Deep Generative Models: Continuous Latent Variables

Bryan Eikema and Wilker Aziz

https://vitutorial.github.io/tour/ua2020

University of Amsterdam
Institute for Logic, Language and Computation

gourmet

Generative Model with NN Likelihood

Goal

Define model $p(x, z|\theta) = p(x|z, \theta)p(z)$ where the likelihood $p(x|z, \theta)$ is given by a neural network.

We fix $p(z)$ for simplicity.

Example: Language Model

A deterministic language model is **one** distribution over observations:

$$p(x|\theta) = \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)$$

Every sentence gets mapped from the same conditioning context, namely, the beginning of sequence symbol.

Example: Language Model (cont.)

With latent variables we can model the data as a draw from a complex marginal, which mixes conditionals from different points in space

$$p(x|\theta) = \int p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)\mathrm{d}z$$

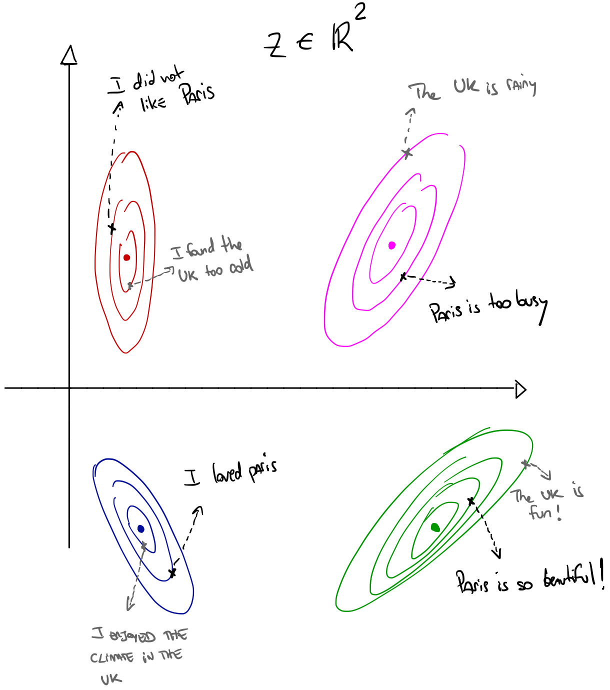
Example: Language Model (cont.)

With latent variables we can model the data as a draw from a complex marginal, which mixes conditionals from different points in space

$$p(x|\theta) = \int p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)\mathrm{d}z$$

Good training can lead to considerable amount of structure in the posterior

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


$z \in \mathbb{R}^2$
I did not like Paris
The UK is rainy
I found the UK too cold
Paris is too busy
I loved paris
The UK is fun!
Paris is so beautiful!
I ENJOYED THE CLIMATE IN THE UK

Example: Language Model (architecture)

Generative model:

$$Z \sim \mathcal{N}(0, I)$$
$$X_i | z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

Example: Language Model (architecture)

Generative model:

$$Z \sim \mathcal{N}(0, I)$$
$$X_i | z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

Example architecture:

$$h_0 = \tanh\left(W^{(\mathrm{init})} z + b^{(\mathrm{init})}\right)$$

Example: Language Model (architecture)

Generative model:

$$Z \sim \mathcal{N}(0, I)$$
$$X_i | z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

Example architecture:

$$h_0 = \tanh\left(W^{(\mathrm{init})} z + b^{(\mathrm{init})}\right)$$
$$h_i = \mathrm{rnn}(h_{i-1}, E_{x_{i-1}}; \theta_{\mathrm{rnn}})$$

Example: Language Model (architecture)

Generative model:

$$Z \sim \mathcal{N}(0, I)$$
$$X_i | z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

Example architecture:

$$h_0 = \tanh\left(W^{(\mathrm{init})} z + b^{(\mathrm{init})}\right)$$
$$h_i = \mathrm{rnn}(h_{i-1}, E_{x_{i-1}}; \theta_{\mathrm{rnn}})$$
$$f(z, x_{<i}) = \mathrm{softmax}(W^{(\mathrm{out})} h_i + b^{(\mathrm{out})})$$

Example: Language Model (architecture)

Generative model:

$$Z \sim \mathcal{N}(0, I)$$
$$X_i | z, x_{<i} \sim \mathrm{Cat}(f(z, x_{<i}; \theta))$$

Example architecture:

$$h_0 = \tanh\left(W^{(\mathrm{init})} z + b^{(\mathrm{init})}\right)$$
$$h_i = \mathrm{rnn}(h_{i-1}, E_{x_{i-1}}; \theta_{\mathrm{rnn}})$$
$$f(z, x_{<i}) = \mathrm{softmax}(W^{(\mathrm{out})} h_i + b^{(\mathrm{out})})$$
$$\theta = \theta_{\mathrm{rnn}} \cup \left\{W^{(\mathrm{init})}, b^{(\mathrm{init})}, W^{(\mathrm{out})}, b^{(\mathrm{out})}\right\}$$

Generative Model with NN Likelihood

Goal

Define model $p(x, z|\theta) = p(x|z, \theta)p(z)$ where the likelihood $p(x|z, \theta)$ is given by a neural network. (We fix $p(z)$ for simplicity.)

Problem

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

Solution: Variational Inference

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

Solution: Variational Inference

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

Solution: Variational Inference

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

Solution: Variational Inference

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

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

Solution: Variational Inference

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

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

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

Solution: Variational Inference

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

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

- assume $\mathrm{KL}\left(q(z|x,\lambda) \,||\, p(z)\right)$ analytical
  true for exponential families
- approximate $\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right]$ by sampling
  feasible because $q(z|x,\lambda)$ is simple

Generator Network Gradient

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

Generator Network Gradient

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

Generator Network Gradient

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

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

$$\overset{\mathrm{MC}}{\approx} \frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\theta}\log p(x|z_i,\theta)$$

where $z_i \sim q(z|x,\lambda)$

Generator Network Gradient

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

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

$$\overset{\mathrm{MC}}{\approx} \frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\theta}\log p(x|z_i,\theta)$$

where $z_i \sim q(z|x,\lambda)$

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

Inference Network Gradient

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

Inference Network Gradient

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

Inference Network Gradient

$$
\begin{aligned}
&\frac{\partial}{\partial \lambda}\left[\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \mathsf{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)\right] \\
=&\frac{\partial}{\partial \lambda}\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \underbrace{\frac{\partial}{\partial \lambda}\mathsf{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)}_{\text{analytical computation}}
\end{aligned}
$$

The first term again requires approximation by sampling

Inference Network Gradient

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q(z|x,\lambda)} \left[\log p(x|z,\theta) \right]$$

Inference Network Gradient

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q(z|x,\lambda)} [\log p(x|z,\theta)]$$
$$= \frac{\partial}{\partial \lambda} \int q(z|x,\lambda) \log p(x|z,\theta) \mathrm{d}z$$

Inference Network Gradient

$$
\begin{aligned}
&\frac{\partial}{\partial \lambda} \mathbb{E}_{q(z|x,\lambda)} [\log p(x|z,\theta)] \\
&= \frac{\partial}{\partial \lambda} \int q(z|x,\lambda) \log p(x|z,\theta) \mathrm{d}z \\
&= \int \frac{\partial}{\partial \lambda} q(z|x,\lambda) \log p(x|z,\theta) \mathrm{d}z
\end{aligned}
$$

Inference Network Gradient

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

Not an expected gradient!

Score function estimator?

Can we apply the log-derivative trick?

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

Score function estimator?

Can we apply the log-derivative trick?

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

Score function estimator?

Can we apply the log-derivative trick?

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

Yes, it's a general result!

What about variance?

The learning signal can only scale the gradient:

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

What about variance?

The learning signal can only scale the gradient:

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

Can we do better?

Inference Network Gradient

Problem

We need to re-express the gradient, but the measure of integration depends on λ

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q(z|x,\lambda)} \left[\log p(x|z,\theta) \right]$$

Inference Network Gradient

Problem

We need to re-express the gradient, but the measure of integration depends on λ

$$\frac{\partial}{\partial \lambda} \mathbb{E}_{q(z|x,\lambda)} \left[\log p(x|z,\theta) \right]$$

What if we could re-express $q(z|x, \lambda)$ in terms of some other distribution that does not depend on λ?

Inference Network Gradient

Reparametrisation trick

Find a transformation $h : z \mapsto \epsilon$ such that ϵ does not depend on λ.

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

Inference Network Gradient

Reparametrisation trick

Find a transformation $h : z \mapsto \epsilon$ such that ϵ does not depend on λ.

- $h(z, \lambda)$ needs to be invertible
- $h(z, \lambda)$ needs to be differentiable
- $h(z, \lambda) = \epsilon$
- $h^{-1}(\epsilon, \lambda) = z$

Gaussian Transformation

Affine property

$$Az + b \sim \mathcal{N}\left(\mu + b, A\Sigma A^T\right) \text{ for } z \sim \mathcal{N}\left(\mu, \Sigma\right)$$

Gaussian Transformation

Affine property

$$Az + b \sim \mathcal{N}\left(\mu + b, A\Sigma A^T\right) \text{ for } z \sim \mathcal{N}\left(\mu, \Sigma\right)$$

Special case

$$Az + b \sim \mathcal{N}\left(b, AA^T\right) \text{ for } z \sim \mathcal{N}\left(0, \mathrm{I}\right)$$

Gaussian Transformation

Let an inference network compute

$$u = \mu(x; \lambda) \quad s = \sigma(x; \lambda)$$

for a posterior $Z \sim \mathcal{N}(u, s^2)$, then we have:

Gaussian Transformation

Let an inference network compute

$$u = \mu(x; \lambda) \quad s \; = \sigma(x; \lambda)$$

for a posterior $Z \sim \mathcal{N}(u, s^2)$, then we have:

$$h(z, \lambda; x) = \frac{z - \mu(x; \lambda)}{\sigma(x; \lambda)} = \epsilon \sim \mathcal{N}(0, \mathrm{I})$$

Gaussian Transformation

Let an inference network compute

$$u = \mu(x;\lambda) \quad s = \sigma(x;\lambda)$$

for a posterior $Z \sim \mathcal{N}(u, s^2)$, then we have:

$$h(z,\lambda;x) = \frac{z-\mu(x;\lambda)}{\sigma(x;\lambda)} = \epsilon \sim \mathcal{N}(0,\mathrm{I})$$

and conversely, for $\epsilon \sim \mathcal{N}(0, I)$, we have:

$$h^{-1}(\epsilon,\lambda;x) = \mu(x;\lambda) + \sigma(x;\lambda) \odot \epsilon = z \sim \mathcal{N}(u, s^2)$$

Inference Network Gradient

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

Inference Network Gradient

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

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

Inference Network Gradient

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

Inference Network Gradient

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

Inference Network Gradient

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

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\lambda}\log p(x|\overbrace{h^{-1}(\epsilon_i,\lambda)}^{=z},\theta)$$

where $\epsilon_i \sim q(\epsilon)$

Inference Network Gradient

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

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\frac{\partial}{\partial\lambda}\log p(x|\overbrace{h^{-1}(\epsilon_i,\lambda)}^{=z},\theta)$$

where $\epsilon_i \sim q(\epsilon)$

$$\overset{\text{MC}}{\approx}\frac{1}{S}\sum_{i=1}^{S}\underbrace{\frac{\partial}{\partial z}\log p(x|\overbrace{h^{-1}(\epsilon_i,\lambda)}^{=z},\theta)\times\frac{\partial}{\partial\lambda}h^{-1}(\epsilon_i,\lambda)}_{\text{chain rule}}$$

Derivatives of Gaussian transformation

Recall:

$$h^{-1}(\epsilon, \lambda) = \mu(x, \lambda) + \sigma(x, \lambda) \odot \epsilon \ .$$

We get two gradient paths!

Derivatives of Gaussian transformation

Recall:

$$h^{-1}(\epsilon, \lambda) = \mu(x, \lambda) + \sigma(x, \lambda) \odot \epsilon \ .$$

We get two gradient paths!

- one is deterministic
  $\frac{\partial h^{-1}(\epsilon,\lambda)}{\partial \mu(x,\lambda)} = \frac{\partial}{\partial \mu(x,\lambda)}[\mu(x, \lambda) + \sigma(x, \lambda) \odot \epsilon] = 1$

Derivatives of Gaussian transformation

Recall:

$$h^{-1}(\epsilon, \lambda) = \mu(x, \lambda) + \sigma(x, \lambda) \odot \epsilon \ .$$

We get two gradient paths!

- one is deterministic
  $\frac{\partial h^{-1}(\epsilon,\lambda)}{\partial \mu(x,\lambda)} = \frac{\partial}{\partial \mu(x,\lambda)}[\mu(x, \lambda) + \sigma(x, \lambda) \odot \epsilon] = 1$
- the other is stochastic
  $\frac{\partial h^{-1}(\epsilon,\lambda)}{\partial \sigma(x,\lambda)} = \frac{\partial}{\partial \sigma(x,\lambda)}[\mu(x, \lambda) + \sigma(x, \lambda) \odot \epsilon] = \epsilon$

Gaussian KL

ELBO

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

Gaussian KL

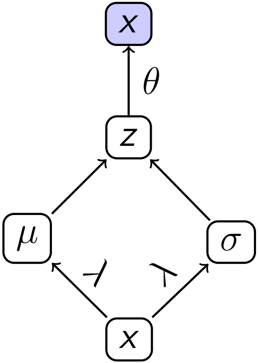
ELBO

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

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

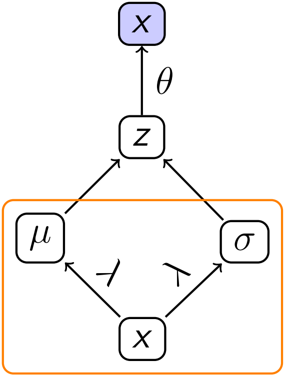
$$\frac{1}{2}\sum_{i=1}^{N}\left(1 + \log\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right)$$

Computation Graph



Computation Graph

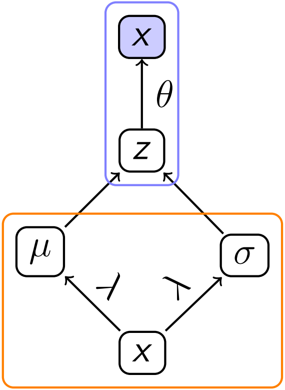
inference model



Computation Graph

generation model

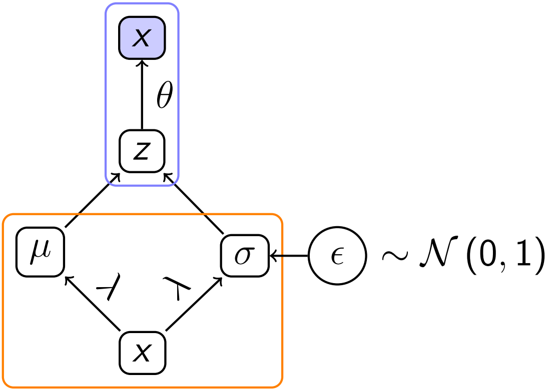
inference model



Computation Graph

generation model

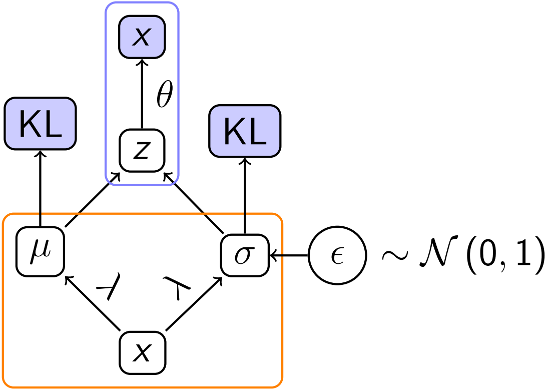
inference model



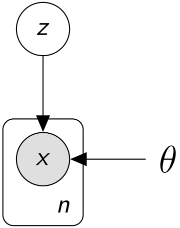
Computation Graph

generation model

inference model



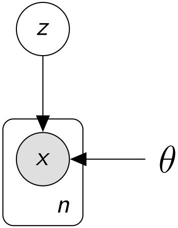
Example: Unigram Document Model



Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Example: Unigram Document Model

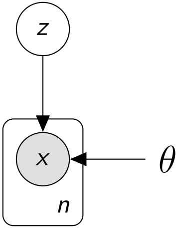


Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \text{Cat}(f(z; \theta))$

Designing $f(z, \theta)$

Example: Unigram Document Model



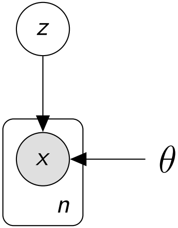
Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Designing $f(z, \theta)$

$$h = \tanh(W_1 z + b_1)$$

Example: Unigram Document Model



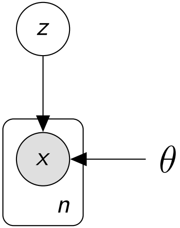
Generative story

- Draw a document embedding $Z \sim \mathcal{N}(0, I)$
- Draw n words $X_i|z \sim \text{Cat}(f(z;\theta))$

Designing $f(z, \theta)$

$$h = \tanh(W_1 z + b_1)$$
$$f(z,\theta) = \text{softmax}(W_2 h + b_2)$$

Example: Unigram Document Model



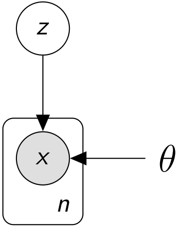
Generative story

- Draw a document embedding $Z \sim \mathcal{N}(0, I)$
- Draw n words $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Designing $f(z, \theta)$

$$\begin{aligned} h &= \tanh(W_1 z + b_1) \\ f(z, \theta) &= \mathrm{softmax}(W_2 h + b_2) \\ \theta &= \{W_1, b_1, W_2, b_2\} \end{aligned}$$

Example: Unigram Document Model

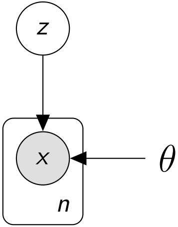


Likelihood

Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Example: Unigram Document Model



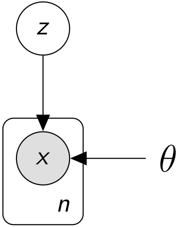
Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \text{Cat}(f(z; \theta))$

Likelihood

$$p(x|z, \theta) = \prod_{i=1}^{n} p(x_i|z, \theta)$$

Example: Unigram Document Model



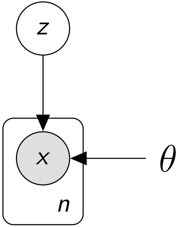
Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Likelihood

$$p(x|z, \theta) = \prod_{i=1}^{n} p(x_i|z, \theta) = \prod_{i=1}^{n} \mathrm{Cat}(x_i| \underbrace{f(z; \theta)}_{=\psi})$$

Example: Unigram Document Model



Generative story

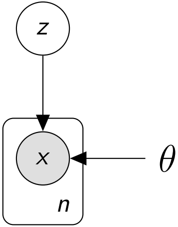
- Draw a document embedding $Z \sim \mathcal{N}(0, I)$
- Draw n words $X_i | z \sim \text{Cat}(f(z; \theta))$

Likelihood

$$p(x|z, \theta) = \prod_{i=1}^{n} p(x_i|z, \theta) = \prod_{i=1}^{n} \text{Cat}(x_i| \underbrace{f(z; \theta)}_{=\psi})$$

$$= \prod_{i=1}^{n} \psi_{x_i}$$

Example: Unigram Document Model

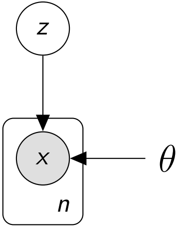


Marginal

Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \text{Cat}(f(z; \theta))$

Example: Unigram Document Model



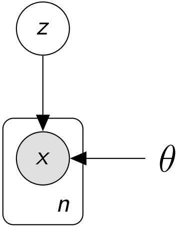
Marginal

Generative story

- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

$$p(x|\theta) = \int p(z) \prod_{i=1}^{n} p(x_i | z, \theta) \, \mathrm{d}z$$

Example: Unigram Document Model



Generative story

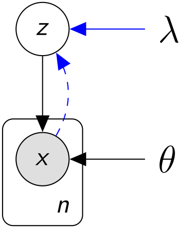
- Draw a document embedding
  $Z \sim \mathcal{N}(0, I)$
- Draw n words
  $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Marginal

$$p(x|\theta) = \int p(z) \prod_{i=1}^{n} p(x_i|z, \theta)\, \mathrm{d}z$$

$$= \int \mathcal{N}(z|0, I) \prod_{i=1}^{n} \mathrm{Cat}(x_i|f(z; \theta))\, \mathrm{d}z$$

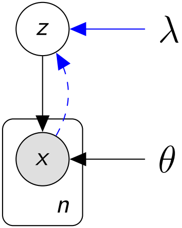
Example: Unigram Document Model



Inference model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

Example: Unigram Document Model

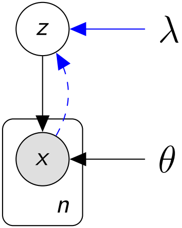


Inference model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

Designing the *inference network*

Example: Unigram Document Model



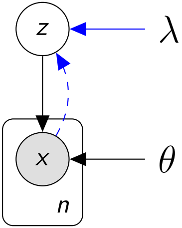
Inference model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

Example: Unigram Document Model



Inference model

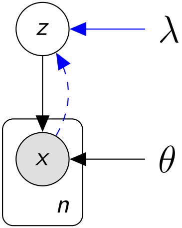
- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

$$h = \tanh(M_1 s + c_1)$$

Example: Unigram Document Model



Inference model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

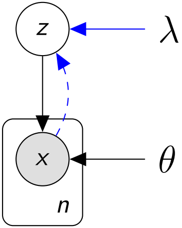
Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

$$\mu(x;\lambda) = M_2 h + c_2$$

$$h = \tanh(M_1 s + c_1)$$

Example: Unigram Document Model



Inference model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

Designing the *inference network*

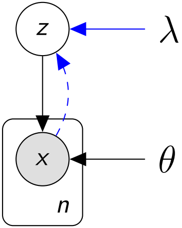
$$s = \sum_{i=1}^{n} E_{x_i}$$

$$h = \tanh(M_1 s + c_1)$$

$$\mu(x;\lambda) = M_2 h + c_2$$

$$\sigma(x;\lambda) = \text{softplus}(M_3 h + c_3)$$

Example: Unigram Document Model



Inference model

- $Z|x \sim \mathcal{N}(\mu(x;\lambda), \sigma(x;\lambda)^2)$

Designing the *inference network*

$$s = \sum_{i=1}^{n} E_{x_i}$$

$$h = \tanh(M_1 s + c_1)$$

$$\mu(x;\lambda) = M_2 h + c_2$$

$$\sigma(x;\lambda) = \text{softplus}(M_3 h + c_3)$$

$$\lambda = \{E, M_1^3, c_1^3\}$$

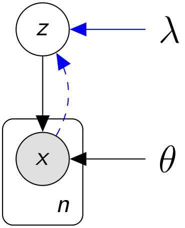
Example: Unigram Document Model

Generative Model

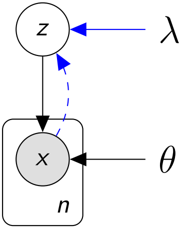
- Prior: $Z \sim \mathcal{N}(0, I)$
- Likelihood: $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Inference Model

- $Z | x \sim \mathcal{N}(\mu(x; \lambda), \sigma(x; \lambda)^2)$



Example: Unigram Document Model



Generative Model

- Prior: $Z \sim \mathcal{N}(0, I)$
- Likelihood: $X_i | z \sim \mathrm{Cat}(f(z; \theta))$

Inference Model

- $Z | x \sim \mathcal{N}(\mu(x; \lambda), \sigma(x; \lambda)^2)$

ELBO

$$\begin{aligned} \log p(x|\theta) \geq\ & \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)} \left[\textstyle\sum_{i=1}^{n} \log \psi_{x_i} \right] \\ & - \mathrm{KL}\left(\mathcal{N}(u, s^2) \,||\, \mathcal{N}(0, I) \right) \end{aligned}$$

where $u = \mu(x; \lambda)$, $s = \sigma(x; \lambda)$, and $\psi = f(z = u + \epsilon \odot s, \theta)$

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$p(z|x, \theta) = \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)}$$

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$p(z|x, \theta) = \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)}$$
$$= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)}$$

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$p(z|x, \theta) = \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)}$$

$$= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} = \frac{p(z)p(x|\theta)}{p(x|\theta)}$$

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$
\begin{aligned}
p(z|x, \theta) &= \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)} \\
&= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} = \frac{p(z)p(x|\theta)}{p(x|\theta)} \\
&= p(z)
\end{aligned}
$$

Posterior collapse

We are point estimating $p(x, z|\theta)$ along with $q(z|x, \lambda)$

- where $p(x, z|\theta) = p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)$
- if we pick θ such that $X_i \perp Z \mid X_{<i}$, then

$$
\begin{aligned}
p(z|x, \theta) &= \frac{p(z) \prod_{i=1}^{n} p(x_i|z, x_{<i}, \theta)}{p(x|\theta)} \\
&= \frac{p(z) \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)}{p(x|\theta)} = \frac{p(z)p(x|\theta)}{p(x|\theta)} \\
&= p(z)
\end{aligned}
$$

- the **true posterior** *collapses* to the prior

Strong generators

If your likelihood model is able to express dependencies between the output variables (e.g. an RNN), the model may simply ignore the latent code.

Note that though $X \perp Z$ (or $X_i \perp Z \mid X_{<i}$) $\prod_{i=1}^{n} p(x_i|x_{<i}, \theta)$ *still is* an exact factorisation of $p(x|\theta)$.

We call such models *strong generators*.

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)]$ is an upperbound on $I(X;Z|\lambda)$

$I(X;Z|\lambda) = \int\int q(x,z|\lambda)\log\frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)}\mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)]$ is an upperbound on $I(X;Z|\lambda)$

- if $\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$ is close to 0 to most training instances, then $I(X;Z|\lambda)$ is 0 or negligible;

$I(X;Z|\lambda) = \int \int q(x,z|\lambda) \log \frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)} \mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\left(q(z|x,\lambda) \,||\, p(z)\right)]$ is an upperbound on $I(X;Z|\lambda)$

- if $\mathrm{KL}\left(q(z|x,\lambda) \,||\, p(z)\right)$ is close to 0 to most training instances, then $I(X;Z|\lambda)$ is 0 or negligible;
- greedy decoding $\arg\max_{x_i} \log p(x_i|z, x_{<i})$ from a prior sample $z \sim p(z)$ is deterministic;

$I(X;Z|\lambda) = \int\int q(x,z|\lambda) \log \frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)} \mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

Diagnosing posterior collapse

Fact: the *rate* $R = \mathbb{E}_X[\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)]$ is an upperbound on $I(X;Z|\lambda)$

- if $\mathrm{KL}\left(q(z|x,\lambda) \mid\mid p(z)\right)$ is close to 0 to most training instances, then $I(X;Z|\lambda)$ is 0 or negligible;
- greedy decoding $\arg\max_{x_i} \log p(x_i|z, x_{<i})$ from a prior sample $z \sim p(z)$ is deterministic;
- this does not mean ancestral samples from $p(x|z,\theta)$ will be bad

$I(X;Z|\lambda) = \int\int q(x,z|\lambda) \log \frac{q(x,z|\lambda)}{q_\star(x)q(z|\lambda)} \mathrm{d}x\mathrm{d}z$ and $q(x,z|\lambda) = q_\star(x)q(z|x,\lambda)$.

KL scaling

Gradually incorporate the KL term into the objective

$$\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \beta\,\mathrm{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)$$

where β starts at 0 and goes to 1 after a number of steps.

KL scaling

Gradually incorporate the KL term into the objective

$$\mathbb{E}_{q(z|x,\lambda)}\left[\log p(x|z,\theta)\right] - \beta\,\mathrm{KL}\left(q(z|x,\lambda)\;||\;p(z)\right)$$

where β starts at 0 and goes to 1 after a number of steps.

This sometimes helps reach better local optimum, but there are not guarantees. In fact, oftentimes, soon after we reach 1, the posterior collapses again.

Free bits

Another strategy is to promote the posterior to deviate a bit from the prior by not penalising for the first few nats of information:

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

where $r \geq 0$ is known as "free bits"

This is an attempt to promote solutions where $R \geq r$

Attention!

But note that if we scale down the KL term permanently, or allow too many free bits, then the conditional $p(x|z, \theta)$ will over-specialise to samples from the approximate posterior $q(z|x, \lambda)$. This can lead to bad generalisation and/or poor samples when generating from the prior.

Variational Autoencoder

Advantages

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

Variational Autoencoder

Advantages

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

Drawbacks

- Discrete latent variables are not possible
- Optimisation may be difficult with several latent variables

Summary

- Wake-Sleep: train inference and generation networks with separate objectives
- VAE: train both networks with same objective
- Reparametrisation
  - Transform parameter-free variable ϵ into latent value z
  - Update parameters with stochastic gradient estimates
- If you employ strong generators, watch out for posterior collapse

Implementation

Try one of our notebooks, e.g.

- Original VAE: MNIST
  https://github.com/vitutorial/VITutorial/blob/master/code/vae_notebook_pytorch.ipynb
- SentenceVAE
  https://github.com/probabll/dgm4nlp/tree/master/notebooks/sentencevae
- WordVAE
  https://github.com/vitutorial/exercises/tree/master/WordVAE

Literature I

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy. Fixing a broken elbo. In *International Conference on Machine Learning*, pages 159–168, 2018.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. In *International Conference on Machine Learning*, 2017.

Literature II

G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161, 1995. URL http://www.gatsby.ucl.ac.uk/~dayan/papers/hdfn95.pdf.

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

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.

Literature III

Tom Pelsmaeker and Wilker Aziz. Effective estimation of deep generative language models. *arXiv preprint arXiv:1904.08194*, 2019.

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.

Literature IV

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.