Probabilistic modelling for NLP powered by deep learning

Wilker Aziz
University of Amsterdam

April 3, 2018

Outline

Problems

Supervised problems

"learn a distribution over observed data"

- sentences in natural language
- images, . . .

Problems

Supervised problems

"learn a distribution over observed data"

- sentences in natural language
- images, ...

Unsupervised problems

"learn a distribution over observed and unobserved data"

- sentences in natural language + parse trees
- images + bounding boxes, ...

Supervised problems

We have data $x^{(1)}, \ldots, x^{(N)}$ e.g.

- sentences, images, ...

generated by some **unknown** procedure

Supervised problems

We have data $x^{(1)}, \dots, x^{(N)}$ e.g.

- sentences, images, ...

generated by some **unknown** procedure
which we assume can be captured by a probabilistic model

Supervised problems

We have data $x^{(1)}, \dots, x^{(N)}$ e.g.

- sentences, images, ...

generated by some **unknown** procedure
which we assume can be captured by a probabilistic model

- with **known** probability (mass/density) function e.g.

$$\underbrace{X \sim \text{Cat}(\pi_1, \dots, \pi_K)}_{\text{e.g. nationality}} \qquad \text{or} \qquad \underbrace{X \sim \mathcal{N}(\mu, \sigma^2)}_{\text{e.g. height}}$$

Supervised problems

We have data $x^{(1)}, \ldots, x^{(N)}$ e.g.

- sentences, images, ...

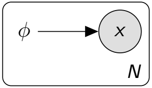
generated by some **unknown** procedure
which we assume can be captured by a probabilistic model

- with **known** probability (mass/density) function e.g.

$$\underbrace{X \sim \text{Cat}(\pi_1, \ldots, \pi_K)}_{\text{e.g. nationality}} \qquad \text{or} \qquad \underbrace{X \sim \mathcal{N}(\mu, \sigma^2)}_{\text{e.g. height}}$$

estimate parameters that assign maximum likelihood to observations

Multiple problems, same language



(Conditional) Density estimation

	Side information (ϕ)	Observation (x)
Parsing	a sentence	parse tree
Translation	a sentence in French	translation in English
Captioning	an image	caption in English
Entailment	a text and hypothesis	entailment relation

Where does deep learning kick in?

Let ϕ be all side information available
e.g. deterministic *inputs/features*

Where does deep learning kick in?

Let ϕ be all side information available
e.g. deterministic *inputs/features*

Have neural networks predict parameters of our probabilistic model

$$X|\phi \sim \mathrm{Cat}(\pi_{\theta}(\phi)) \qquad \text{or} \qquad X|\phi \sim \mathcal{N}(\mu_{\theta}(\phi), \sigma_{\theta}(\phi)^2)$$

Where does deep learning kick in?

Let ϕ be all side information available
e.g. deterministic *inputs/features*

Have neural networks predict parameters of our probabilistic model

$$X|\phi \sim \text{Cat}(\pi_\theta(\phi)) \qquad \text{or} \qquad X|\phi \sim \mathcal{N}(\mu_\theta(\phi), \sigma_\theta(\phi)^2)$$

and proceed to estimate parameters θ of the NNs

NN as efficient parametrisation

From the statistical point of view NNs do not generate data

- they parametrise distributions that *by assumption* govern data
- compact and efficient way to map from complex side information to parameter space

NN as efficient parametrisation

From the statistical point of view NNs do not generate data

- they parametrise distributions that *by assumption* govern data
- compact and efficient way to map from complex side information to parameter space

Prediction is done by a decision rule outside the statistical model

- e.g. beam search

Maximum likelihood estimation

Let $p(x|\theta)$ be the probability of an observation x
and θ refer to all of its parameters
e.g. parameters of NNs involved

Maximum likelihood estimation

Let $p(x|\theta)$ be the probability of an observation x
and θ refer to all of its parameters
e.g. parameters of NNs involved

Given a dataset $x^{(1)}, \ldots, x^{(N)}$ of i.i.d. observations, the likelihood function

$$\mathcal{L}(\theta|x^{(1:N)}) = \log \prod_{s=1}^{N} p(x^{(s)}|\theta)$$

Maximum likelihood estimation

Let $p(x|\theta)$ be the probability of an observation x
and θ refer to all of its parameters
e.g. parameters of NNs involved

Given a dataset $x^{(1)}, \ldots, x^{(N)}$ of i.i.d. observations, the likelihood function

$$\mathcal{L}(\theta|x^{(1:N)}) = \log \prod_{s=1}^{N} p(x^{(s)}|\theta) = \sum_{s=1}^{N} \log p(x^{(s)}|\theta)$$

Maximum likelihood estimation

Let $p(x|\theta)$ be the probability of an observation x
and θ refer to all of its parameters
e.g. parameters of NNs involved

Given a dataset $x^{(1)}, \ldots, x^{(N)}$ of i.i.d. observations, the likelihood function

$$\mathcal{L}(\theta|x^{(1:N)}) = \log \prod_{s=1}^{N} p(x^{(s)}|\theta) = \sum_{s=1}^{N} \log p(x^{(s)}|\theta)$$

quantifies the fitness of our model to data

MLE via gradient-based optimisation

If assessing the log-likelihood is **differentiable** and assessing it is **tractable**, then backpropagation can give us the gradient

$$\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) = \nabla_\theta \sum_{s=1}^{N} \log p(x^{(s)}|\theta)$$

MLE via gradient-based optimisation

If assessing the log-likelihood is **differentiable** and assessing it is **tractable**, then backpropagation can give us the gradient

$$\begin{aligned}\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) &= \nabla_\theta \sum_{s=1}^{N} \log p(x^{(s)}|\theta) \\ &= \sum_{s=1}^{N} \nabla_\theta \log p(x^{(s)}|\theta)\end{aligned}$$

MLE via gradient-based optimisation

If assessing the log-likelihood is **differentiable** and assessing it is **tractable**, then backpropagation can give us the gradient

$$\begin{aligned}\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) &= \nabla_\theta \sum_{s=1}^{N} \log p(x^{(s)}|\theta) \\ &= \sum_{s=1}^{N} \nabla_\theta \log p(x^{(s)}|\theta)\end{aligned}$$

and we can update θ in the direction

$$\gamma \nabla_\theta \mathcal{L}(\theta|x^{(1:N)})$$

to attain a local optimum of the likelihood function

Stochastic optimisation

We can also use a gradient estimate

$$\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) = \nabla_\theta \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \log p(x^{(S)}|\theta)\right]}_{\mathcal{L}(\theta|x^{(1:N)})}$$

Stochastic optimisation

We can also use a gradient estimate

$$\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) = \nabla_\theta \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \log p(x^{(S)}|\theta)\right]}_{\mathcal{L}(\theta|x^{(1:N)})}$$

$$= \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \nabla_\theta \log p(x^{(S)}|\theta)\right]}_{\text{expected gradient :)}}$$

Stochastic optimisation

We can also use a gradient estimate

$$\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) = \nabla_\theta \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \log p(x^{(S)}|\theta)\right]}_{\mathcal{L}(\theta|x^{(1:N)})}$$

$$= \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \nabla_\theta \log p(x^{(S)}|\theta)\right]}_{\text{expected gradient :)}}$$

$$\overset{\text{MC}}{\approx} \frac{1}{M}\sum_{m=1}^{M} N \nabla_\theta \log p(x^{(s_i)}|\theta)$$

$$S_i \sim \mathcal{U}(1..N)$$

Stochastic optimisation

We can also use a gradient estimate

$$\nabla_\theta \mathcal{L}(\theta|x^{(1:N)}) = \nabla_\theta \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \log p(x^{(S)}|\theta)\right]}_{\mathcal{L}(\theta|x^{(1:N)})}$$

$$= \underbrace{\mathbb{E}_{S\sim\mathcal{U}(1..N)}\left[N \nabla_\theta \log p(x^{(S)}|\theta)\right]}_{\text{expected gradient :)}}$$

$$\overset{\text{MC}}{\approx} \frac{1}{M}\sum_{m=1}^{M} N \nabla_\theta \log p(x^{(s_i)}|\theta)$$

$$S_i \sim \mathcal{U}(1..N)$$

and take steps in the direction

$$\gamma \frac{N}{M} \nabla_\theta \mathcal{L}(\theta|x^{(s_1:s_M)})$$

where $x^{(s_1)}, \ldots, x^{(s_M)}$ is a random mini-batch of size M

DL in NLP recipe

Maximum likelihood estimation

- tells you which loss to optimise
  (i.e. negative log-likelihood)

DL in NLP recipe

Maximum likelihood estimation

- tells you which loss to optimise
  (i.e. negative log-likelihood)

Automatic differentiation (*backprop*)

- chain rule of derivatives: "give me a tractable forward pass and I will give you gradients"

DL in NLP recipe

Maximum likelihood estimation

- tells you which loss to optimise (i.e. negative log-likelihood)

Automatic differentiation (*backprop*)

- chain rule of derivatives: "give me a tractable forward pass and I will give you gradients"

Stochastic optimisation powered by backprop

- general purpose gradient-based optimisers

Tractability is central

Likelihood gives us a differentiable objective to optimise for

- but we need to stick with tractable likelihood functions

When do we have intractable likelihood?

Unsupervised problems contain unobserved random variables

$$p_\theta(x, z) = \overbrace{p(z)}^{\text{latent variable model}} \underbrace{p_\theta(x|z)}_{\text{observation model}}$$

When do we have intractable likelihood?

Unsupervised problems contain unobserved random variables

$$p_\theta(x, z) = \overbrace{p(z)}^{\text{latent variable model}} \underbrace{p_\theta(x|z)}_{\text{observation model}}$$

thus assessing the marginal likelihood requires marginalisation of latent variables

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

Examples of latent variable models

Discrete latent variable, continuous observation

- too many forward passes

$$p_\theta(x) = \sum_{c=1}^{K} \mathrm{Cat}(c|\pi_1, \ldots, \pi_K) \underbrace{\mathcal{N}(x|\mu_\theta(c), \sigma_\theta(c)^2)}_{\text{forward pass}}$$

Examples of latent variable models

Discrete latent variable, continuous observation

- too many forward passes

$$p_\theta(x) = \sum_{c=1}^{K} \mathrm{Cat}(c|\pi_1, \ldots, \pi_K) \underbrace{\mathcal{N}(x|\mu_\theta(c), \sigma_\theta(c)^2)}_{\text{forward pass}}$$

Continuous latent variable, discrete observation

- infinitely many forward passes

$$p_\theta(x) = \int \mathcal{N}(z|0, I) \underbrace{\mathrm{Cat}(x|\pi_\theta(z))}_{\text{forward pass}} \, \mathrm{d}z$$

Deep generative models

Joint distribution with **deep observation model**

$$p_\theta(x, z) = \underbrace{p(z)}_{\text{prior}} \underbrace{p_\theta(x|z)}_{\text{likelihood}}$$

mapping from latent variable z to $p(x|z)$ is a NN with parameters θ

Deep generative models

Joint distribution with **deep observation model**

$$p_\theta(x, z) = \underbrace{p(z)}_{\text{prior}} \underbrace{p_\theta(x|z)}_{\text{likelihood}}$$

mapping from latent variable z to $p(x|z)$ is a NN with parameters θ

Marginal likelihood (or evidence)

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

intractable in general

Gradient

Exact gradient is intractable

$$\nabla_\theta \log p_\theta(x)$$

Gradient

Exact gradient is intractable

$$\nabla_\theta \log p_\theta(x) = \nabla_\theta \log \underbrace{\int p_\theta(x, z)\, \mathrm{d}z}_{\text{marginal}}$$

Gradient

Exact gradient is intractable

$$\begin{aligned}\nabla_\theta \log p_\theta(x) &= \nabla_\theta \log \underbrace{\int p_\theta(x,z)\,\mathrm{d}z}_{\text{marginal}} \\ &= \underbrace{\frac{1}{\int p_\theta(x,z)\,\mathrm{d}z} \int \nabla_\theta p_\theta(x,z)\,\mathrm{d}z}_{\text{chain rule}}\end{aligned}$$

Gradient

Exact gradient is intractable

$$\begin{aligned}
\nabla_\theta \log p_\theta(x) &= \nabla_\theta \log \underbrace{\int p_\theta(x,z)\,\mathrm{d}z}_{\text{marginal}} \\
&= \underbrace{\frac{1}{\int p_\theta(x,z)\,\mathrm{d}z} \int \nabla_\theta p_\theta(x,z)\,\mathrm{d}z}_{\text{chain rule}} \\
&= \frac{1}{p_\theta(x)} \int \underbrace{p_\theta(x,z) \nabla_\theta \log p_\theta(x,z)}_{\text{log-identity for derivatives}}\,\mathrm{d}z
\end{aligned}$$

Gradient

Exact gradient is intractable

$$
\begin{aligned}
\nabla_\theta \log p_\theta(x) &= \nabla_\theta \log \underbrace{\int p_\theta(x,z)\,\mathrm{d}z}_{\text{marginal}} \\
&= \underbrace{\frac{1}{\int p_\theta(x,z)\,\mathrm{d}z} \int \nabla_\theta p_\theta(x,z)\,\mathrm{d}z}_{\text{chain rule}} \\
&= \frac{1}{p_\theta(x)} \int \underbrace{p_\theta(x,z) \nabla_\theta \log p_\theta(x,z)}_{\text{log-identity for derivatives}}\,\mathrm{d}z \\
&= \int \underbrace{\frac{p_\theta(x,z)}{p_\theta(x)}}_{\text{posterior}} \nabla_\theta \log p_\theta(x,z)\,\mathrm{d}z
\end{aligned}
$$

Gradient

Exact gradient is intractable

$$
\begin{aligned}
\nabla_\theta \log p_\theta(x) &= \nabla_\theta \log \underbrace{\int p_\theta(x,z)\,\mathrm{d}z}_{\text{marginal}} \\
&= \underbrace{\frac{1}{\int p_\theta(x,z)\,\mathrm{d}z} \int \nabla_\theta p_\theta(x,z)\,\mathrm{d}z}_{\text{chain rule}} \\
&= \frac{1}{p_\theta(x)} \int \underbrace{p_\theta(x,z)\nabla_\theta \log p_\theta(x,z)}_{\text{log-identity for derivatives}}\,\mathrm{d}z \\
&= \int \underbrace{\frac{p_\theta(x,z)}{p_\theta(x)}}_{\text{posterior}} \nabla_\theta \log p_\theta(x,z)\,\mathrm{d}z \\
&= \int p_\theta(z|x)\nabla_\theta \log p_\theta(x,z)\,\mathrm{d}z
\end{aligned}
$$

Gradient

Exact gradient is intractable

$$
\begin{aligned}
\nabla_\theta \log p_\theta(x) &= \nabla_\theta \log \underbrace{\int p_\theta(x, z)\,\mathrm{d}z}_{\text{marginal}} \\
&= \underbrace{\frac{1}{\int p_\theta(x, z)\,\mathrm{d}z} \int \nabla_\theta p_\theta(x, z)\,\mathrm{d}z}_{\text{chain rule}} \\
&= \frac{1}{p_\theta(x)} \int \underbrace{p_\theta(x, z) \nabla_\theta \log p_\theta(x, z)}_{\text{log-identity for derivatives}}\,\mathrm{d}z \\
&= \int \underbrace{\frac{p_\theta(x, z)}{p_\theta(x)}}_{\text{posterior}} \nabla_\theta \log p_\theta(x, z)\,\mathrm{d}z \\
&= \int p_\theta(z|x) \nabla_\theta \log p_\theta(x, z)\,\mathrm{d}z \\
&= \underbrace{\mathbb{E}_{p_\theta(z|x)} \left[\nabla_\theta \log p_\theta(x, Z)\right]}_{\text{expected gradient :)}}
\end{aligned}
$$

Can we get an estimate?

$$\nabla_\theta \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}\left[\nabla_\theta \log p_\theta(x, Z)\right]$$

Can we get an estimate?

$$\begin{aligned}\nabla_\theta \log p_\theta(x) &= \mathbb{E}_{p_\theta(z|x)}\left[\nabla_\theta \log p_\theta(x, Z)\right] \\ &\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta \log p_\theta(x, z^{(k)}) \\ z^{(k)} &\sim p_\theta(Z|x)\end{aligned}$$

Can we get an estimate?

$$\begin{aligned}\nabla_\theta \log p_\theta(x) &= \mathbb{E}_{p_\theta(z|x)}\left[\nabla_\theta \log p_\theta(x, Z)\right] \\ &\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta \log p_\theta(x, z^{(k)}) \\ z^{(k)} &\sim p_\theta(Z|x)\end{aligned}$$

MC estimate of gradient requires sampling from posterior

$$p_\theta(z|x) = \frac{p(z)p_\theta(x|z)}{p_\theta(x)}$$

unavailable due to the intractability of the marginal

Summary

- We like probabilistic models because can make explicit modelling assumptions

Summary

- We like probabilistic models because can make explicit modelling assumptions
- We want complex observation models parameterised by NNs

Summary

- We like probabilistic models because can make explicit modelling assumptions
- We want complex observation models parameterised by NNs
- But we cannot use backprop for parameter estimation

Summary

- We like probabilistic models because can make explicit modelling assumptions
- We want complex observation models parameterised by NNs
- But we cannot use backprop for parameter estimation

We need approximate inference techniques!

Outline

1. Deep generative models
2. Variational inference
3. Variational auto-encoder

The Basic Problem

The marginal likelihood

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

is generally **intractable**, which prevents us from computing quantities that depend on the posterior $p(z|x)$

- e.g. gradients in MLE
- e.g. predictive distribution in Bayesian modelling

Strategy

Accept that $p(z|x)$ is not computable.

Strategy

Accept that $p(z|x)$ is not computable.

- approximate it by an auxiliary distribution $q(z|x)$ that is computable
- choose $q(z|x)$ as close as possible to $p(z|x)$ to obtain a faithful approximation

Evidence lowerbound

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

Evidence lowerbound

$$\begin{aligned}\log p(x) &= \log \int p(x, z)\mathrm{d}z \\ &= \log \int q(z|x)\frac{p(x, z)}{q(z|x)}\mathrm{d}z\end{aligned}$$

Evidence lowerbound

$$
\begin{aligned}
\log p(x) &= \log \int p(x, z)\mathrm{d}z \\
&= \log \int {\color{red} q(z|x)} \frac{p(x, z)}{{\color{red} q(z|x)}}\mathrm{d}z \\
&= \log \left(\mathbb{E}_{q(z|x)} \left[\frac{p(x, Z)}{q(Z|x)} \right] \right)
\end{aligned}
$$

Evidence lowerbound

$$
\begin{aligned}
\log p(x) &= \log \int p(x, z)\mathrm{d}z \\
&= \log \int {\color{red} q(z|x)} \frac{p(x, z)}{{\color{red} q(z|x)}} \mathrm{d}z \\
&= \log \left(\mathbb{E}_{q(z|x)} \left[\frac{p(x, Z)}{q(Z|x)} \right] \right) \\
&\geq \underbrace{\mathbb{E}_{q(z|x)} \left[\log \frac{p(x, Z)}{q(Z|x)} \right]}_{\text{ELBO}}
\end{aligned}
$$

Evidence lowerbound

$$
\begin{aligned}
\log p(x) &= \log \int p(x, z) \mathrm{d}z \\
&= \log \int q(z|x) \frac{p(x, z)}{q(z|x)} \mathrm{d}z \\
&= \log \left(\mathbb{E}_{q(z|x)} \left[\frac{p(x, Z)}{q(Z|x)} \right] \right) \\
&\geq \underbrace{\mathbb{E}_{q(z|x)} \left[\log \frac{p(x, Z)}{q(Z|x)} \right]}_{\text{ELBO}} \\
&= \mathbb{E}_{q(z|x)} \left[\log p(x, Z) \right] - \mathbb{E}_{q(z|x)} \left[\log q(Z) \right]
\end{aligned}
$$

Evidence lowerbound

$$
\begin{aligned}
\log p(x) &= \log \int p(x, z)\mathrm{d}z \\
&= \log \int q(z|x) \frac{p(x, z)}{q(z|x)} \mathrm{d}z \\
&= \log \left(\mathbb{E}_{q(z|x)} \left[\frac{p(x, Z)}{q(Z|x)} \right] \right) \\
&\geq \underbrace{\mathbb{E}_{q(z|x)} \left[\log \frac{p(x, Z)}{q(Z|x)} \right]}_{\text{ELBO}} \\
&= \mathbb{E}_{q(z|x)} \left[\log p(x, Z) \right] - \mathbb{E}_{q(z|x)} \left[\log q(Z) \right] \\
&= \mathbb{E}_{q(z|x)} \left[\log p(x, Z) \right] + \mathbb{H}\left(q(z|x) \right)
\end{aligned}
$$

An approximate posterior

$$\log p(x) \geq \underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{p(x, Z)}{q(Z|x)}\right]}_{\text{ELBO}}$$

An approximate posterior

$$\log p(x) \geq \underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{p(x, Z)}{q(Z|x)}\right]}_{\text{ELBO}}$$
$$= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)p(x)}{q(Z|x)}\right]$$

An approximate posterior

$$
\begin{aligned}
\log p(x) &\geq \underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{p(x, Z)}{q(Z|x)}\right]}_{\text{ELBO}} \\
&= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)p(x)}{q(Z|x)}\right] \\
&= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)}{q(Z|x)}\right] + \underbrace{\log p(x)}_{\text{constant}}
\end{aligned}
$$

An approximate posterior

$$
\begin{aligned}
\log p(x) &\geq \underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{p(x, Z)}{q(Z|x)}\right]}_{\text{ELBO}} \\
&= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)p(x)}{q(Z|x)}\right] \\
&= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)}{q(Z|x)}\right] + \underbrace{\log p(x)}_{\text{constant}} \\
&= -\underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{q(Z|x)}{p(Z|x)}\right]}_{\text{KL}(q(z|x)||p(z|x))} + \log p(x)
\end{aligned}
$$

An approximate posterior

$$\log p(x) \geq \underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{p(x, Z)}{q(Z|x)}\right]}_{\text{ELBO}}$$

$$= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)p(x)}{q(Z|x)}\right]$$

$$= \mathbb{E}_{q(z|x)}\left[\log \frac{p(Z|x)}{q(Z|x)}\right] + \underbrace{\log p(x)}_{\text{constant}}$$

$$= -\underbrace{\mathbb{E}_{q(z|x)}\left[\log \frac{q(Z|x)}{p(Z|x)}\right]}_{\mathrm{KL}(q(z|x)||p(z|x))} + \log p(x)$$

We have derived a lower bound on the log-evidence whose gap is exactly $\mathrm{KL}\left(q(z|x) \mid\mid p(z|x)\right)$.

Variational Inference

Objective

$$\max_{q(z|x)} \mathbb{E}\left[\log p(x, Z)\right] + \mathbb{H}\left(q(z|x)\right)$$

- The ELBO is a lower bound on $\log p(x)$

Blei et al. (2016)

Mean field assumption

Suppose we have N latent variables

- assume the posterior factorises as N independent terms
- each with an independent set of parameters

$$q(z_1, \ldots, q_N) = \underbrace{\prod_{i=1}^{N} q_{\lambda_i}(z_i)}_{\text{mean field}}$$

Amortised variational inference

Amortise the cost of inference using NNs

$$q(z_1, \ldots, q_N | x_1, \ldots, x_N) = \prod_{i=1}^{N} q_\lambda(z_i | x_i)$$

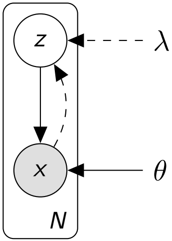
with a shared set of parameters

- e.g. $Z|x \sim \mathcal{N}(\underbrace{\mu_\lambda(x), \sigma_\lambda(x)^2}_{\text{inference network}})$

Outline

Variational auto-encoder

Generative model with NN likelihood



- complex (non-linear) observation model $p_\theta(x|z)$
- complex (non-linear) mapping from data to latent variables $q_\lambda(z|x)$

Jointly optimise generative model $p_\theta(x|z)$ and inference model $q_\lambda(z|x)$ under the same objective (ELBO)

Kingma and Welling (2013)

Objective

$$\log p_\theta(x) \geq \overbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x, Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)}^{\text{ELBO}}$$

Objective

$$\log p_\theta(x) \geq \overbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x, Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)}^{\text{ELBO}}$$

$$= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z) + \log p(Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)$$

Objective

$$
\begin{aligned}
\log p_\theta(x) &\geq \overbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x, Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)}^{\text{ELBO}} \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z) + \log p(Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right) \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \;||\; p(z)\right)
\end{aligned}
$$

Objective

$$
\begin{aligned}
\log p_\theta(x) &\geq \overbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x, Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)}^{\text{ELBO}} \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z) + \log p(Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right) \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)
\end{aligned}
$$

Parameter estimation

$$
\arg\max_{\theta,\lambda} \; \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)
$$

Objective

$$
\begin{aligned}
\log p_\theta(x) &\geq \overbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x, Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)}^{\text{ELBO}} \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z) + \log p(Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right) \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \;||\; p(z)\right)
\end{aligned}
$$

Parameter estimation

$$
\arg\max_{\theta,\lambda} \; \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \;||\; p(z)\right)
$$

- assume $\mathrm{KL}\left(q_\lambda(z|x) \;||\; p(z)\right)$ analytical
  true for exponential families

Objective

$$\log p_\theta(x) \geq \overbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x, Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)}^{\text{ELBO}}$$
$$= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z) + \log p(Z)\right] + \mathbb{H}\left(q_\lambda(z|x)\right)$$
$$= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$$

Parameter estimation

$$\arg\max_{\theta,\lambda} \; \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|Z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$$

- assume $\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$ analytical
  true for exponential families
- approximate $\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right]$ by sampling
  true because we design $q_\lambda(z|x)$ to be simple

Generative Network Gradient

$$\frac{\partial}{\partial \theta}\left(\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)}^{\text{constant wrt } \theta}\right)$$

Generative Network Gradient

$$\frac{\partial}{\partial \theta}\left(\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x) \,||\, p(z)\right)}^{\text{constant wrt } \theta}\right)$$

$$= \underbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\frac{\partial}{\partial \theta}\log p_\theta(x|z)\right]}_{\text{expected gradient :)}}$$

Generative Network Gradient

$$\frac{\partial}{\partial\theta}\left(\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x)\,||\,p(z)\right)}^{\text{constant wrt }\theta}\right)$$

$$= \underbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\frac{\partial}{\partial\theta}\log p_\theta(x|z)\right]}_{\text{expected gradient :)}}$$

$$\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K}\frac{\partial}{\partial\theta}\log p_\theta(x|z^{(k)})$$

$$z^{(k)} \sim q_\lambda(Z|x)$$

Generative Network Gradient

$$\frac{\partial}{\partial \theta}\left(\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x) \,||\, p(z)\right)}^{\text{constant wrt } \theta}\right)$$

$$= \underbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\frac{\partial}{\partial \theta}\log p_\theta(x|z)\right]}_{\text{expected gradient :)}}$$

$$\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K}\frac{\partial}{\partial \theta}\log p_\theta(x|z^{(k)})$$

$$z^{(k)} \sim q_\lambda(Z|x)$$

Note: $q_\lambda(z|x)$ does not depend on θ.

Inference Network Gradient

$$\frac{\partial}{\partial \lambda} \left(\mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z) \right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z) \right)}^{\text{analytical}} \right)$$

Inference Network Gradient

$$\frac{\partial}{\partial\lambda}\left(\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x)\ ||\ p(z)\right)}^{\text{analytical}}\right)$$

$$= \frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \underbrace{\frac{\partial}{\partial\lambda}\mathrm{KL}\left(q_\lambda(z|x)\ ||\ p(z)\right)}_{\text{analytical computation}}$$

Inference Network Gradient

$$\frac{\partial}{\partial \lambda}\left(\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \overbrace{\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)}^{\text{analytical}}\right)$$

$$= \frac{\partial}{\partial \lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \underbrace{\frac{\partial}{\partial \lambda}\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)}_{\text{analytical computation}}$$

The first term again requires approximation by sampling, but there is a problem

Inference Network Gradient

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} [\log p_\theta(x|z)]$$

Inference Network Gradient

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z)\right]$$
$$= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z) \mathrm{d}z$$

Inference Network Gradient

$$
\begin{aligned}
&\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \frac{\partial}{\partial\lambda}\int q_\lambda(z|x)\log p_\theta(x|z)\mathrm{d}z \\
&= \underbrace{\int \frac{\partial}{\partial\lambda}\left(q_\lambda(z|x)\right)\log p_\theta(x|z)\,\mathrm{d}z}_{\text{not an expectation}}
\end{aligned}
$$

Inference Network Gradient

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z)\right]$$
$$= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z) \mathrm{d}z$$
$$= \underbrace{\int \frac{\partial}{\partial \lambda} (q_\lambda(z|x)) \log p_\theta(x|z) \, \mathrm{d}z}_{\text{not an expectation}}$$

- MC estimator is non-differentiable: cannot sample first

Inference Network Gradient

$$
\begin{aligned}
&\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} [\log p_\theta(x|z)] \\
&= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z) \mathrm{d}z \\
&= \underbrace{\int \frac{\partial}{\partial \lambda} (q_\lambda(z|x)) \log p_\theta(x|z) \, \mathrm{d}z}_{\text{not an expectation}}
\end{aligned}
$$

- MC estimator is non-differentiable: cannot sample first
- Differentiating the expression does not yield an expectation: cannot approximate via MC

Score function estimator

We can again use the log identity for derivatives

$$
\begin{aligned}
&\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \frac{\partial}{\partial\lambda}\int q_\lambda(z|x)\log p_\theta(x|z)\mathrm{d}z \\
&= \underbrace{\int \frac{\partial}{\partial\lambda}(q_\lambda(z|x))\log p_\theta(x|z)\,\mathrm{d}z}_{\text{not an expectation}}
\end{aligned}
$$

Score function estimator

We can again use the log identity for derivatives

$$
\begin{aligned}
&\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \frac{\partial}{\partial\lambda}\int q_\lambda(z|x)\log p_\theta(x|z)\mathrm{d}z \\
&= \underbrace{\int \frac{\partial}{\partial\lambda}(q_\lambda(z|x))\log p_\theta(x|z)\,\mathrm{d}z}_{\text{not an expectation}} \\
&= \int q_\lambda(z|x)\frac{\partial}{\partial\lambda}(\log q_\lambda(z|x))\log p_\theta(x|z)\,\mathrm{d}z
\end{aligned}
$$

Score function estimator

We can again use the log identity for derivatives

$$
\begin{aligned}
&\frac{\partial}{\partial \lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \frac{\partial}{\partial \lambda}\int q_\lambda(z|x)\log p_\theta(x|z)\mathrm{d}z \\
&= \underbrace{\int \frac{\partial}{\partial \lambda}(q_\lambda(z|x))\log p_\theta(x|z)\,\mathrm{d}z}_{\text{not an expectation}} \\
&= \int q_\lambda(z|x)\frac{\partial}{\partial \lambda}(\log q_\lambda(z|x))\log p_\theta(x|z)\,\mathrm{d}z \\
&= \underbrace{\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\frac{\partial}{\partial \lambda}\log q_\lambda(z|x)\right]}_{\text{expected gradient :)}}
\end{aligned}
$$

Score function estimator: high variance

We can now build an MC estimator

$$\begin{aligned} &\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z) \right] \\ &= \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z) \frac{\partial}{\partial \lambda} \log q_\lambda(z|x) \right] \end{aligned}$$

Score function estimator: high variance

We can now build an MC estimator

$$
\begin{aligned}
&\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z)\right] \\
&= \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z) \frac{\partial}{\partial \lambda} \log q_\lambda(z|x)\right] \\
&\overset{\text{MC}}{\approx} \frac{1}{K} \sum_{k=1}^{K} \color{red}{\log p_\theta(x|z^{(k)})} \frac{\partial}{\partial \lambda} \log q_\lambda(z^{(k)}|x) \\
&z^{(k)} \sim q_\lambda(Z|x)
\end{aligned}
$$

Score function estimator: high variance

We can now build an MC estimator

$$\begin{aligned}
&\frac{\partial}{\partial \lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\frac{\partial}{\partial \lambda}\log q_\lambda(z|x)\right] \\
&\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K} \log p_\theta(x|z^{(k)})\frac{\partial}{\partial \lambda}\log q_\lambda(z^{(k)}|x) \\
&z^{(k)} \sim q_\lambda(Z|x)
\end{aligned}$$

but

- magnitude of $\log p_\theta(x|z)$ varies widely

Score function estimator: high variance

We can now build an MC estimator

$$\begin{aligned}
&\frac{\partial}{\partial \lambda} \mathbb{E}_{q_\lambda(z|x)} [\log p_\theta(x|z)] \\
&= \mathbb{E}_{q_\lambda(z|x)} \left[\log p_\theta(x|z) \frac{\partial}{\partial \lambda} \log q_\lambda(z|x) \right] \\
&\overset{\text{MC}}{\approx} \frac{1}{K} \sum_{k=1}^{K} \log p_\theta(x|z^{(k)}) \frac{\partial}{\partial \lambda} \log q_\lambda(z^{(k)}|x) \\
&z^{(k)} \sim q_\lambda(Z|x)
\end{aligned}$$

but

- magnitude of $\log p_\theta(x|z)$ varies widely
- model likelihood does not contribute to direction of gradient

Score function estimator: high variance

We can now build an MC estimator

$$\begin{aligned}
&\frac{\partial}{\partial \lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] \\
&= \mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\frac{\partial}{\partial \lambda}\log q_\lambda(z|x)\right] \\
&\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K} \color{red}{\log p_\theta(x|z^{(k)})}\frac{\partial}{\partial \lambda}\log q_\lambda(z^{(k)}|x) \\
&z^{(k)} \sim q_\lambda(Z|x)
\end{aligned}$$

but

- magnitude of $\log p_\theta(x|z)$ varies widely
- model likelihood does not contribute to direction of gradient
- too much variance to be useful

When variance is high we can

- sample more

When variance is high we can

- sample more
  won't scale

When variance is high we can

- sample more
  won't scale
- use variance reduction techniques (e.g. baselines and control variates)

When variance is high we can

- sample more
  won't scale
- use variance reduction techniques (e.g. baselines and control variates)
  excellent idea, but not just yet

When variance is high we can

- sample more
  won't scale
- use variance reduction techniques (e.g. baselines and control variates)
  excellent idea, but not just yet
- stare at this $\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right]$

When variance is high we can

- sample more
  won't scale
- use variance reduction techniques (e.g. baselines and control variates)
  excellent idea, but not just yet
- stare at this $\frac{\partial}{\partial\lambda}\mathbb{E}_{q_\lambda(z|x)}[\log p_\theta(x|z)]$
  until we find a way to rewrite the expectation in terms of a density that **does not depend on** λ

Reparametrisation

Find a transformation $h : z \mapsto \epsilon$ that expresses z through a random variable ϵ such that $q(\epsilon)$ does not depend on λ

(Kingma and Welling, 2013; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014)

Reparametrisation

Find a transformation $h : z \mapsto \epsilon$ that expresses z through a random variable ϵ such that $q(\epsilon)$ does not depend on λ

- $h(z, \lambda)$ needs to be invertible

(Kingma and Welling, 2013; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014)

Reparametrisation

Find a transformation $h : z \mapsto \epsilon$ that expresses z through a random variable ϵ such that $q(\epsilon)$ does not depend on λ

- $h(z, \lambda)$ needs to be invertible
- $h(z, \lambda)$ needs to be differentiable

(Kingma and Welling, 2013; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014)

Reparametrisation

Find a transformation $h : z \mapsto \epsilon$ that expresses z through a random variable ϵ such that $q(\epsilon)$ does not depend on λ

- $h(z, \lambda)$ needs to be invertible
- $h(z, \lambda)$ needs to be differentiable

Invertibility implies

- $h(z, \lambda) = \epsilon$
- $h^{-1}(\epsilon, \lambda) = z$

(Kingma and Welling, 2013; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014)

Gaussian Transformation

If $Z \sim \mathcal{N}(\mu_\lambda(x), \sigma_\lambda(x)^2)$ then

$$h(z, \lambda) = \frac{z - \mu_\lambda(x)}{\sigma_\lambda(x)} = \epsilon \sim \mathcal{N}(0, \mathrm{I})$$
$$h^{-1}(\epsilon, \lambda) = \mu_\lambda(x) + \sigma_\lambda(x) \odot \epsilon \quad \epsilon \sim \mathcal{N}(0, \mathrm{I})$$

Inference Network – Reparametrised Gradient

$$= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z) \, \mathrm{d}z$$

Inference Network – Reparametrised Gradient

$$= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z)\,\mathrm{d}z$$

$$= \frac{\partial}{\partial \lambda} \int q(\epsilon) \log p_\theta(x|\overbrace{h^{-1}(\epsilon, \lambda)}^{=z})\,\mathrm{d}\epsilon$$

Inference Network – Reparametrised Gradient

$$
\begin{aligned}
&= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z) \, \mathrm{d}z \\
&= \frac{\partial}{\partial \lambda} \int q(\epsilon) \log p_\theta(x| \overbrace{h^{-1}(\epsilon, \lambda)}^{=z}) \, \mathrm{d}\epsilon \\
&= \int q(\epsilon) \frac{\partial}{\partial \lambda} \left[\log p_\theta(x| \overbrace{h^{-1}(\epsilon, \lambda)}^{=z}) \right] \mathrm{d}\epsilon
\end{aligned}
$$

Inference Network – Reparametrised Gradient

$$
\begin{aligned}
&= \frac{\partial}{\partial \lambda} \int q_\lambda(z|x) \log p_\theta(x|z)\, \mathrm{d}z \\
&= \frac{\partial}{\partial \lambda} \int q(\epsilon) \log p_\theta(x|\overbrace{h^{-1}(\epsilon, \lambda)}^{=z})\, \mathrm{d}\epsilon \\
&= \int q(\epsilon) \frac{\partial}{\partial \lambda} \left[\log p_\theta(x|\overbrace{h^{-1}(\epsilon, \lambda)}^{=z}) \right] \mathrm{d}\epsilon \\
&= \underbrace{\mathbb{E}_{q(\epsilon)} \left[\frac{\partial}{\partial \lambda} \log p_\theta(x|h^{-1}(\epsilon, \lambda)) \right] \mathrm{d}\epsilon}_{\text{expected gradient :D}}
\end{aligned}
$$

Reparametrised gradient estimate

$$= \underbrace{\mathbb{E}_{q(\epsilon)}\left[\frac{\partial}{\partial\lambda}\log p_\theta(x|h^{-1}(\epsilon,\lambda))\right]\mathrm{d}\epsilon}_{\text{expected gradient :D}}$$

Reparametrised gradient estimate

$$= \underbrace{\mathbb{E}_{q(\epsilon)}\left[\frac{\partial}{\partial \lambda} \log p_\theta(x|h^{-1}(\epsilon, \lambda))\right] \mathrm{d}\epsilon}_{\text{expected gradient :D}}$$

$$= \mathbb{E}_{q(\epsilon)}\left[\underbrace{\frac{\partial}{\partial z} \log p_\theta(x|\overbrace{h^{-1}(\epsilon, \lambda)}^{=z}) \times \frac{\partial}{\partial \lambda} h^{-1}(\epsilon, \lambda)}_{\text{chain rule}}\right]$$

Reparametrised gradient estimate

$$= \underbrace{\mathbb{E}_{q(\epsilon)}\left[\frac{\partial}{\partial \lambda} \log p_\theta(x|h^{-1}(\epsilon, \lambda))\right] \mathrm{d}\epsilon}_{\text{expected gradient :D}}$$

$$= \mathbb{E}_{q(\epsilon)}\left[\underbrace{\frac{\partial}{\partial z} \log p_\theta(x|\overbrace{h^{-1}(\epsilon, \lambda)}^{=z}) \times \frac{\partial}{\partial \lambda} h^{-1}(\epsilon, \lambda)}_{\text{chain rule}}\right]$$

$$\overset{\text{MC}}{\approx} \frac{1}{K}\sum_{k=1}^{K} \underbrace{\frac{\partial}{\partial z} \log p_\theta(x|\overbrace{h^{-1}(\epsilon^{(k)}, \lambda)}^{=z}) \times \frac{\partial}{\partial \lambda} h^{-1}(\epsilon^{(k)}, \lambda)}_{\text{backprop's job}}$$

$$\epsilon^{(k)} \sim q(\epsilon)$$

Note that both models contribute with gradients

Gaussian KL

ELBO

$$\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$$

Gaussian KL

ELBO

$$\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$$

Analytical computation of $-\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$:

$$\frac{1}{2}\sum_{i=1}^{d}\left(1 + \log\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right)$$

Gaussian KL

ELBO

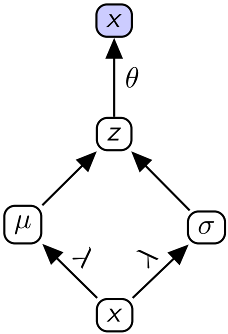
$$\mathbb{E}_{q_\lambda(z|x)}\left[\log p_\theta(x|z)\right] - \mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$$

Analytical computation of $-\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$:

$$\frac{1}{2}\sum_{i=1}^{d}\left(1 + \log\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right)$$

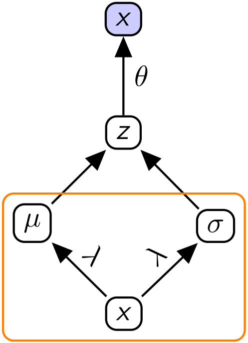
Thus backprop will compute $-\frac{\partial}{\partial\lambda}\mathrm{KL}\left(q_\lambda(z|x) \mid\mid p(z)\right)$ for us

Computation Graph



Computation Graph

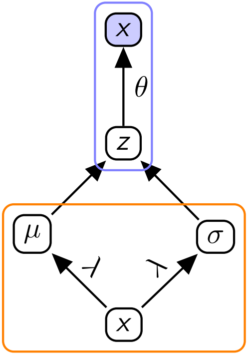
inference model



Computation Graph

generative model

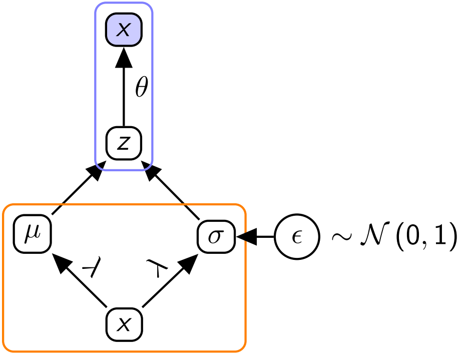
inference model



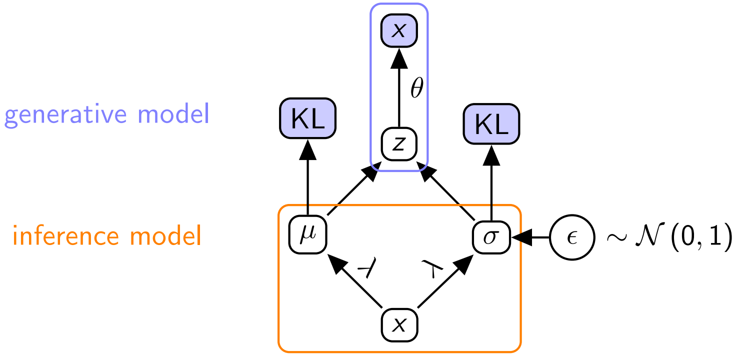
Computation Graph

generative model

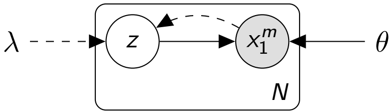
inference model



Computation Graph



Example



Generative model

- $Z \sim \mathcal{N}(0, I)$
- $X_i | z, x_{<i} \sim \text{Cat}(f_\theta(z, x_{<i}))$

Inference model

- $Z \sim \mathcal{N}(\mu_\lambda(x_1^m), \sigma_\lambda(x_1^m)^2)$

Bowman et al. (2016)

VAEs – Summary

Advantages

- Backprop training
- Easy to implement
- Posterior inference possible
- One objective for both NNs

VAEs – Summary

Advantages

- Backprop training
- Easy to implement
- Posterior inference possible
- One objective for both NNs

Drawbacks

- Discrete latent variables are difficult
- Optimisation may be difficult with several latent variables
- Location-scale families only
  but see Ruiz et al. (2016) and Kucukelbir et al. (2017)

Summary

Deep learning in NLP

- task-driven feature extraction
- models with more realistic assumptions

Probabilistic modelling

- better (or at least more explicit) statistical assumptions
- compact models
- semi-supervised learning

Literature I

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2014. URL http://arxiv.org/abs/1409.0473.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. 01 2016. URL https://arxiv.org/abs/1601.00670.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/K16-1002.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. 2013. URL http://arxiv.org/abs/1312.6114.

Literature II

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M. Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18(14):1–45, 2017. URL http://jmlr.org/papers/v18/16-107.html.

Danilo J. Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286, 2014. URL http://jmlr.org/proceedings/papers/v32/rezende14.pdf.

Francisco R Ruiz, Michalis Titsias RC AUEB, and David Blei. The generalized reparameterization gradient. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *NIPS*, pages 460–468. 2016. URL http://papers.nips.cc/paper/6328-the-generalized-reparameterization-gradient.pdf.

Literature III

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany, August 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P16-1009`.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In Tony Jebara and Eric P. Xing, editors, *ICML*, pages 1971–1979, 2014. URL `http://jmlr.org/proceedings/papers/v32/titsias14.pdf`.