

Memory-based Bayesian Reasoning with Deep Learning

Shakir Mohamed



November 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 knowledge integration, inference, and decision making that has established it as the key tool for data-efficient learning, uncertainty quantification and robust model composition, 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 and one that enhances our machine learning practice.

One powerful outcome of this convergence is our ability to develop systems for probabilistic inference with memory. A memory-based inference amortises the cost of probabilistic reasoning by cleverly reusing prior computations. To explore this, we shall take a statistical tour of deep learning, re-examine latent variable models and approximate Bayesian inference, and make connections to denoising auto-encoders and other stochastic encoder-decoder systems. In this way, we will make sense of what memory in inference might mean, and highlight the use of amortised inference in many other parts of machine learning.

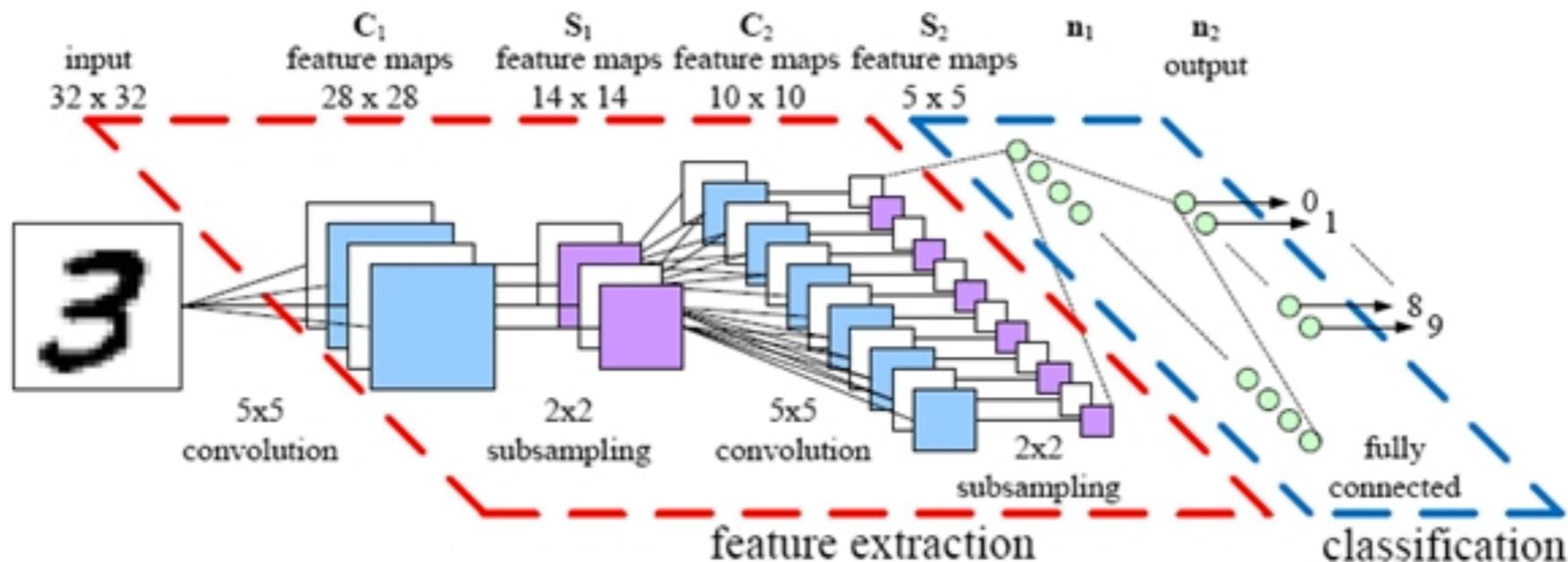


Deep Learning

Bayesian Reasoning

Better ML

Deep Learning

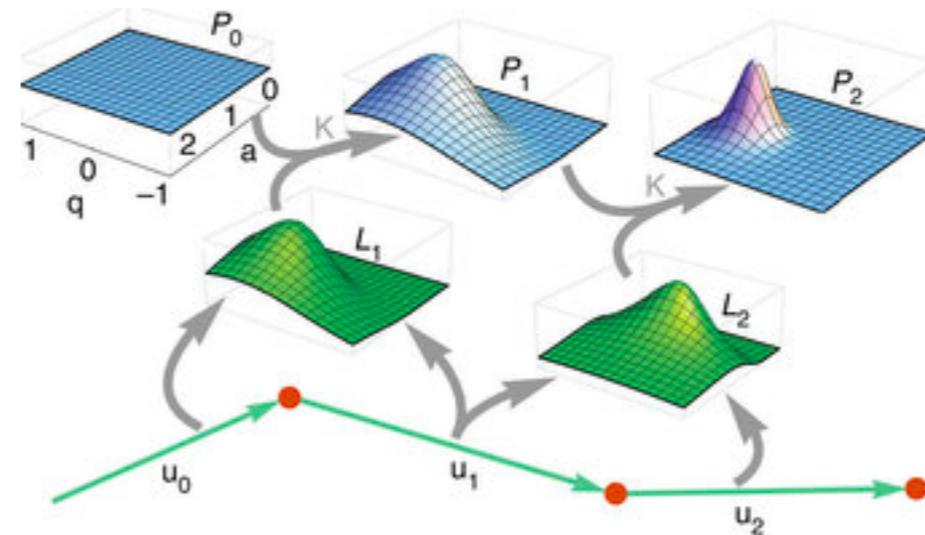
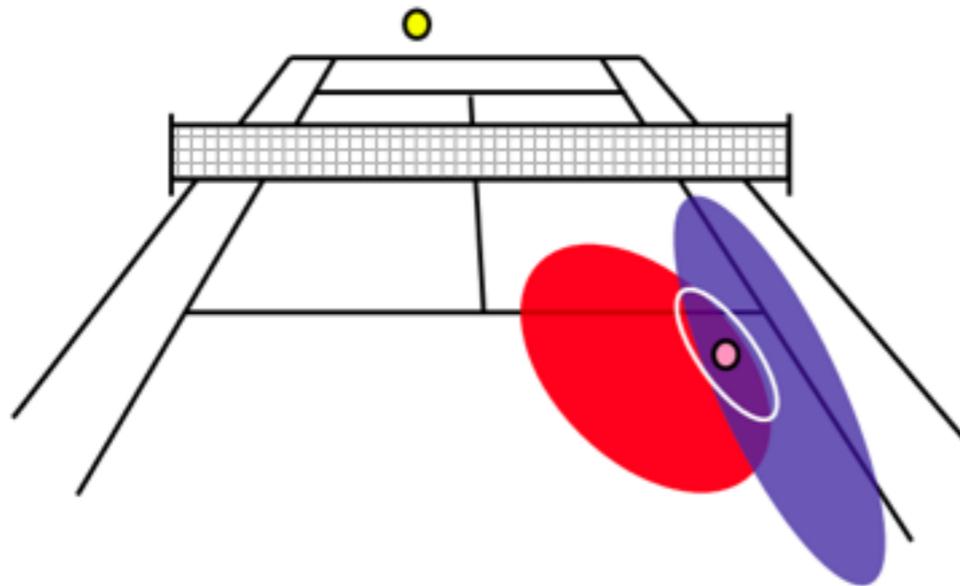


A framework for constructing flexible **models**

- + 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

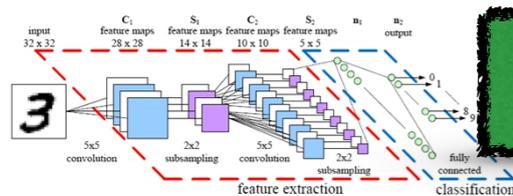


A framework for **inference and decision making**

- + 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.

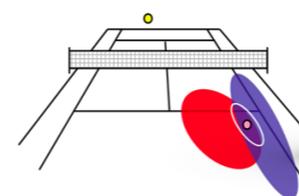
- Mainly conjugate and linear models
- Potentially intractable inference leading to expensive computation or long simulation times.

Two Streams of Machine Learning



Deep Learning

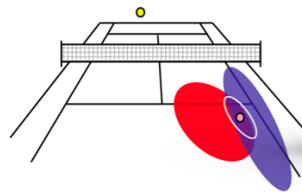
- + Rich non-linear models for classification and sequence prediction.
- + Scalable learning using stochastic approximation and conceptually simple.
- + Easily composable with other gradient-based methods
- Only point estimates
- Hard to score models, do selection and complexity penalisation.



Bayesian Reasoning

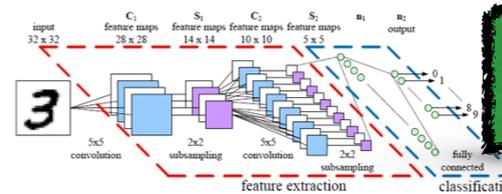
- Mainly conjugate and linear models
- Potentially intractable inference, computationally expensive or long simulation time.
- + Unified framework for model building, inference, prediction and decision making
- + Explicit accounting for uncertainty and variability of outcomes
- + Robust to overfitting; 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 Convergence of research streams

- ❖ Deep latent variable models and auto-encoders
- ❖ Inference with memory

3 More amortised probabilistic inference

- ❖ Message passing, posterior predictive distributions, semi-supervised learning, sequential inference with memory.

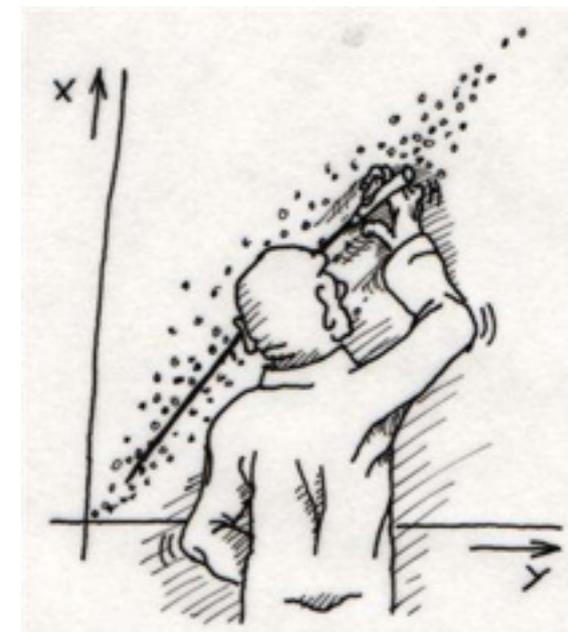
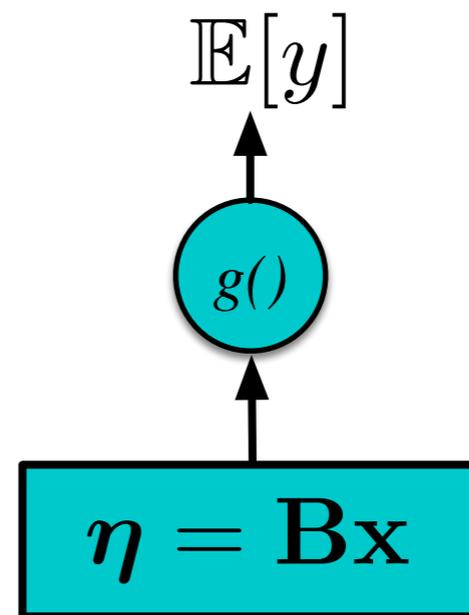
A (Statistical) Review of Deep Learning

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.
- ◆ $g(\cdot)$ is an *inverse link function* that we'll refer to as an activation function.

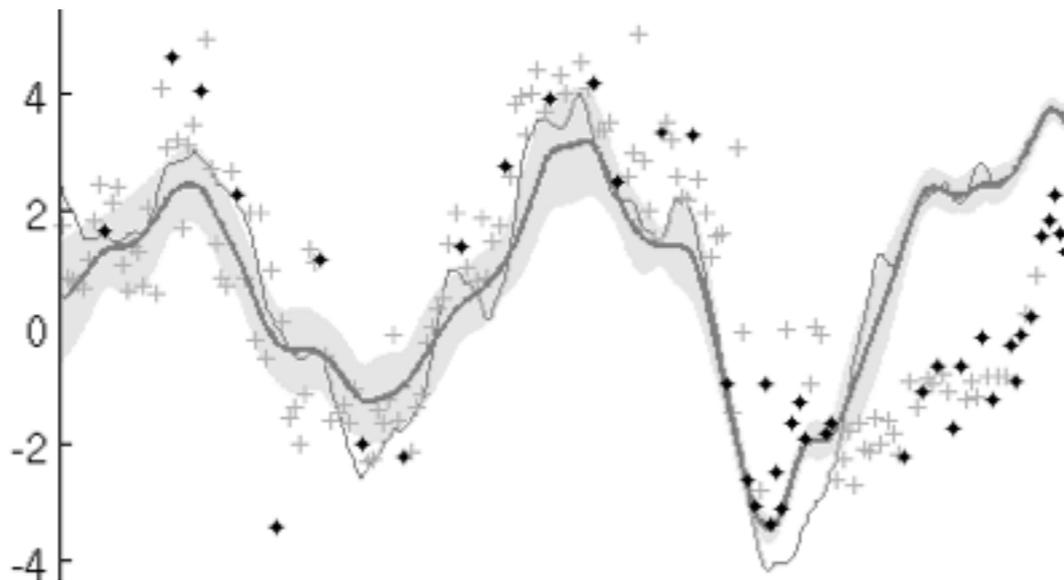


Target	Regression	Link	Inv link	Activation
Real	Linear	Identity	Identity	
Binary	Logistic	Logit $\log \frac{\mu}{1-\mu}$	Sigmoid $\frac{1}{1+\exp(-\eta)}$	Sigmoid
Binary	Probit	Inv Gauss CDF $\Phi^{-1}(\mu)$	Gauss CDF $\Phi(\eta)$	Probit
Binary	Gumbel	Compl. log-log $\log(-\log(\mu))$	Gumbel CDF $e^{-e^{-x}}$	
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

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

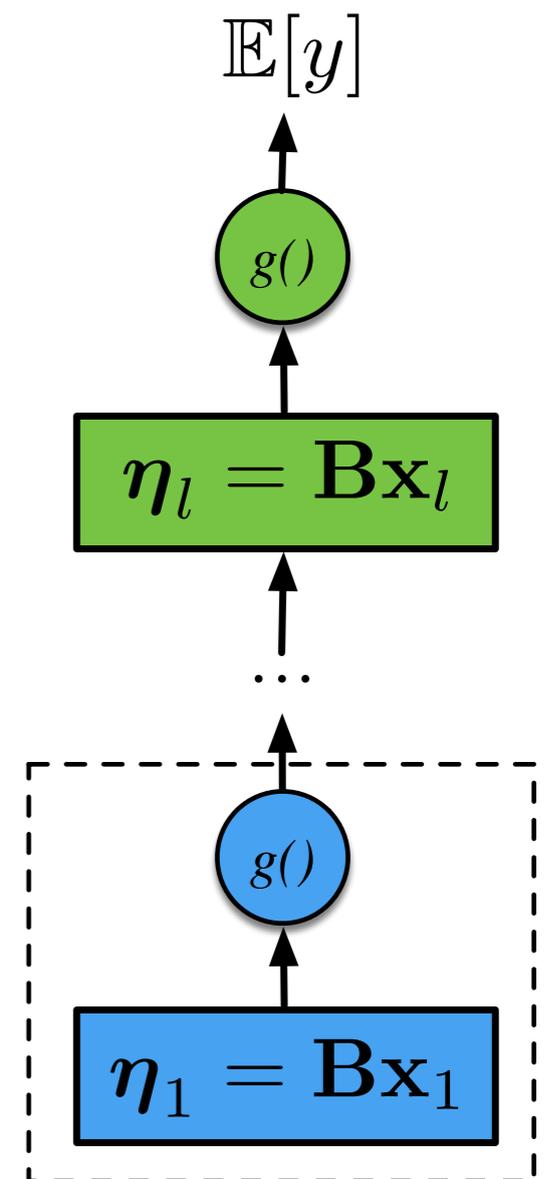
A (Statistical) Review of Deep Learning



Recursive Generalised Linear Regression

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

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



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:

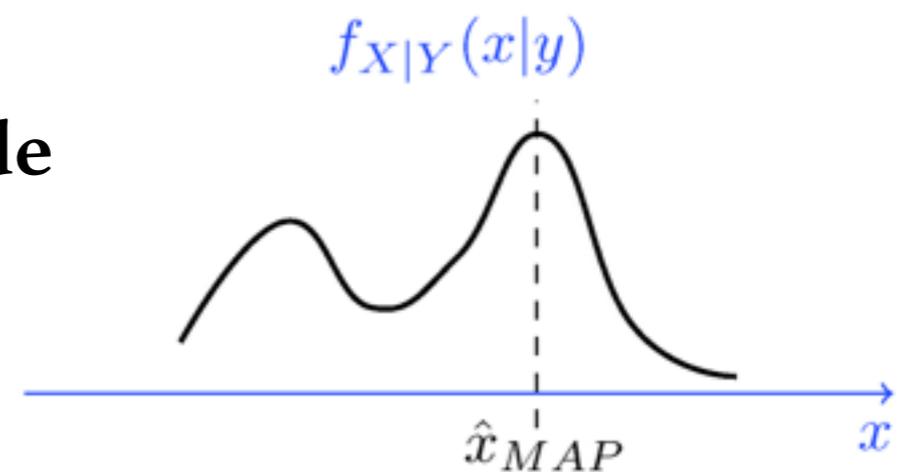
- ▶ Large data sets
- ▶ Input noise/jittering and data augmentation/expansion.
- ▶ L2 /L1 regularisation (Weight decay, Gaussian prior)
- ▶ Binary or Gaussian Dropout
- ▶ Batch normalisation

More robust loss function using MAP estimation instead.

More Robust Learning

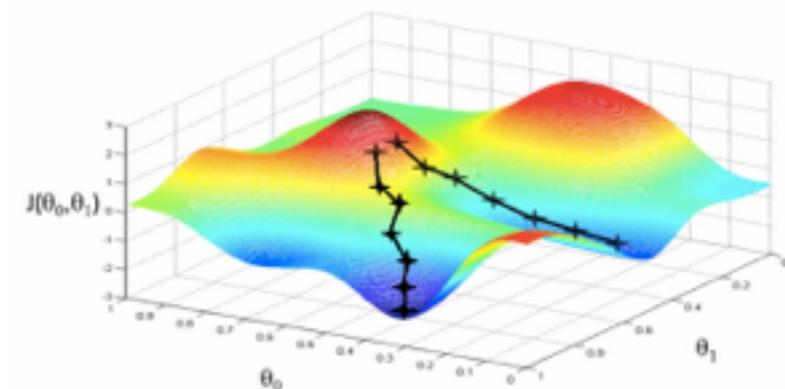
MAP estimators and limitations

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

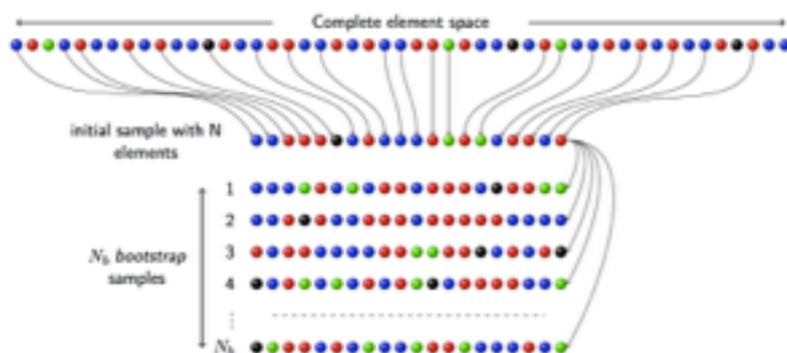


1. Sensitivities affect gradients and can make learning hard

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



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



Can generate confidence intervals and bootstrap estimates.

Towards Bayesian Reasoning

Proposed solutions have not fully dealt with the underlying issues.

Issues arise as a consequence of:

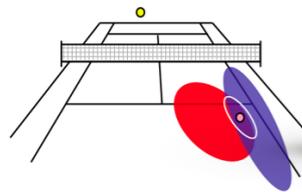
- ▶ Reasoning only about the most likely solution and
- ▶ Not maintaining knowledge of the underlying variability (and averaging over this).
- ▶ Motivates Deep Learning with Bayesian reasoning

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.

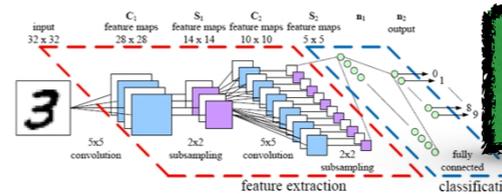
Bayesian reasoning over some, but not all parts of our models (yet).

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 Convergence of research streams

- ❖ Deep latent variable models and auto-encoders
- ❖ Inference with memory

3 More amortised probabilistic inference

- ❖ Message passing, posterior predictive distributions, semi-supervised learning, sequential inference with memory.

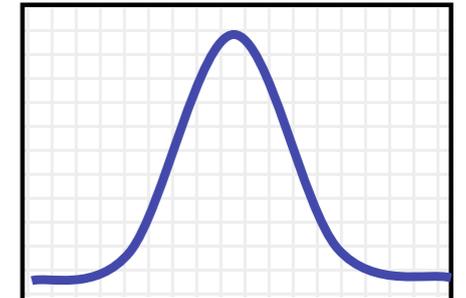
Density Estimation and Latent Variable Models

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

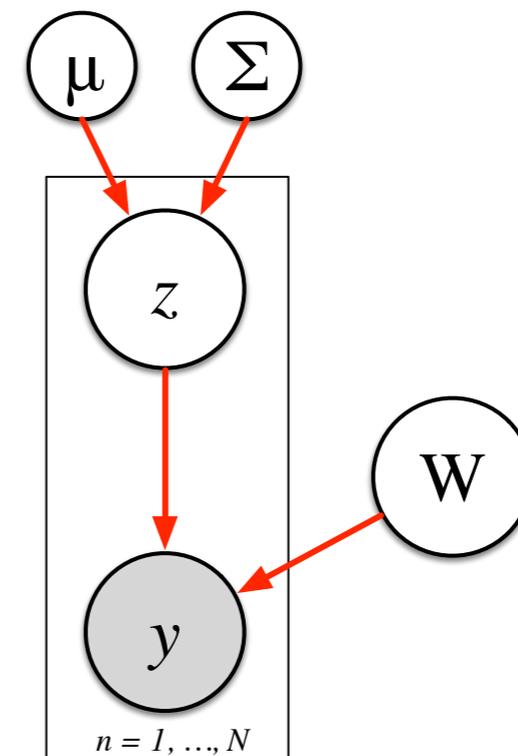
$$\mathbf{z} \sim \mathcal{N}(\mathbf{z} | \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Observation Model

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

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

Exponential fam natural parameters $\boldsymbol{\eta}$.



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

Deep Generative Models

Rich extension of previous model using deep neural networks.
Non-linear factor analysis, non-linear Gaussian belief networks,
Deep latent Gaussian models (DLGM).

Latent Variables (Stochastic layers)

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

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

DLGM

Deterministic layers

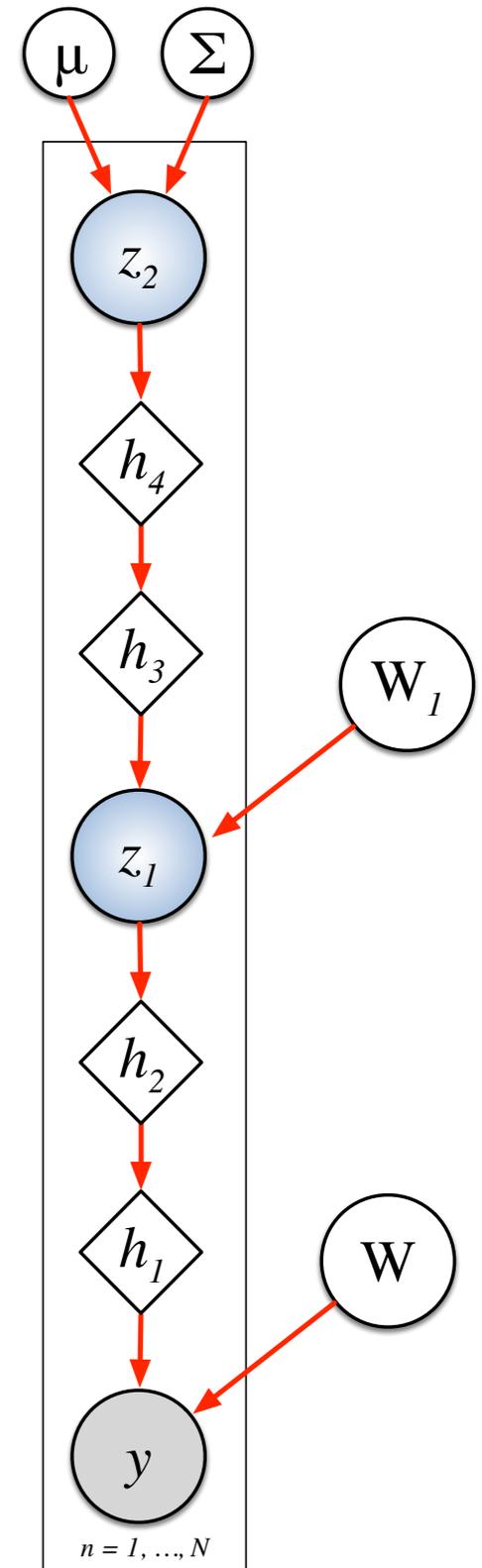
$$h_i(\mathbf{x}) = \sigma(\mathbf{A}\mathbf{x} + \mathbf{c})$$

Observation Model

$$\boldsymbol{\eta} = \mathbf{W}\mathbf{h}_1 + \mathbf{b}$$

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

Can also use non-exponential family.



Deep Latent Gaussian Models

Our inferential tasks are:

1. Explain this data

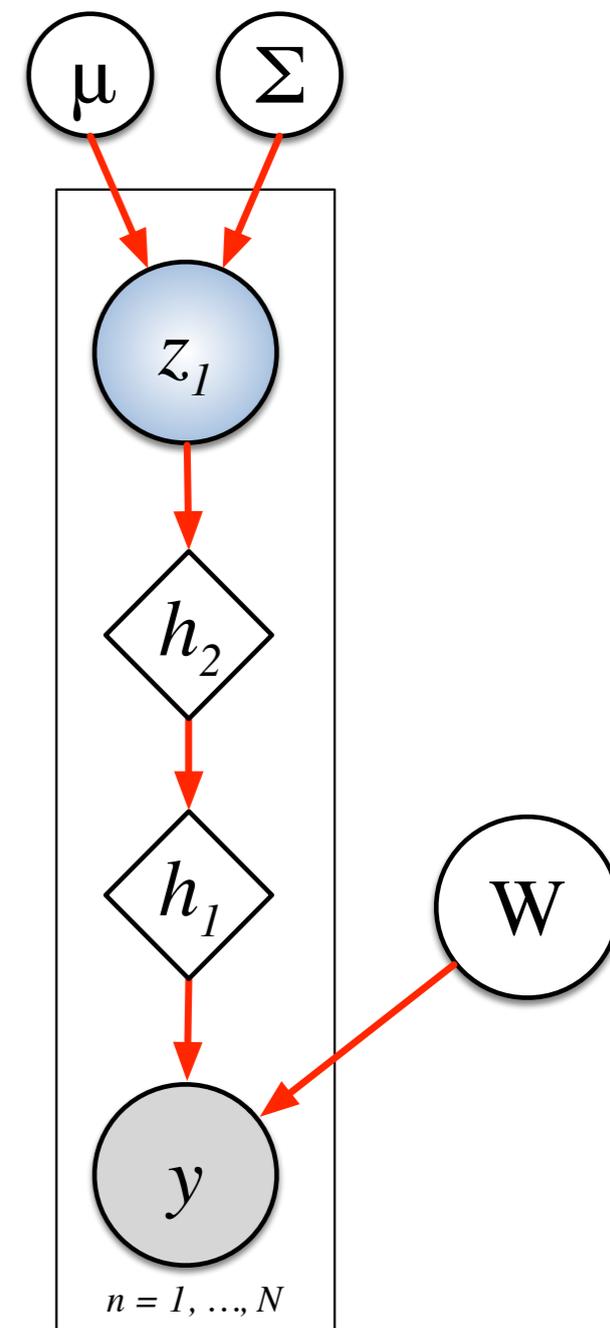
$$p(\mathbf{z}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{z})p(\mathbf{z})$$

2. Make predictions:

$$p(\mathbf{y}^i|\mathbf{y}^{\setminus i}) = \int p(\mathbf{y}^i|\mathbf{z})p(\mathbf{z}|\mathbf{y}^{\setminus i})d\mathbf{z}$$

3. Choose the best model

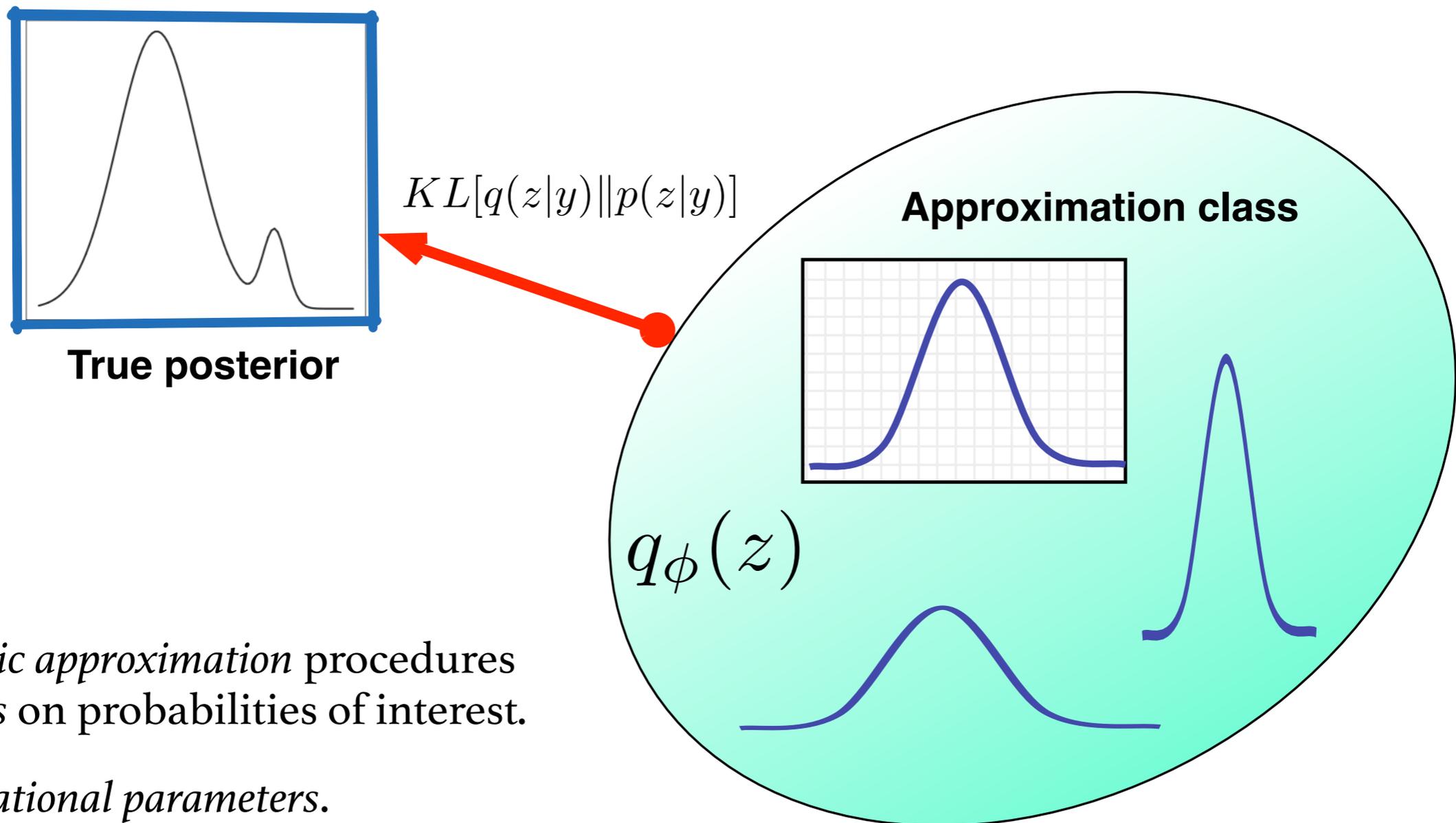
$$p(\mathbf{y}|\mathcal{M}) = \int p(\mathbf{y}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$



What is a Variational Method?

Variational Principle

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



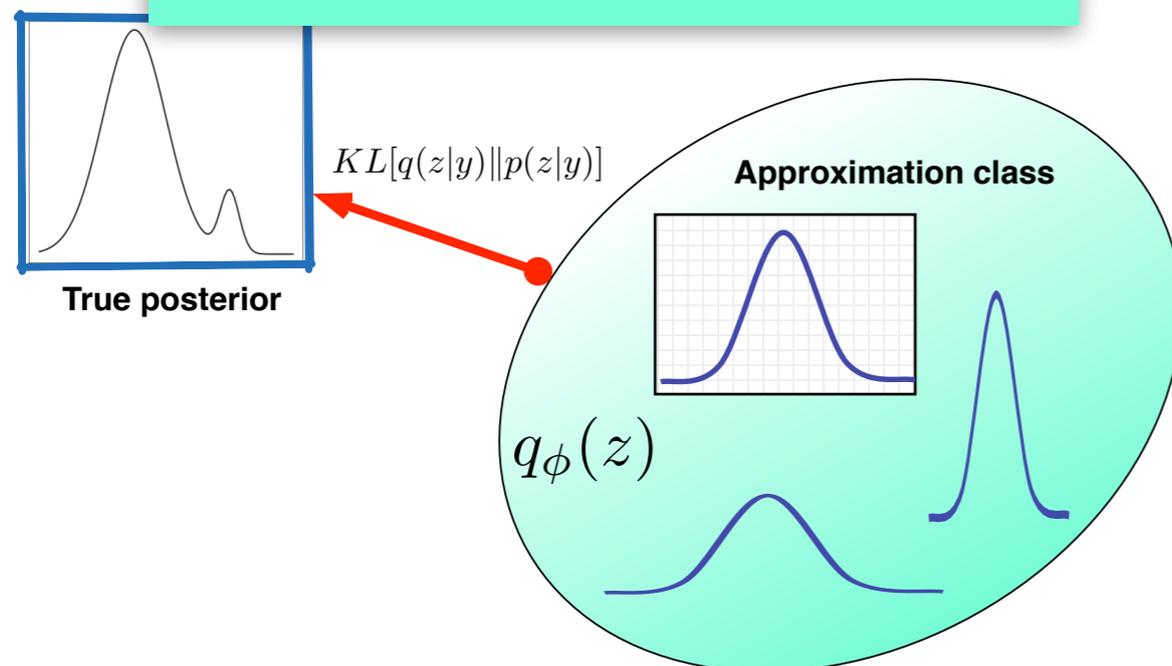
Deterministic approximation procedures with *bounds* on probabilities of interest.

Fit the *variational parameters*.

Variational Inference

Use tools from approximate inference to handle intractable integrals.

$$p(\mathbf{y}|\mathcal{M}) = \int p(\mathbf{y}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$



- **Reconstruction cost:** Expected log-likelihood measures how well $q(z)$ is 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.

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

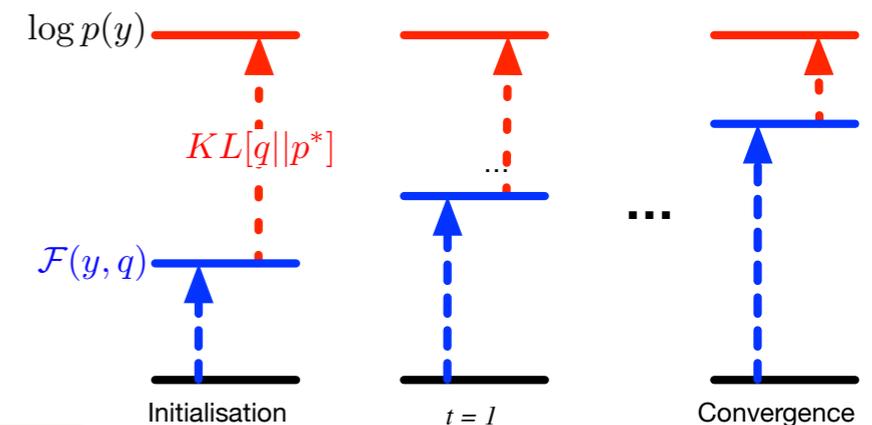
The equation is presented with two yellow boxes: the first box contains $\mathbb{E}_{q(z)} [\log p(y|z)]$ and is labeled "Reconstruction" above it; the second box contains $KL[q(z)||p(z)]$ and is labeled "Penalty" above it.

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

Memoryless Inference: Variational EM

Optimise the variational lower bound using the EM algorithm - using a **stochastic gradient based on a mini-batch** of data.

Many names: *online EM*, *stochastic approximation EM*, *stochastic variational inference*.



Repeat:

E-step

For $i = 1, \dots, N$ (*Inference*)

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

N is a mini-batch: sampled with replacement from the full data set or received online.

M-step

$$\theta \propto \frac{1}{N} \sum_n \mathbb{E}_{q_{\phi}(z)} [\nabla_{\theta} \log p_{\theta}(y_n | z_n)]$$

(*Parameter Learning*)

Memoryless: Any inference computations are discarded after the M-step update

Inference with Memory

Repeat:

E-step

For $i = 1, \dots, N$

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

Instead of solving for every observation, use a model.

M-step

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

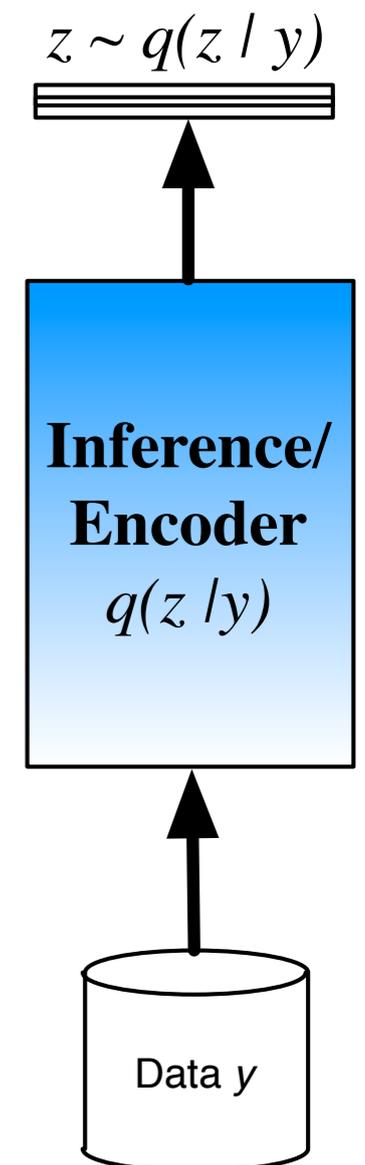
Inference network: q is an encoder or inverse model.

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

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

Joint optimisation of variational and model parameters.

Inference networks provide an efficient mechanism for **posterior inference with memory**



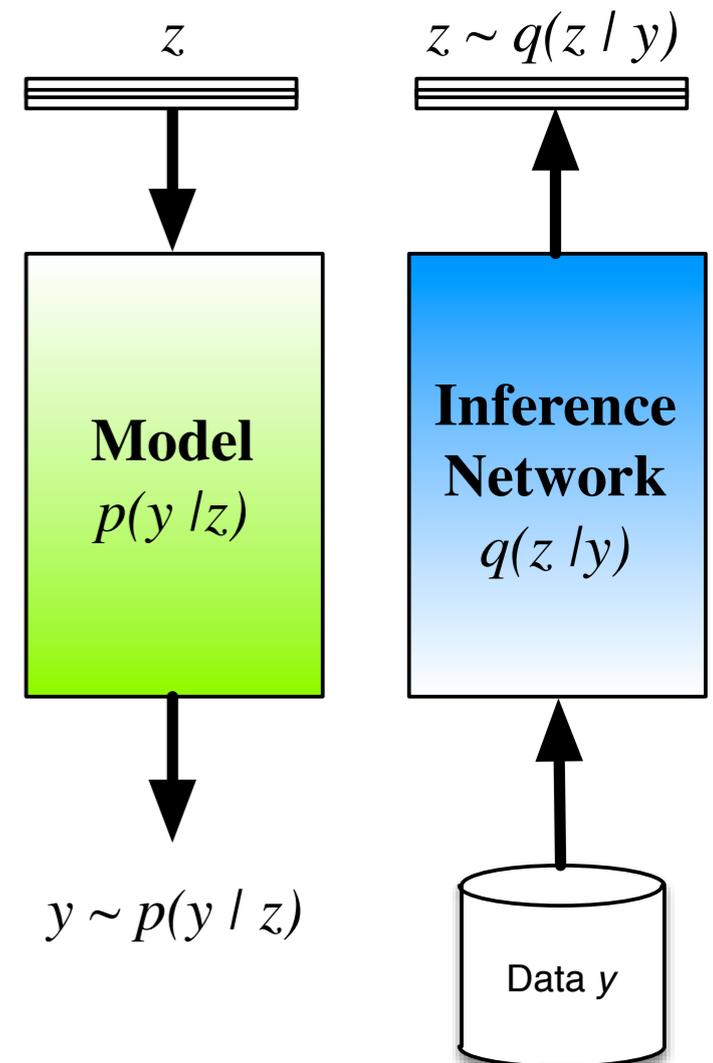
Auto-encoders and Inference in DGMs

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

Approx. Posterior Reconstruction Penalty

Stochastic encoder-decoder system to implement variational inference.

- **Model (Decoder):** likelihood $p(y|z)$.
- **Inference (Encoder):** variational distribution $q(z|y)$
- Transforms an auto-encoder into a generative model



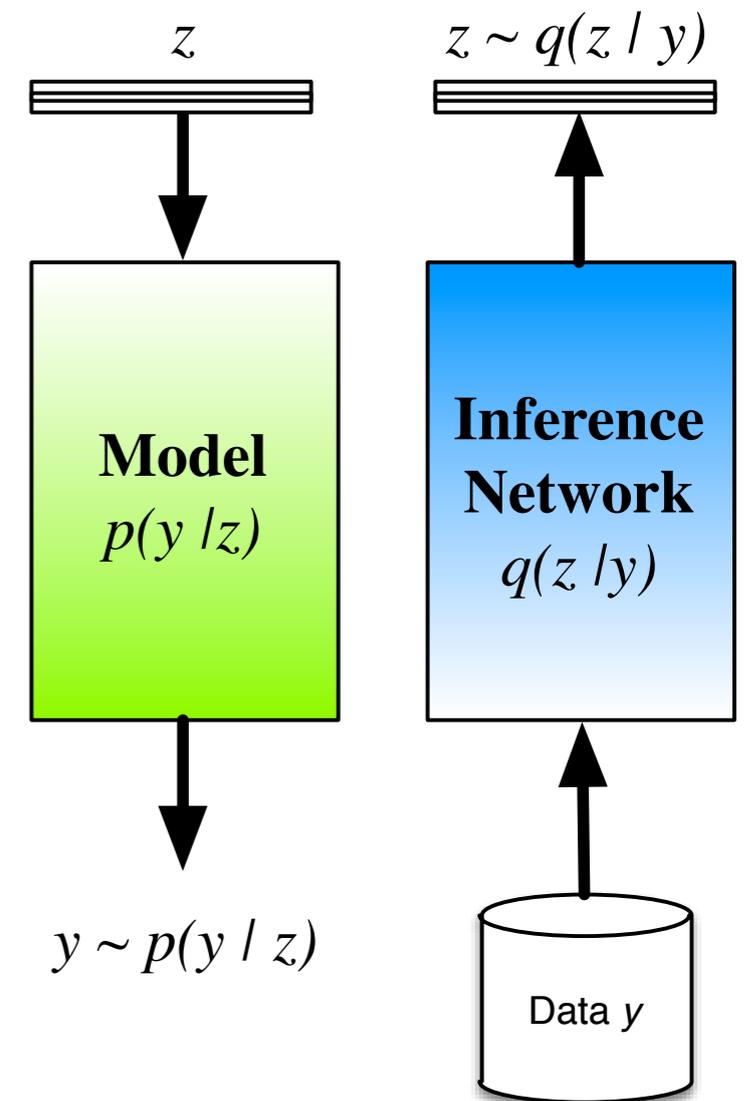
Specific combination of **variational inference** in **latent variable models** using **inference networks**
Variational Auto-encoder

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

What Have we Gained

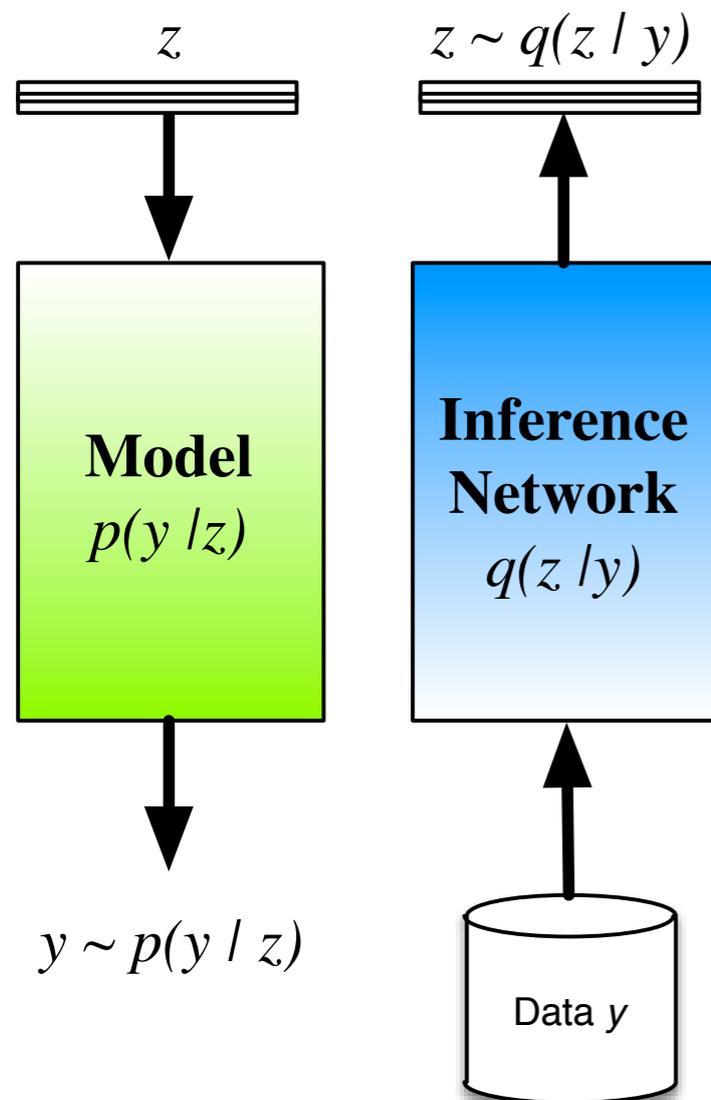
- + Transformed an auto-encoders into more interesting **deep generative models**.
- + Rich new class of density estimators built with **non-linear models**.
- + Used a **principled approach** for deriving loss functions that automatically include appropriate penalty functions.
- + Explained **how an 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.

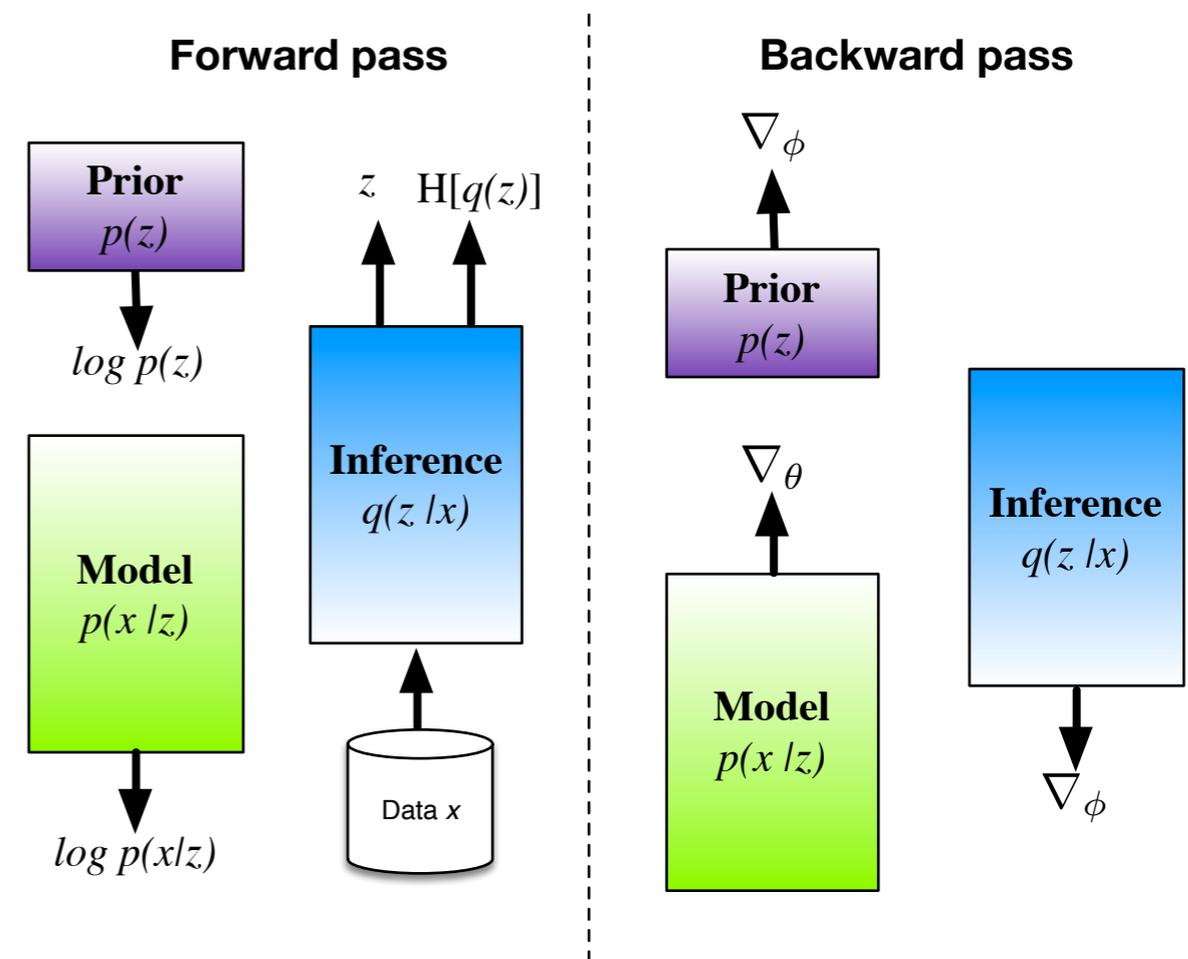
A true marriage of Bayesian Reasoning and Deep Learning

Implementing your Variational Algorithm

Variational inference turns integration into optimisation:

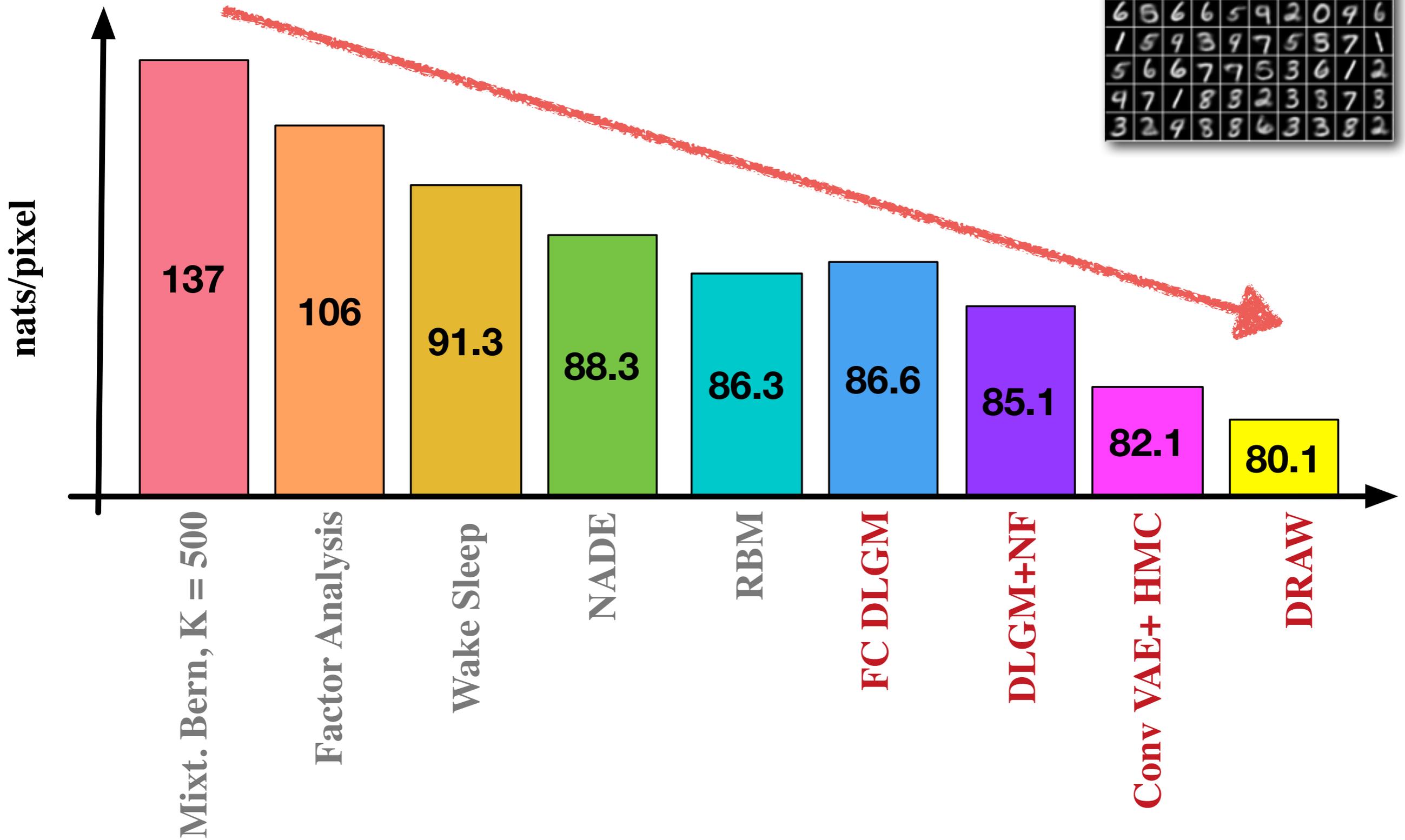
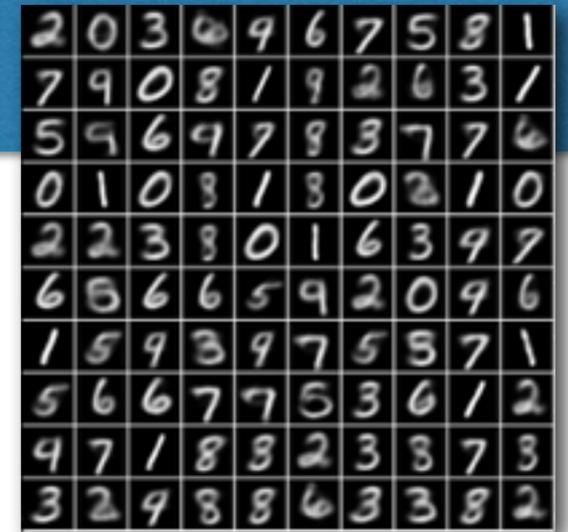
- **Automated Tools:**
 - Differentiation:** Theano, Torch7, Stan
 - Message passing:** infer.NET
- Stochastic gradient descent and other preconditioned optimisation.
- Same code can run on both GPUs or on distributed clusters.
- Probabilistic models are modular, can easily be combined.

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



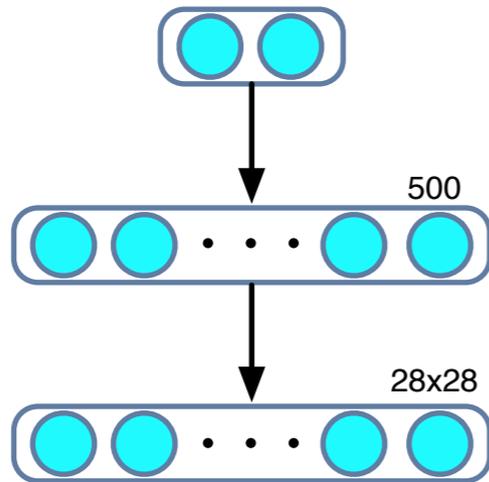
Ideally want probabilistic programming using variational inference.

MNIST Improvements

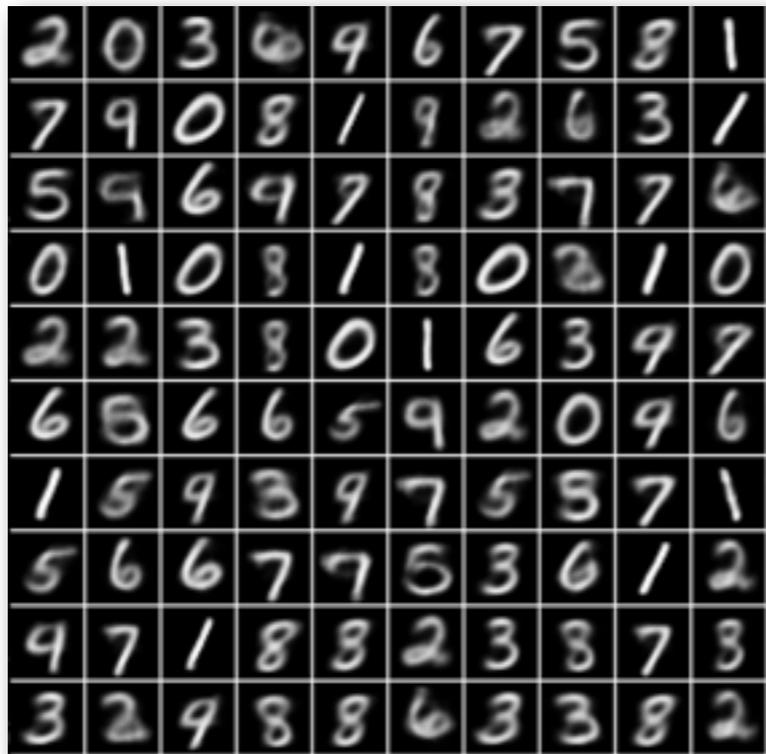


Data Visualisation

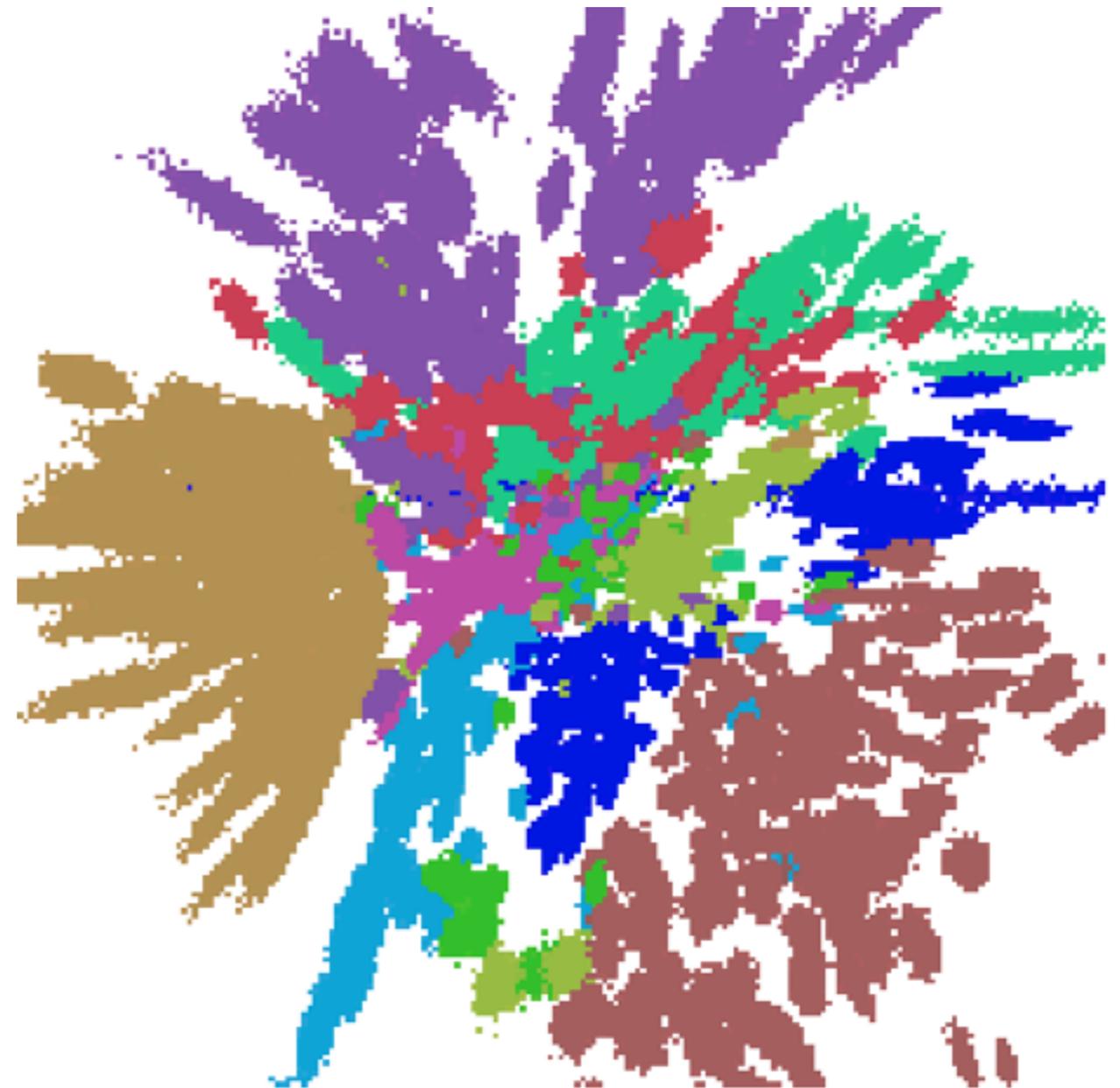
MNIST Handwritten digits



DLGM



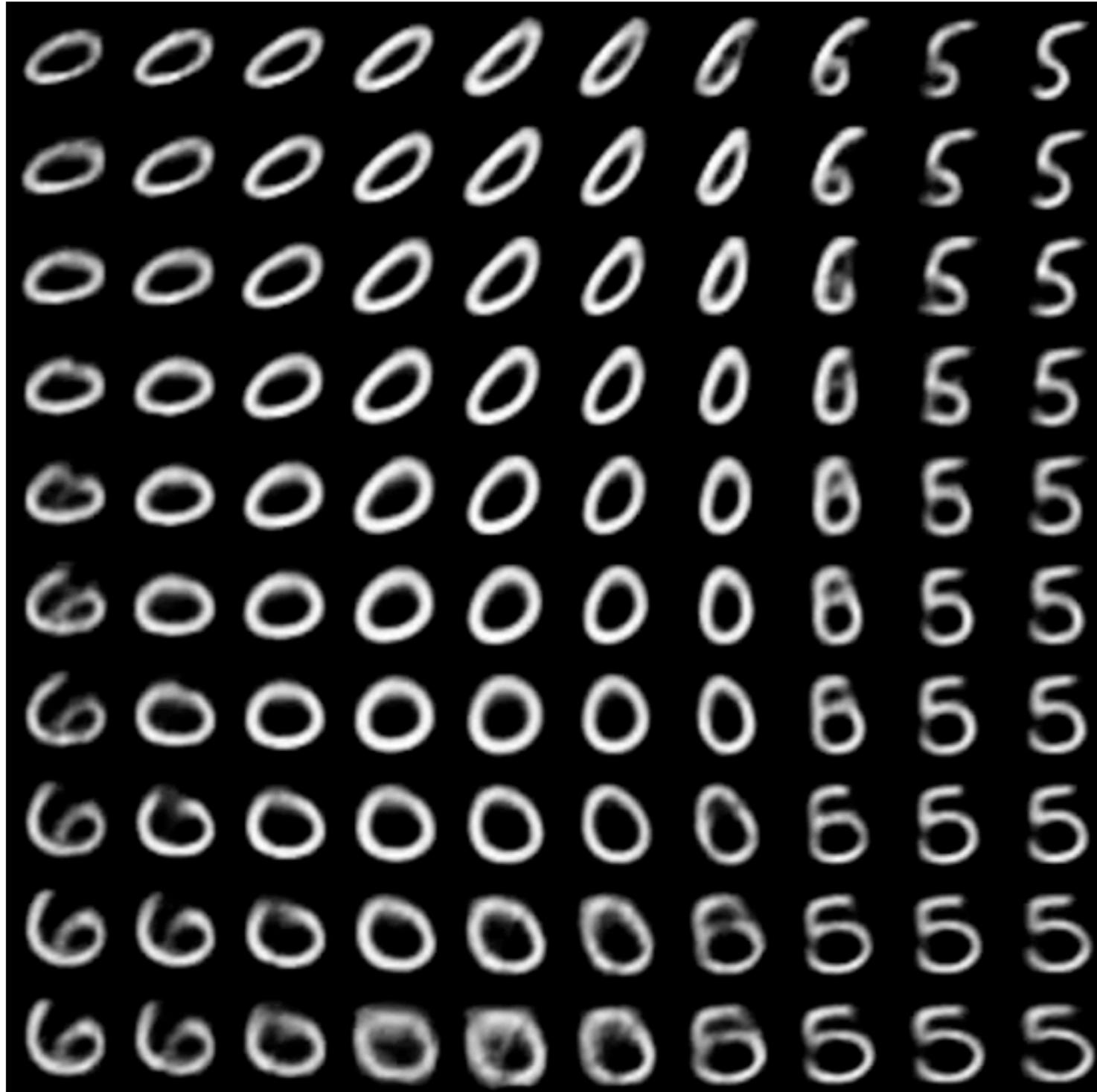
Samples from 2D latent model



Labels in 2D latent space

Visualising MNIST in 3D

DLGM



Missing Data Imputation

Original Data

■ unobserved pixels

Inferred Image

10%
observed

50%
observed

DLGM



Analogical Reasoning

Semi-supervised DLGM

4	0	1	2	3	4	5	6	7	8	9
9	0	1	2	3	4	5	6	7	8	9
5	0	1	2	3	4	5	6	7	8	9
4	0	1	2	3	4	5	6	7	8	9
2	0	1	2	3	4	5	6	7	8	9
7	0	1	2	3	4	5	6	7	8	9
5	0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8	9
7	0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8	9

40	1	2	3	4	5	6	7	8	9	0
15	1	2	3	4	5	6	7	8	9	0
36	10	20	30	40	50	60	70	80	90	00
27	1	2	3	4	5	6	7	8	9	0
13	11	12	13	14	15	16	17	18	19	10
30	1	2	3	4	5	6	7	8	9	0
61	11	21	31	41	51	61	71	81	91	01
20	10	20	30	40	50	60	70	80	90	00
28	21	22	23	24	25	26	27	28	29	20
22	21	22	23	24	25	26	27	28	29	20

Data Simulation

DLGM



Data



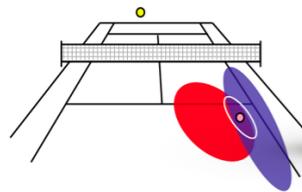
Samples

Data Simulation

DRAW

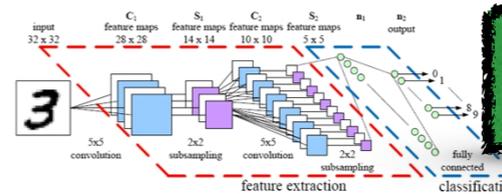


Outline



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 Convergence of research streams

- ❖ Deep latent variable models and auto-encoders
- ❖ Inference with memory

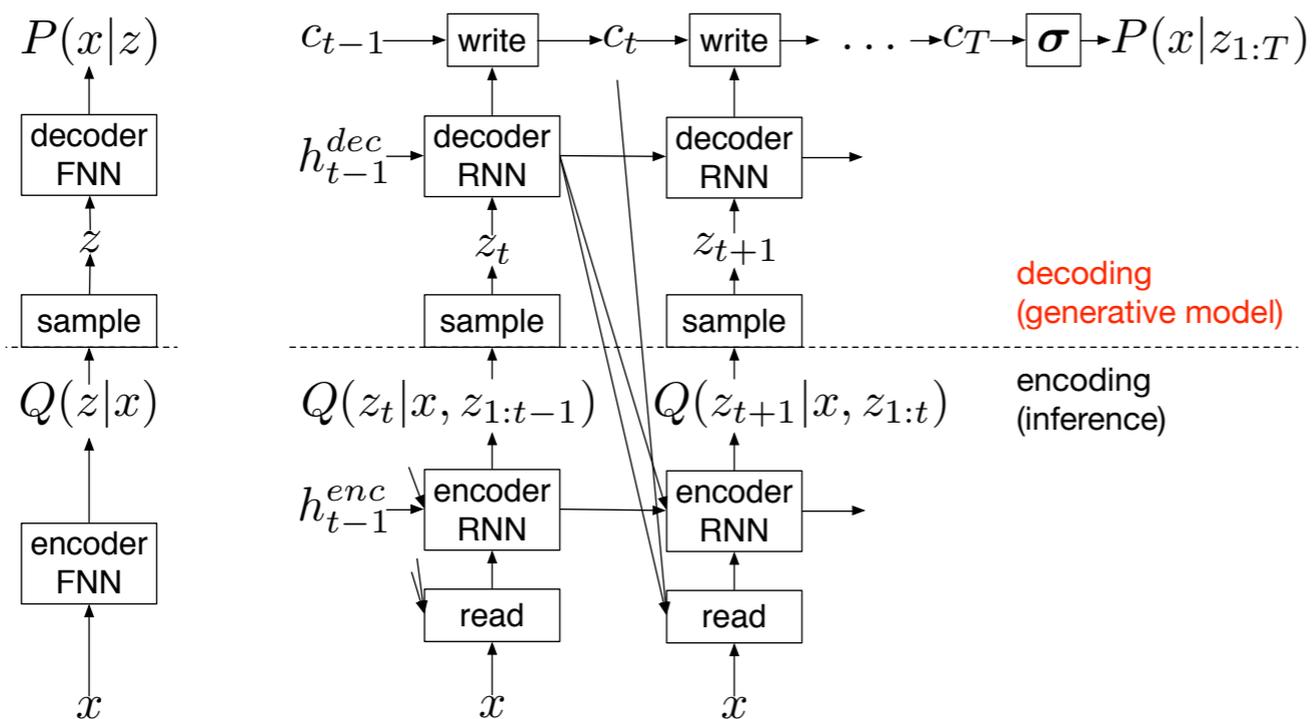
3 More amortised probabilistic inference

- ❖ Message passing, posterior predictive distributions, semi-supervised learning, sequential inference with memory.

Sequential Inference with Memory

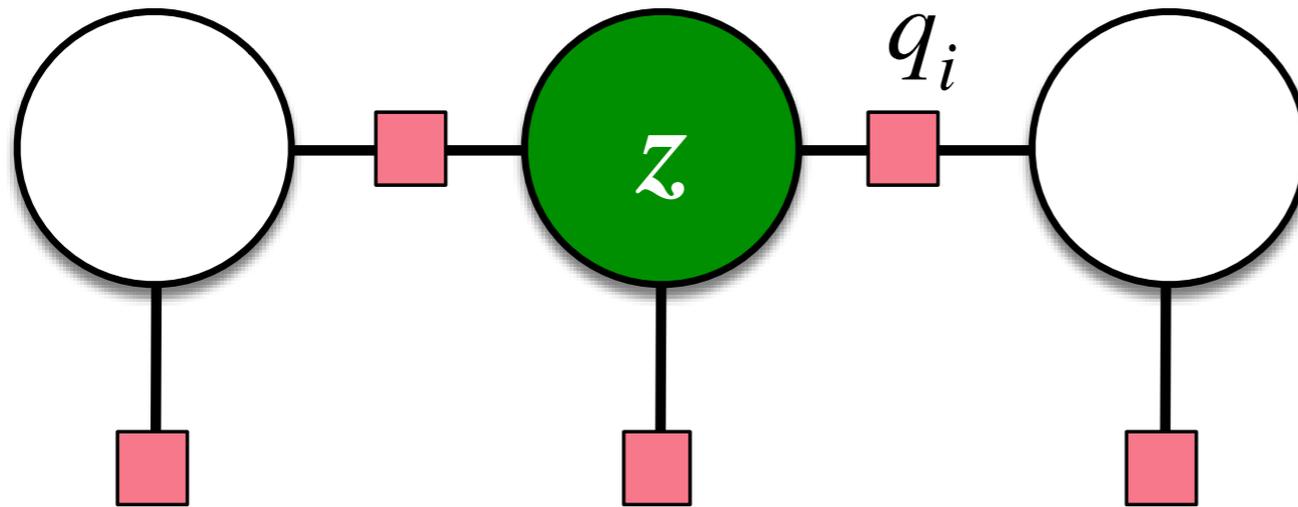
We can also combine other tools from deep learning to design even more powerful generative models: recurrent networks and attention.

DRAW



Message Passing with Memory

Expectation Propagation



Factorised assumption

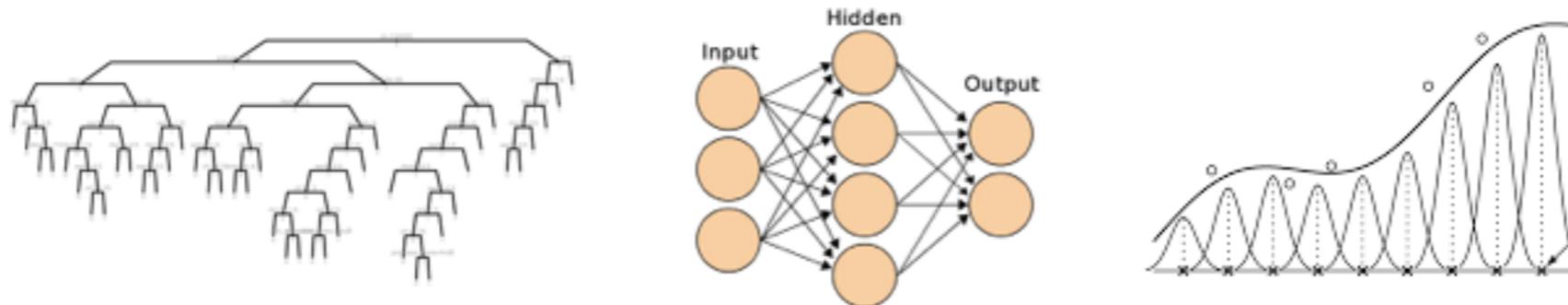
$$p(z|\mathcal{D}) = \prod_i f_i(z)$$
$$\approx \prod_i q_i(z) = q(z)$$

Memoryless inference: $q_i = \arg \min_{q \in \mathcal{Q}} D_{KL}[f^i q^{\setminus i} || q^i q^{\setminus i}]$

Amortised Inference:

Use a model

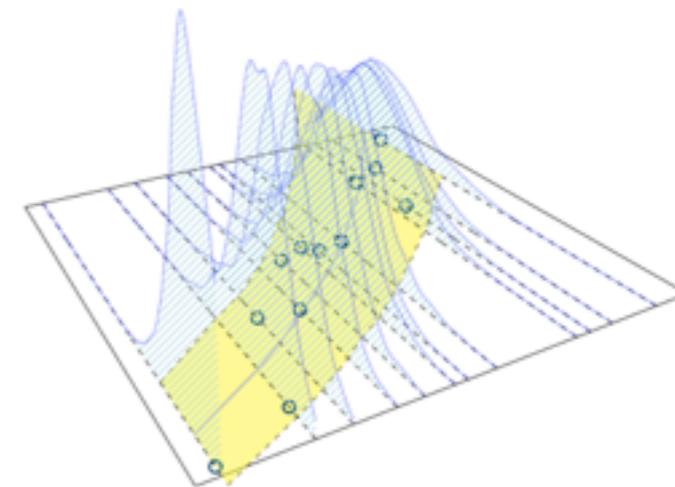
$$q_i = h(\{q^i\}, \mathcal{D}; \theta)$$



Amortised Posterior Predictive Distributions

Posterior predictive distributions were one of our desired inferential tasks.

$$p(y^* | x^*, X, Y) = \int p(y^* | x^*, W) p(W | X, Y) dW$$



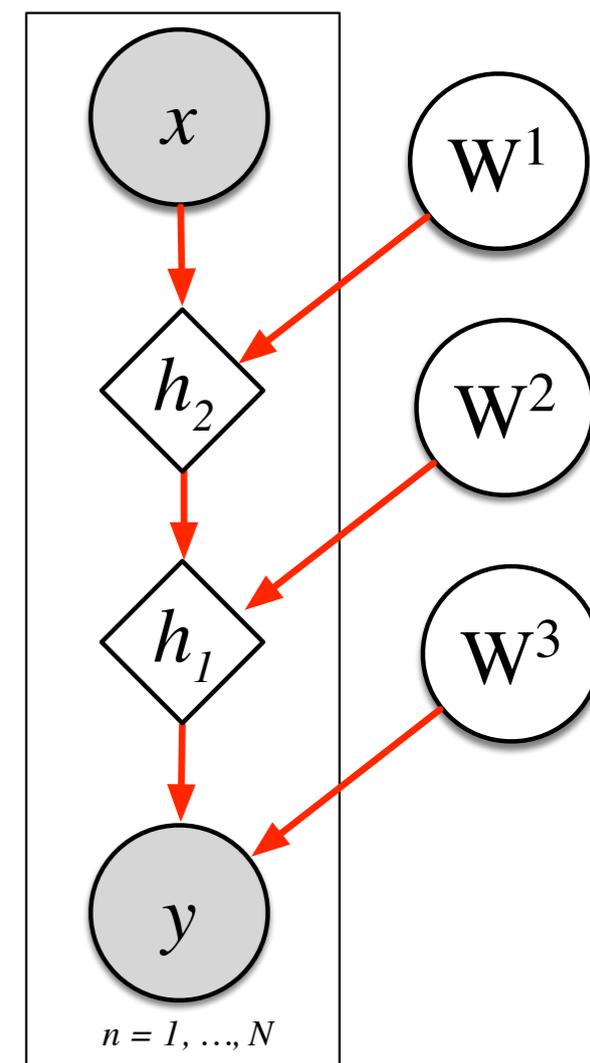
Dark Knowledge

Memoryless prediction: compute by Monte Carlo

$$q(y^* | x^*) = \frac{1}{S} \sum_{s=1}^S p(y^* | x^*, W^{(s)}) \quad W^{(s)} \sim p(W | X, Y)$$

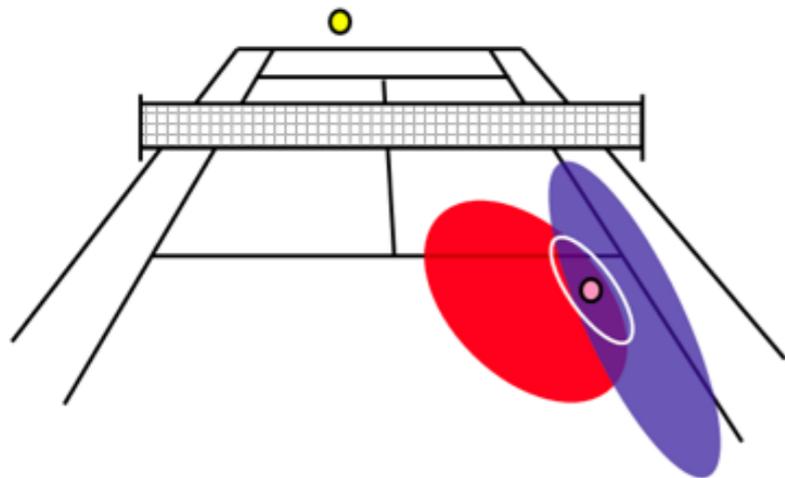
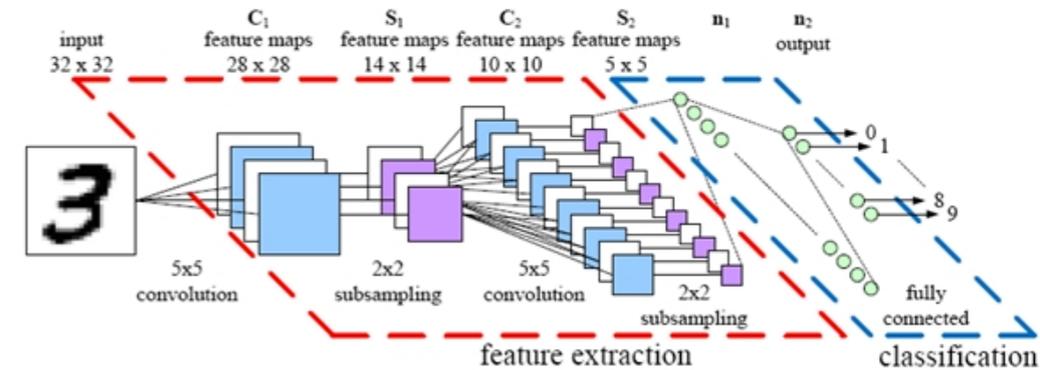
Amortised prediction: use a model

$$p(y^* | x^*, X, Y) = f(x^*, \theta)$$



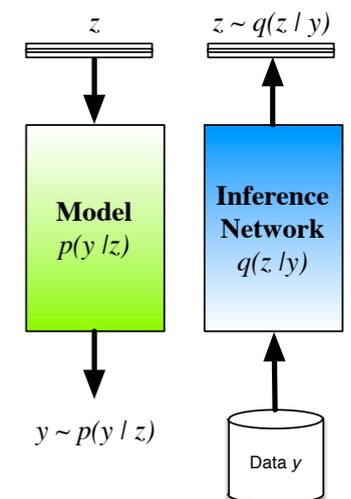
In Review ...

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.

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

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

Stochastic Backpropagation

A Monte Carlo method that works with continuous latent variables

Original problem

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

Pathwise derivative approach

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-function or likelihood ratio approach.

Score ratio

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

Score Estimator

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

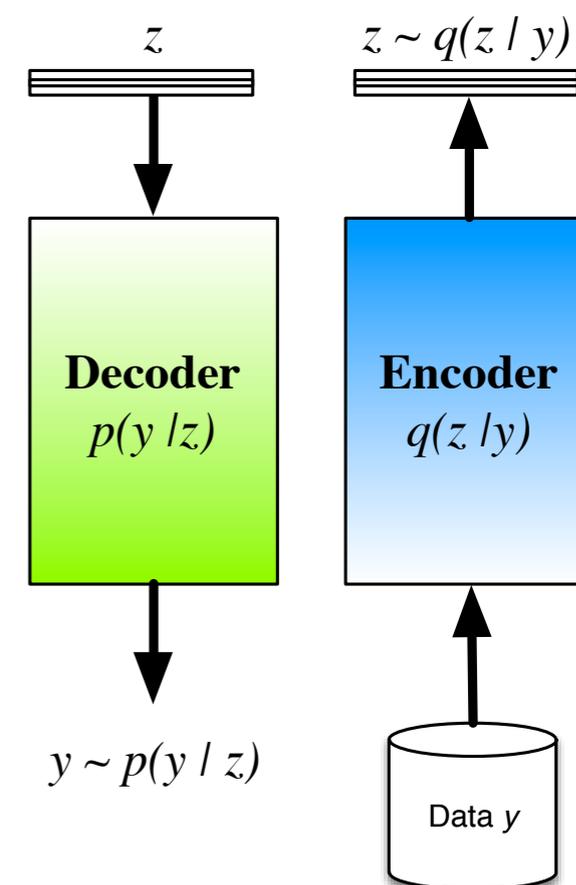
c is known as a **control variate** and is used to control the variance of the estimator.

Denoising Auto-encoders (DAE)

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

Stochastic encoder Reconstruction Penalty

- DAE: A mechanism for finding representations or features of data (i.e. latent variable explanations).
 - No natural handling of missing data
 - No representation of variability of the embedding space.
- **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.*

Minimum Description Length (MDL)

$$\mathcal{F}(y, q) = \underbrace{\mathbb{E}_{q(z)}[\log p(y|z)]}_{\text{Data code-length}} - \underbrace{KL[q(z) \parallel p(z)]}_{\text{Hypothesis code}}$$

Stochastic encoder

Stochastic encoder-decoder systems implement variational inference.

- 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 $p(y|z)$.

