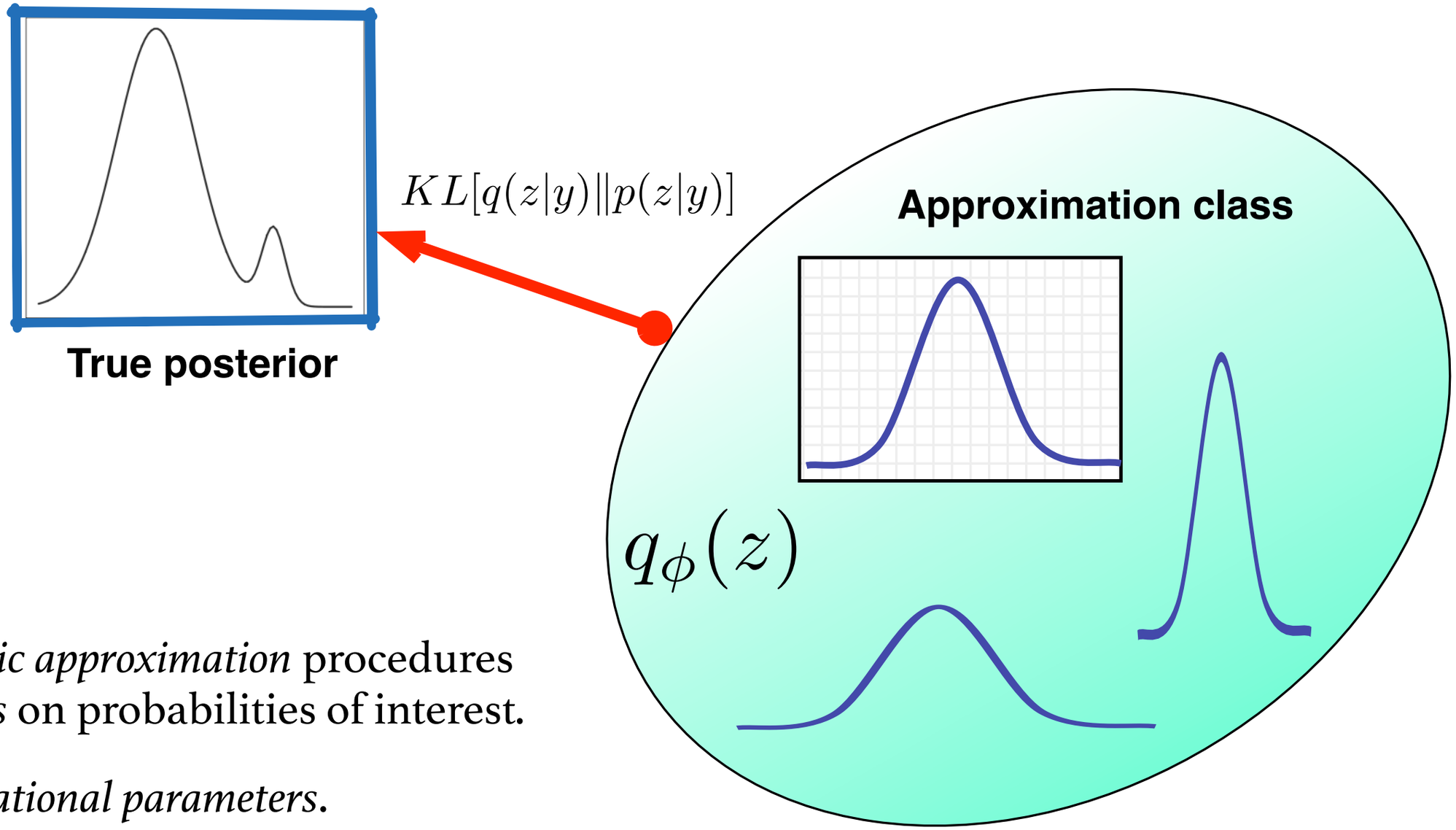
Probabilistic Deep Learning

- Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra. "***Stochastic backpropagation and approximate inference in deep generative models***." ICML (2014).
- Kingma, Diederik P., and Max Welling. "***Auto-encoding variational Bayes***." ICLR 2014.
- Mnih, Andriy, and Karol Gregor. "***Neural variational inference and learning in belief networks***." ICML (2014).
- Gregor, Karol, et al. "***Deep autoregressive networks***." ICML (2014).
- Kingma, D. P., Mohamed, S., Rezende, D. J., & Welling, M. (2014). ***Semi-supervised learning with deep generative models***. NIPS (pp. 3581-3589).
- Gregor, K., Danihelka, I., Graves, A., & Wierstra, D. (2015). **DRAW: A recurrent neural network for image generation.** arXiv preprint arXiv: 1502.04623.
- Rezende, D. J., & Mohamed, S. (2015). **Variational Inference with Normalizing Flows**. arXiv preprint arXiv:1505.05770.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. (2015). **Weight Uncertainty in Neural Networks**. arXiv preprint arXiv:1505.05424.
- Hernández-Lobato, J. M., & Adams, R. P. (2015). **Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks**. arXiv preprint arXiv:1502.05336.
- Gal, Y., & Ghahramani, Z. (2015). **Dropout as a Bayesian approximation: Representing model uncertainty in deep learning.** arXiv preprint arXiv:1506.02142.

What is a Variational Method?

Variational Principle

General family of methods for approximating complicated densities by a simpler class of densities.



Deterministic *approximation* procedures with *bounds* on probabilities of interest.

Fit the *variational parameters*.

From IS to Variational Inference

Integral problem

$$\log p(y) = \log \int p(y|z)p(z)dz$$

Proposal

$$\log p(y) = \log \int p(y|z)p(z)\frac{q(z)}{q(z)}dz$$

Importance Weight

$$\log p(y) = \log \int p(y|z)\frac{p(z)}{q(z)}q(z)dz$$

Jensen's inequality

$$\log \int p(x)g(x)dx \geq \int p(x) \log g(x)dx$$

$$\log p(y) \geq \int q(z) \log \left(p(y|z)\frac{p(z)}{q(z)} \right) dz$$

$$= \int q(z) \log p(y|z) - \int q(z) \log \frac{q(z)}{p(z)}$$

Variational lower bound

$$= \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)\|p(z)]$$

Minimum Description Length (MDL)

$$\mathcal{F}(y,q) = \mathbb{E}_{q(z)}[\log p(y|z)] - KL[q(z)\|p(z)]$$

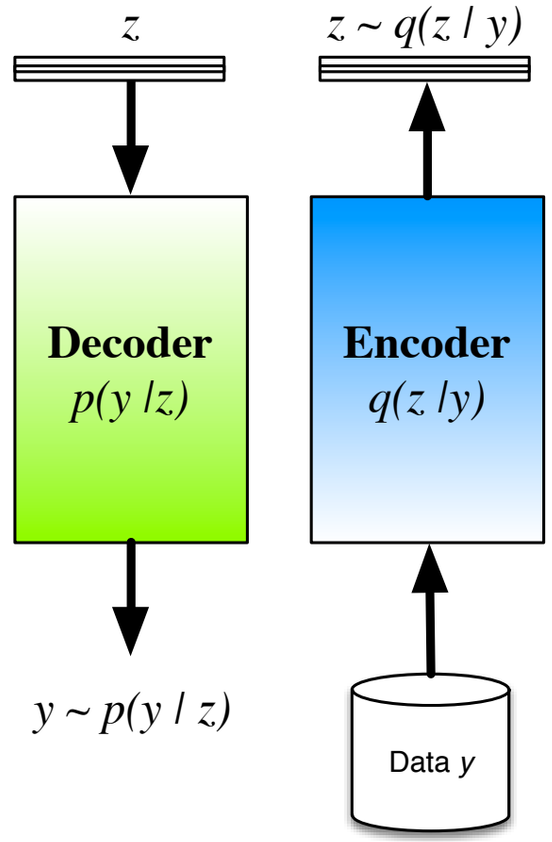
Stochastic encoder

Data code-length

Hypothesis code

Stochastic encoder-decoder systems implement variational inference.

- Regularity in our data that can be explained with latent variables, implies that the data is compressible.
- MDL: inference seen as a problem of compression — we must find the ideal shortest message of our data *y:* marginal likelihood.
- Must introduce an approximation to the ideal message.
- **Encoder:** variational distribution $q(z|y)$,
- **Decoder:** likelihood $p(y|z)$.



Denoising Auto-encoders (DAE)

$$\mathcal{F}(y,q) = \mathbb{E}_{q(z)}[\log p(y|z)] - \Omega(z,y)$$

Stochastic encoder

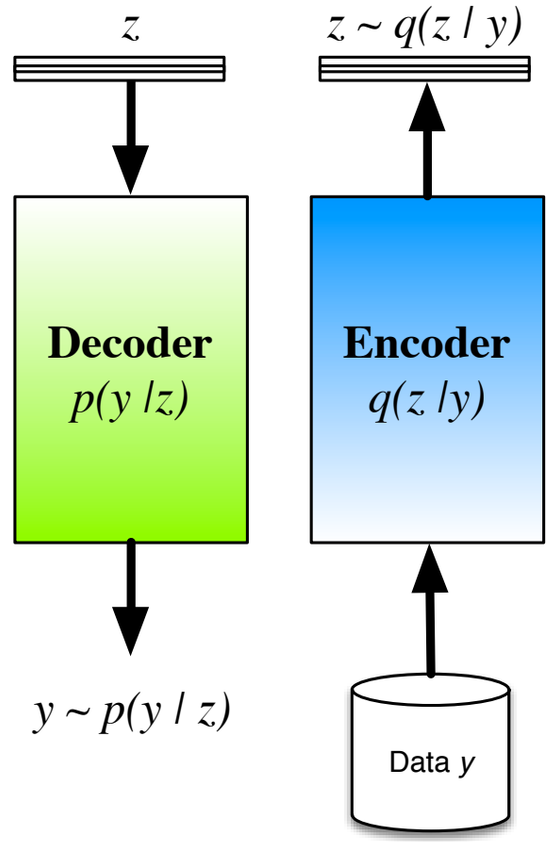
Reconstruction

Penalty

Stochastic encoder-decoder systems implement variational inference.

- DAE: A mechanism for finding representations or features of data (i.e. latent variable explanations).
- **Encoder:** variational distribution *q(z|y)*,
- **Decoder:** likelihood *p(y|z)*.

The variational approach requires you to be explicit about your assumptions. Penalty is derived from your model and does not need to be designed.



Amortising the Cost of Inference

Repeat:

E-step

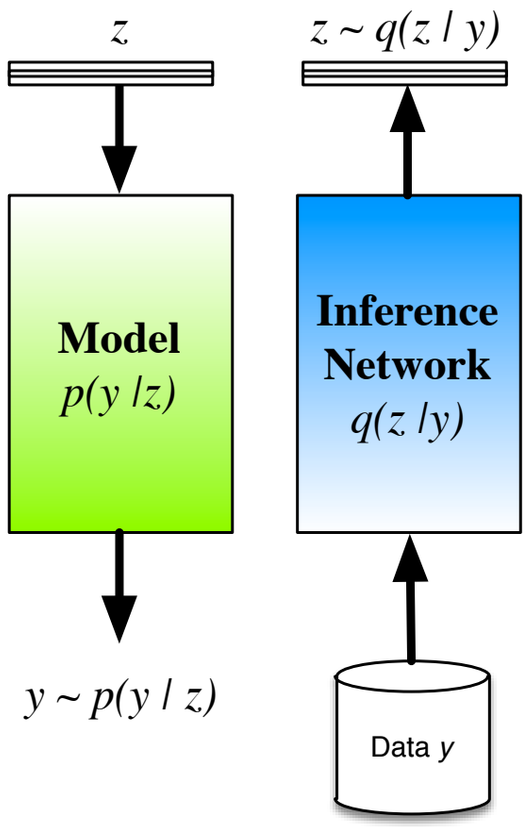
For i = 1, ... N

$$\phi_n \propto \nabla_\phi \mathbb{E}_{q_\phi(z)}[\log p_\theta(y_n|z_n)] - \nabla_\phi KL[q(z_n)\|p(z_n)]$$

Instead of solving this optimisation for every data point *n*, we can instead use a model.

M-step

$$\theta \propto \frac{1}{N}\sum_n \nabla_\theta \log p_\theta(y_n|z_n)$$



Inference network: q is an encoder or inverse model.

Parameters of q are now a set of global parameters used for inference of all data points - test and train.

Share the cost of inference (amortise) over all data.

Combines easily with mini-batches and Monte Carlo expectations.

Can jointly optimise variational and model parameters: no need for alternating optimisation.

Implementing your Variational Algorithm

Avoid deriving pages of gradient updates for variational inference.

Variational inference turns integration into optimisation:

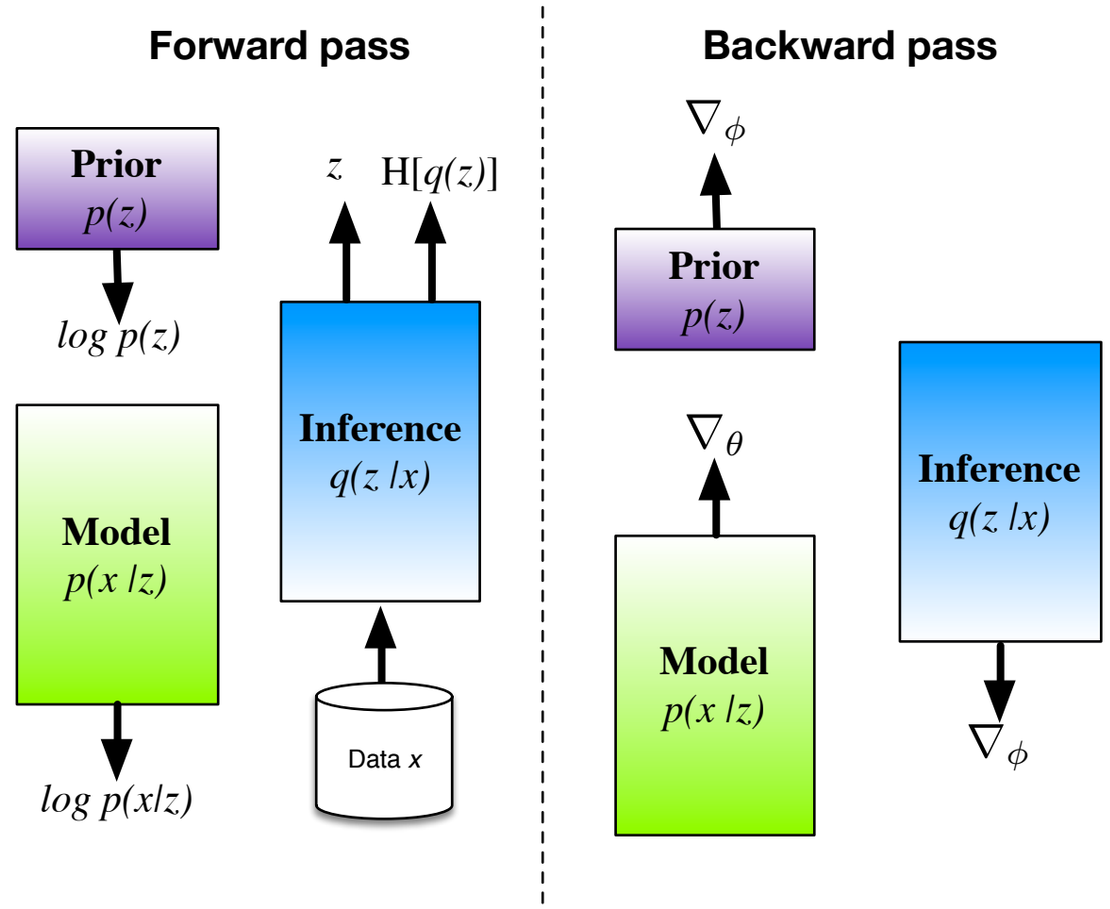
- **Automated Tools:**

  Differentiation: Theano, Torch7, Stan

  Message passing: infer.NET

- Stochastic gradient descent and other preconditioned optimisation.
- Same code can run on both GPUs or on distributed clusters.
- Probabilistic models are modular, can easily be combined.

$$\mathbb{E}_q[(-\log p(y|z) + \log q(z) - \log p(z)]$$



Ideally want probabilistic programming using variational inference.

Stochastic Backpropagation

A Monte Carlo method that works with continuous latent variables.

Original problem

$$\nabla_\xi \mathbb{E}_{q(z)}[f(z)]$$

Reparameterisation

$$z \sim \mathcal{N}(\mu, \sigma^2)$$
$$z = \mu + \sigma\epsilon \quad \epsilon \sim \mathcal{N}(0,1)$$

Backpropagation with Monte Carlo

$$\nabla_\xi \mathbb{E}_{\mathcal{N}(0,1)}[f(\mu + \sigma\epsilon)]$$
$$\mathbb{E}_{\mathcal{N}(0,1)}[\nabla_{\xi=\{\mu,\sigma\}} f(\mu + \sigma\epsilon)]$$

- Can use *any likelihood function,* avoids the need for additional lower bounds.
- *Low-variance*, unbiased estimator of the gradient.
- Can use just *one sample* from the base distribution.
- Possible for many distributions with location-scale or other known transformations, such as the CDF.

Monte Carlo Control Variate Estimators

More general Monte Carlo approach that can be used with both discrete or continuous latent variables.

Property of the score function: $$\nabla_\xi \log q_\xi(z|x) = \frac{\nabla_\xi q_\xi(z|x)}{q_\xi(z|x)}$$

Original problem $$\nabla_\phi \mathbb{E}_{q_\phi(z)}[\log p_\theta(y|z)]$$

Score ratio $$\mathbb{E}_{q_\phi(z)}[\log p_\theta(y|z) \nabla_\phi \log q(z|y)]$$

MCCV Estimate $$\mathbb{E}_{q_\phi(z)}[(\log p_\theta(y|z) - c) \nabla_\phi \log q(z|y)]$$

c is known as a ***control variate*** and is used to control the variance of the estimator.