Normalising Flows

Philip Schulz
https://vitutorial.github.io
https://github.com/vitutorial/VITutorial

The Case of Pictures

Have you modeled pixels as Gaussian variables? Do we really believe that the pixels follow a Gaussian distribution?

The case of Word Embeddings

The case of Word Embeddings

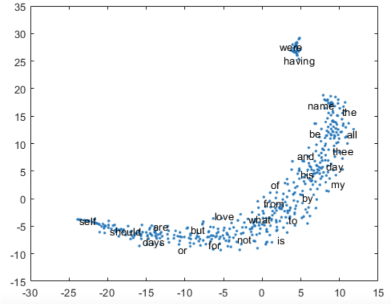


Figure: https://it.mathworks.com/help/textanalytics/ref/trainwordembedding.html

Posterior Approximations

We often use exponential families to approximate posteriors. Thus we assume unimodal posteriors. Is that realistic?

Posterior Approximations

We often use exponential families to approximate posteriors. Thus we assume unimodal posteriors. Is that realistic?

Counter example

Gaussian mixture model

Recap: Change Of Variable

What if a function is multivariate

The Jacobian matrix $J_{\mathcal{T}}(x)$ of $\mathcal{T}$
assessed at x is the matrix of partial derivatives

$$J_{ij} = \frac{\partial y_i}{\partial x_j}$$

Recap: Change Of Variable

What if a function is multivariate

The Jacobian matrix $J_{\mathcal{T}}(x)$ of $\mathcal{T}$
assessed at x is the matrix of partial derivatives

$$J_{ij} = \frac{\partial y_i}{\partial x_j}$$

Inverse function theorem

$$J_{\mathcal{T}^{-1}}(y) = \left(J_{\mathcal{T}}(x)\right)^{-1}$$

Recap: Change Of Variable

Express the density of a variable Y in terms of the density of a variable X. Assume that a differentiable, invertible mapping $h : \mathcal{X} \to \mathcal{Y}$ exists.

$$h(x) = y$$

Recap: Change Of Variable

Express the density of a variable Y in terms of the density of a variable X. Assume that a differentiable, invertible mapping $h : \mathcal{X} \to \mathcal{Y}$ exists.

$$h(x) = y$$

$$p_Y(y) = p_X(x)|\det J_{h^{-1}}(y)| = p_X(h^{-1}(y))|\det J_{h^{-1}}(y)|$$

Recap: Change Of Variable

Express the density of a variable Y in terms of the density of a variable X. Assume that a differentiable, invertible mapping $h : \mathcal{X} \to \mathcal{Y}$ exists.

$$
\begin{aligned}
h(x) &= y \\
p_Y(y) &= p_X(x)|\det J_{h^{-1}}(y)| = p_X(h^{-1}(y))|\det J_{h^{-1}}(y)| \\
p_X(x) &= p_Y(y)|\det J_h(x)| = p_Y(h(x))|\det J_h(x)|
\end{aligned}
$$

Recap: Change Of Variable

Express the density of a variable Y in terms of the density of a variable X. Assume that a differentiable, invertible mapping $h : \mathcal{X} \to \mathcal{Y}$ exists.

$$
\begin{aligned}
h(x) &= y \\
p_Y(y) &= p_X(x)|\det J_{h^{-1}}(y)| = p_X(h^{-1}(y))|\det J_{h^{-1}}(y)| \\
p_X(x) &= p_Y(y)|\det J_h(x)| = p_Y(h(x))|\det J_h(x)|
\end{aligned}
$$

The Challenge

The mapping h (or its inverse) needs to be defined.

Normalising Flows

Approach

Let's learn the transformation h (or its inverse).

Normalising Flows

Approach

Let's learn the transformation h (or its inverse).

Problem

If we want to express $p(y)$ in terms of $p(x)$, we need to provide $|\det J_{h^{-1}}(y)|$ **in the forward pass**. But that's hard!

We are going to devise ways to get $|\det J_{h^{-1}}(y)|$.

Normalising Flows

Core Idea

Decompose mapping $h : \mathcal{X} \to \mathcal{Y}$ into

$$h = h_1 \circ h_2 \circ \ldots \circ h_K \ .$$

Normalising Flows

Core Idea

Decompose mapping $h : \mathcal{X} \to \mathcal{Y}$ into

$$h = h_1 \circ h_2 \circ \ldots \circ h_K \ .$$

Now we can learn K mappings with simple jacobian determinants.

Normalising Flows

Core Idea

Decompose mapping $h : \mathcal{X} \to \mathcal{Y}$ into

$$h = h_1 \circ h_2 \circ \ldots \circ h_K \ .$$

Now we can learn K mappings with simple jacobian determinants.

$$h^{-1} = h_K^{-1} \circ h_{K-1}^{-1} \circ \ldots \circ h_1^{-1}$$

Normalising Flows

Core Idea

Decompose mapping $h : \mathcal{X} \to \mathcal{Y}$ into

$$h = h_1 \circ h_2 \circ \ldots \circ h_K \ .$$

Now we can learn K mappings with simple jacobian determinants.

$$h^{-1} = h_K^{-1} \circ h_{K-1}^{-1} \circ \ldots \circ h_1^{-1}$$
$$p_X(x) = p_Y(y) \left| \det J_{h_1} \left(y^{(1)}\right) \right| \left| \det J_{h_2} \left(y^{(2)}\right) \right| \ldots \left| \det J_{h_K} (x) \right|$$

Normalising Flows

Core Idea

Decompose mapping $h : \mathcal{X} \rightarrow \mathcal{Y}$ into

$$h = h_1 \circ h_2 \circ \ldots \circ h_K \ .$$

Now we can learn K mappings with simple jacobian determinants.

$$h^{-1} = h_K^{-1} \circ h_{K-1}^{-1} \circ \ldots \circ h_1^{-1}$$

$$p_X(x) = p_Y(y) \left| \det J_{h_1} \left(y^{(1)}\right) \right| \left| \det J_{h_2} \left(y^{(2)}\right) \right| \ldots \left| \det J_{h_K} (x) \right|$$

$$p_Y(y) = p_X(x) \left| \det J_{h_K^{-1}} \left(y^{(K-1)}\right) \right| \left| \det J_{h_{K-1}^{-1}} \left(y^{(K-2)}\right) \right| \ldots \left| \det J_{h_1^{-1}} (y) \right|$$

Normalising Flows

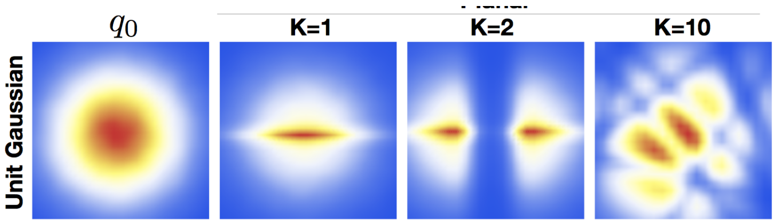


Figure: Taken from Rezende and Mohamed (2015)

Inverse Autoregressive Flows: Density Estimation

Setting

Our data x has unknown continuous density $p(x)$. We can therefore not handcraft a likelihood.

Inverse Autoregressive Flows: Density Estimation

Setting

Our data x has unknown continuous density $p(x)$. We can therefore not handcraft a likelihood.

Examples: word embeddings, pictures

Inverse Autoregressive Flows: Density Estimation

Setting

Our data x has unknown continuous density $p(x)$. We can therefore not handcraft a likelihood.
Examples: word embeddings, pictures

Goal

Transform unknown (observed) variable x into $\epsilon = h(x)$ with **known** density $p_\epsilon(\epsilon)$ and express the likelihood as

Inverse Autoregressive Flows: Density Estimation

Setting

Our data x has unknown continuous density $p(x)$. We can therefore not handcraft a likelihood.

Examples: word embeddings, pictures

Goal

Transform unknown (observed) variable x into $\epsilon = h(x)$ with **known** density $p_\epsilon(\epsilon)$ and express the likelihood as

$$p_X(x) = p_\epsilon(\epsilon)\left|\det J_h(x)\right|$$

Inverse Autoregressive Flows: Density Estimation

Setting

Our data x has unknown continuous density $p(x)$. We can therefore not handcraft a likelihood.

Examples: word embeddings, pictures

Goal

Transform unknown (observed) variable x into $\epsilon = h(x)$ with **known** density $p_\epsilon(\epsilon)$ and express the likelihood as

$$\begin{aligned} p_X(x) &= p_\epsilon(\epsilon)\left|\det J_h(x)\right| \\ &= p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}\left(\epsilon^{(2)}\right)\right|\ldots\left|\det J_{h_K}(x)\right| \end{aligned}$$

Inverse Autoregressive Flows: Density Estimation

Setting

Our data x has unknown continuous density $p(x)$. We can therefore not handcraft a likelihood.

Examples: word embeddings, pictures

Goal

Transform unknown (observed) variable x into $\epsilon = h(x)$ with **known** density $p_\epsilon(\epsilon)$ and express the likelihood as

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\epsilon)\left|\det J_h\left(x\right)\right| \\
&= p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}\left(\epsilon^{(2)}\right)\right| \dots \left|\det J_{h_K}\left(x\right)\right| \\
&= p_\epsilon(h_1(\epsilon^{(1)}))\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right| \dots \left|\det J_{h_K}\left(x\right)\right|
\end{aligned}
$$

2-step Flow

$$p_X(x) = p_\epsilon(\overbrace{\epsilon}^{\epsilon^{(0)}})\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right|$$

2-step Flow

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\overbrace{\epsilon}^{\epsilon^{(0)}})\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right| \\
&= p_\epsilon(h_1(h_2(x)))\left|\det J_{h_1}(h_2(x))\right|\left|\det J_{h_2}(x)\right|
\end{aligned}
$$

2-step Flow

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\overbrace{\epsilon}^{\epsilon^{(0)}})\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right||\det J_{h_2}(x)| \\
&= p_\epsilon(h_1(h_2(x)))|\det J_{h_1}(h_2(x))||\det J_{h_2}(x)|
\end{aligned}
$$

The transformations h_1 and h_2 are learned by backprop (while still being invertible). The determinants need to be computed analytically.

Designing a Transformation

Assume: $x = (x_1, x_2, \ldots x_M)$. Then factorise the density according to the chain rule.

$$\log p(x|\theta) = \sum_{j=1}^{M} \log p(x_j|x_{<j}\theta)$$

Next assume an invertible mapping $h(x_j) = \epsilon_j$.

Simple Mapping

$$h(x) = \epsilon$$
$$h^{-1}(\epsilon) = x$$

Designing a Transformation

Assume: $x = (x_1, x_2, \ldots x_M)$. Then factorise the density according to the chain rule.

$$\log p(x|\theta) = \sum_{j=1}^{M} \log p(x_j|x_{<j}\theta)$$

Next assume a mapping $h(x_j) = \epsilon_j$.

Flow Mapping

$$h_1 \circ h_2 \circ \ldots \circ h_K(x) = \epsilon$$
$$h_K^{-1} \circ h_{K-1}^{-1} \circ \ldots \circ h_1^{-1}(\epsilon) = x$$

Designing a Transformation

Assume: $x = (x_1, x_2, \dots x_M)$. Then factorise the density according to the chain rule.

$$\log p(x|\theta) = \sum_{j=1}^{M} \log p(x_j|x_{<j}\theta)$$

Next assume a mapping $h(x_j) = \epsilon_j$.

Flow Mapping

$$h_1 \circ h_2 \circ \dots \circ h_K(x) = \epsilon$$
$$h_K^{-1} \circ h_{K-1}^{-1} \circ \dots \circ h_1^{-1}(\epsilon) = x = \epsilon^{(K)}$$

Designing a Transformation

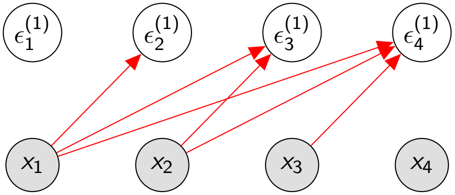
MADE (Germain et al., 2015)

An autoregressive network that takes constant time. Its connectivity matrix is lower-triangular.

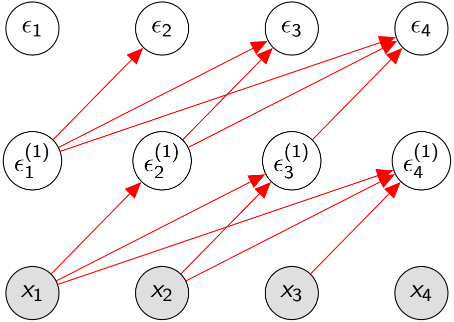
$$\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 & 0 \end{bmatrix}$$

Designing a Transformation

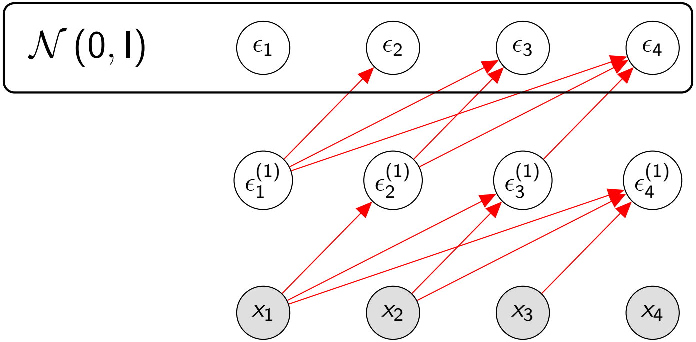
Designing a Transformation



Designing a Transformation



Designing a Transformation



Designing a Transformation

We use a MADE $g_\theta^{(2)}$ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = g_\theta^{(2)}(x_{<j})$. Then we apply the first transformation.

$$\epsilon_j^{(1)} = h_2(x)_j \ = \frac{x_j - \mu(x_{<j})}{\sigma(x_{<j})}$$

Designing a Transformation

We use a MADE $g_\theta^{(2)}$ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = g_\theta^{(2)}(x_{<j})$. Then we apply the first transformation.

$$\epsilon_j^{(1)} = h_2(x)_j = \frac{x_j - \mu(x_{<j})}{\sigma(x_{<j})}$$

$$\epsilon^{(1)} = h_2(x) = \frac{x - \mu}{\sigma}$$

Designing a Transformation

We use a MADE $g_\theta^{(2)}$ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = g_\theta^{(2)}(x_{<j})$. Then we apply the first transformation.

$$\begin{aligned}\epsilon_j^{(1)} = h_2(x)_j &= \frac{x_j - \mu(x_{<j})}{\sigma(x_{<j})} \\ \epsilon^{(1)} = h_2(x) &= \frac{x - \mu}{\sigma}\end{aligned}$$

The Jacobian is

$$J_{h_2}(x) =$$

Designing a Transformation

We use a MADE $g_\theta^{(2)}$ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = g_\theta^{(2)}(x_{<j})$. Then we apply the first transformation.

$$\begin{aligned}\epsilon_j^{(1)} = h_2(x)_j &= \frac{x_j - \mu(x_{<j})}{\sigma(x_{<j})} \\ \epsilon^{(1)} = h_2(x) &= \frac{x - \mu}{\sigma}\end{aligned}$$

The Jacobian is

$$J_{h_2}(x) = \mathrm{I}\,\sigma^{-1} + J_{\frac{-\mu}{\sigma}}(x)$$

Designing a Transformation

Define $\alpha_{lj} = \frac{d}{dx_l}\frac{-\mu_j}{\sigma_j}$.

$$J_{h_K^{-1}}(x) = \mathrm{I}\,\sigma^{-1} + J_{\frac{-\mu}{\sigma}}(x) =$$

Designing a Transformation

Define $\alpha_{lj} = \frac{d}{dx_l}\frac{-\mu_j}{\sigma_j}$.

$$J_{h_K^{-1}}(x) = \mathrm{I}\,\sigma^{-1} + J_{\frac{-\mu}{\sigma}}(x) =$$

$$\begin{bmatrix} \sigma_{11}^{-1} & 0 & \cdots & 0 & 0 \\ 0 & \sigma_{22}^{-1} & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \sigma_{mm}^{-1} \end{bmatrix}$$

Designing a Transformation

Define $\alpha_{lj} = \frac{d}{dx_l}\frac{-\mu_j}{\sigma_j}$.

$$J_{h_K^{-1}}(x) = \mathrm{I}\,\sigma^{-1} + J_{\frac{-\mu}{\sigma}}(x) =$$

$$\begin{bmatrix} \sigma_{11}^{-1} & 0 & \cdots & 0 & 0 \\ 0 & \sigma_{22}^{-1} & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \sigma_{mm}^{-1} \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ \alpha_{21} & 0 & \cdots & 0 & 0 \\ \alpha_{31} & \alpha_{32} & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{m,m-1} & 0 \end{bmatrix}$$

Designing a Transformation

Simple Jacobian Determinant

$$\left|\det J_{h_2}(x)\right| = \prod_{j=1}^{M} \sigma_j^{-1}$$

Designing a Transformation

Simple Jacobian Determinant

$$\left|\det J_{h_2}(x)\right| = \prod_{j=1}^{M} \sigma_j^{-1}$$

In practice we work with the log-likelihood.

$$\log\left|\det J_{h_2}(x)\right| = -\sum_{j=1}^{M} \log \sigma_j$$

2-step Flow

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right| \\
&= p_\epsilon(h_1(h_2(x)))\left|\det J_{h_1}(h_2(x))\right|\left|\det J_{h_2}(x)\right|
\end{aligned}
$$

2-step Flow

$$p_X(x) = p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right|$$
$$= p_\epsilon(h_1(h_2(x)))\left|\det J_{h_1}(h_2(x))\right|\left|\det J_{h_2}(x)\right|$$

$$\log p_X(x) = \log p_\epsilon(h_1(h_2(x)))$$

2-step Flow

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right| \\
&= p_\epsilon(h_1(h_2(x)))\left|\det J_{h_1}(h_2(x))\right|\left|\det J_{h_2}(x)\right|
\end{aligned}
$$

$$
\log p_X(x) = \log p_\epsilon(h_1(h_2(x))) - \sum_{j=1}^{M} \log \sigma_j^{(2)} - \sum_{j=1}^{M} \log \sigma_j^{(1)}
$$

2-step Flow

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right| \\
&= p_\epsilon(h_1(h_2(x)))\left|\det J_{h_1}(h_2(x))\right|\left|\det J_{h_2}(x)\right|
\end{aligned}
$$

$$
\log p_X(x) = \log p_\epsilon(h_1(h_2(x))) - \sum_{j=1}^{M} \log \sigma_j^{(2)} - \sum_{j=1}^{M} \log \sigma_j^{(1)}
$$

$$
\epsilon^{(1)} = h_2 = \frac{x - \mu^{(2)}}{\sigma^{(2)}} \text{ where } \left[\mu^{(2)}, \sigma^{(2)}\right] = g^{(2)}(x)
$$

2-step Flow

$$
\begin{aligned}
p_X(x) &= p_\epsilon(\epsilon)\left|\det J_{h_1}\left(\epsilon^{(1)}\right)\right|\left|\det J_{h_2}(x)\right| \\
&= p_\epsilon(h_1(h_2(x)))\left|\det J_{h_1}(h_2(x))\right|\left|\det J_{h_2}(x)\right|
\end{aligned}
$$

$$
\log p_X(x) = \log p_\epsilon(h_1(h_2(x))) - \sum_{j=1}^{M} \log \sigma_j^{(2)} - \sum_{j=1}^{M} \log \sigma_j^{(1)}
$$

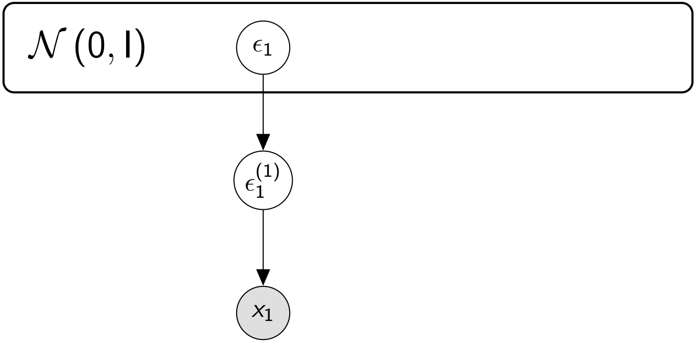
$$
\epsilon^{(1)} = h_2 = \frac{x - \mu^{(2)}}{\sigma^{(2)}} \text{ where } \left[\mu^{(2)}, \sigma^{(2)}\right] = g^{(2)}(x)
$$

$$
\epsilon = h_1 = \frac{\epsilon^{(1)} - \mu^{(1)}}{\sigma^{(1)}} \text{ where } \left[\mu^{(1)}, \sigma^{(1)}\right] = g^{(1)}(\epsilon^{(1)})
$$

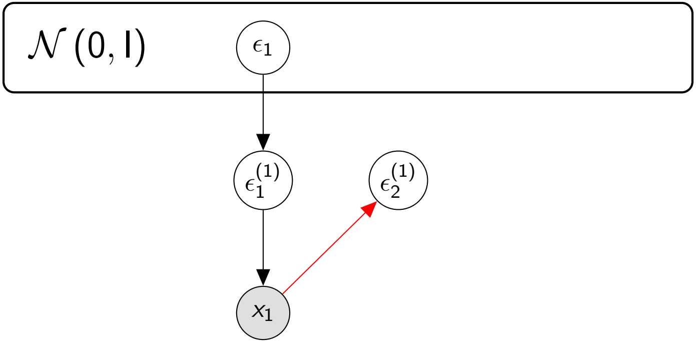
Intermediate Summary

- NFs map transform complex distributions to simpler ones (or vice versa)
- Use in density estimation for complex distributions
- Jacobian needs to be carefully designed
- Sampling is slow because sequential

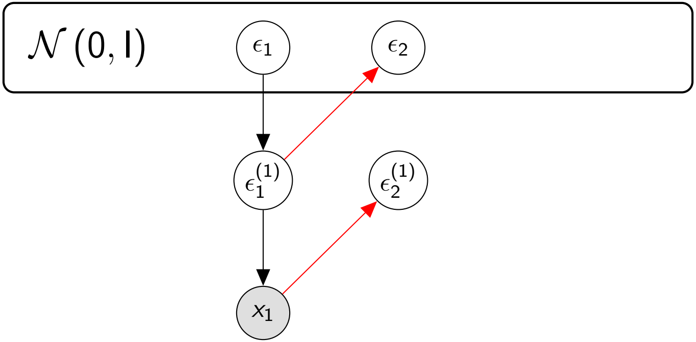
Sampling From the Flow



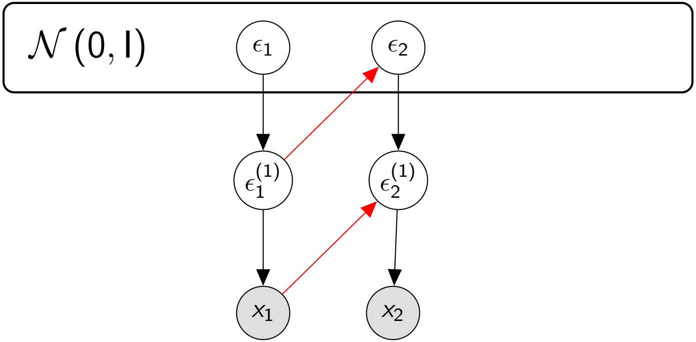
Sampling From the Flow



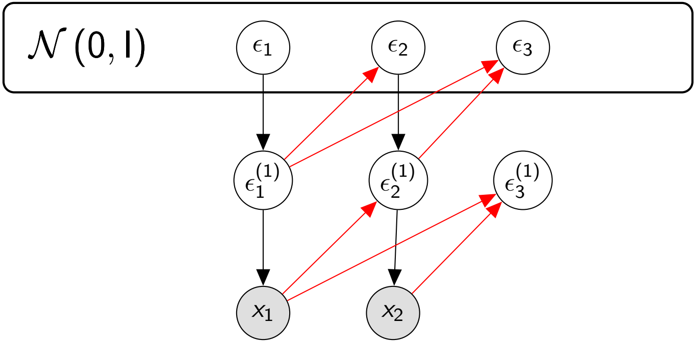
Sampling From the Flow



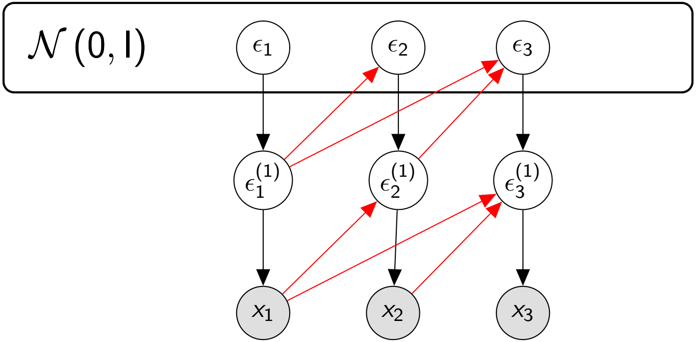
Sampling From the Flow



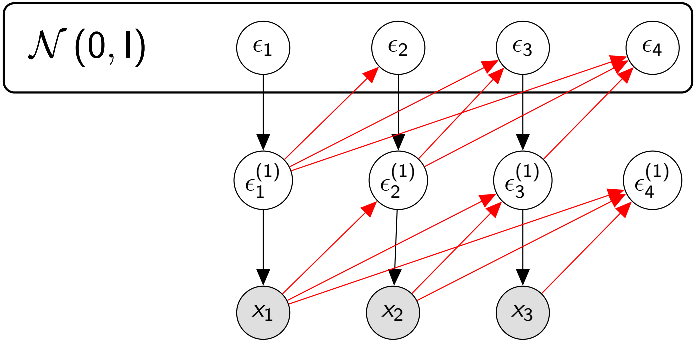
Sampling From the Flow



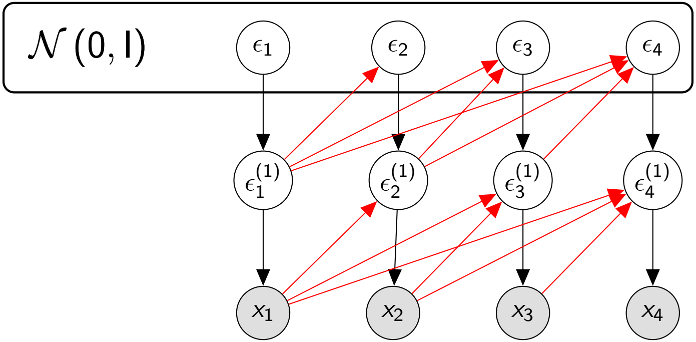
Sampling From the Flow



Sampling From the Flow



Sampling From the Flow



Setting

We have a generative model $p(x, z)$. We want to approximate the posterior $p(z|x)$ using an amortized variational distribution $q(z|x)$ computed by a neural net.

Setting

We have a generative model $p(x, z)$. We want to approximate the posterior $p(z|x)$ using an amortized variational distribution $q(z|x)$ computed by a neural net.

Goal

We want a complex, multimodal approximate posterior $q(z|x)$.

Inverse Autoregressive Flow: Inference

$$
\begin{aligned}
\text{ELBO} &= -\,\text{KL}\left(q(z|x) \;||\; p(z|x)\right) \\
&= \mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \text{KL}\left(q(z|\lambda)) \;||\; p(z)\right) \\
&= \underbrace{\mathbb{E}_{q(\epsilon)}\left[\log p(x|h^{-1}(\epsilon))\right]}_{\text{sample } z} - \underbrace{\text{KL}\left(q(z|\lambda) \;||\; p(z)\right)}_{\text{assess density}}
\end{aligned}
$$

Simple Mapping

$$
\begin{aligned}
h(z) &= \epsilon \text{ s.t. } \epsilon \perp \lambda \\
h^{-1}(\epsilon) &= z
\end{aligned}
$$

Normalising Flows: Inference

$$
\begin{aligned}
&- \mathrm{KL}\left(q(z|x) \,||\, p(z|x)\right) \propto \mathrm{ELBO} = \\
&= \mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \mathrm{KL}\left(q(z|\lambda)) \,||\, p(z)\right) \\
&= \underbrace{\mathbb{E}_{q(\epsilon)}\left[\log p(x|h^{-1}(\epsilon))\right]}_{\text{sample } z} - \underbrace{\mathrm{KL}\left(q(z|\lambda) \,||\, p(z)\right)}_{\text{assess density}}
\end{aligned}
$$

Flow Mapping

$$
h_1 \circ h_2 \circ \ldots \circ h_K(z) = \epsilon \text{ s.t. } \epsilon \perp \lambda
$$

$$
h_K^{-1} \circ h_{K-1}^{-1} \circ \ldots \circ h_1^{-1}(\epsilon) = z
$$

Normalising Flows: Inference

$$
\begin{aligned}
&- \mathrm{KL}\left(q(z|x) \,||\, p(z|x)\right) \propto \mathrm{ELBO} = \\
&= \mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \mathrm{KL}\left(q(z|\lambda)) \,||\, p(z)\right) \\
&= \underbrace{\mathbb{E}_{q(\epsilon)}\left[\log p(x|h^{-1}(\epsilon))\right]}_{\text{sample } z} - \underbrace{\mathrm{KL}\left(q(z|\lambda) \,||\, p(z)\right)}_{\text{assess density}}
\end{aligned}
$$

Flow Mapping

$$
\begin{aligned}
h_1 \circ h_2 \circ \ldots \circ h_K(z) = \epsilon \text{ s.t. } \epsilon \perp \lambda \\
h_K^{-1} \circ h_{K-1}^{-1} \circ \ldots \circ h_1^{-1}(\epsilon) = z = z^{(K)}
\end{aligned}
$$

2-step Flow

$$q_Z(z^{(2)}) = q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(z^{(1)}\right)\right|\left|\det J_{h_2^{-1}}\left(z^{(2)}\right)\right|$$

2-step Flow

$$
\begin{aligned}
q_Z(z^{(2)}) &= q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(z^{(1)}\right)\right|\left|\det J_{h_2^{-1}}\left(z^{(2)}\right)\right| \\
&= q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(h_1^{-1}(\epsilon)\right)\right|\left|\det J_{h_2^{-1}}\left(h_2^{-1}(h_1^{-1}(\epsilon))\right)\right|
\end{aligned}
$$

2-step Flow

$$q_Z(z^{(2)}) = q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(z^{(1)}\right)\right|\left|\det J_{h_2^{-1}}\left(z^{(2)}\right)\right|$$
$$= q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(h_1^{-1}(\epsilon)\right)\right|\left|\det J_{h_2^{-1}}\left(h_2^{-1}(h_1^{-1}(\epsilon))\right)\right|$$

The transformations h_1^{-1} and h_2^{-1} are learned by backprop (while still being invertible). The determinants need to be computed analytically.

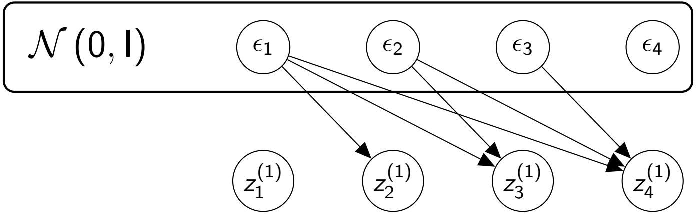
Designing a Transformation

We are again going to use a MADE to predict parameters.
However, this time we will use it in the other direction.

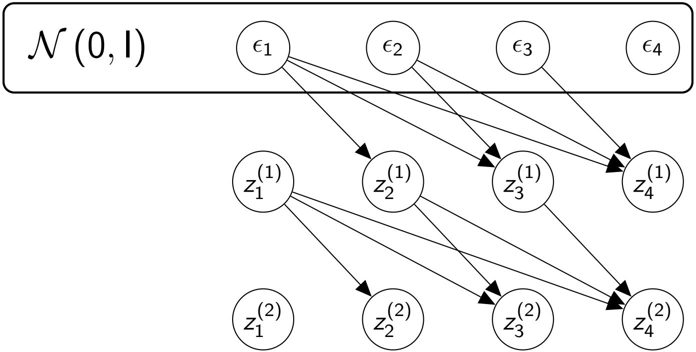
Designing a Transformation

$\mathcal{N}(0, I)$ $\quad \epsilon_1 \quad \epsilon_2 \quad \epsilon_3 \quad \epsilon_4$

Designing a Transformation



Designing a Transformation



Designing a Transformation

We use a MADE f_λ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = f_\lambda(\epsilon_{<j})$. Then we apply the first transformation.

$$z_j^{(1)} = h_1^{-1}(\epsilon)_j \; = \mu(\epsilon_{<j}) + \sigma(\epsilon_{<j})\epsilon_j$$

Designing a Transformation

We use a MADE f_λ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = f_\lambda(\epsilon_{<j})$. Then we apply the first transformation.

$$\begin{aligned} z_j^{(1)} &= h_1^{-1}(\epsilon)_j &= \mu(\epsilon_{<j}) + \sigma(\epsilon_{<j})\epsilon_j \\ z^{(1)} &= h_1^{-1}(\epsilon) &= \mu + \sigma\epsilon \end{aligned}$$

Designing a Transformation

We use a MADE f_λ to predict the parameters of the first transformation: $\begin{bmatrix}\mu_j & \sigma_j\end{bmatrix} = f_\lambda(\epsilon_{<j})$. Then we apply the first transformation.

$$
\begin{aligned}
z_j^{(1)} &= h_1^{-1}(\epsilon)_j &&= \mu(\epsilon_{<j}) + \sigma(\epsilon_{<j})\epsilon_j \\
z^{(1)} &= h_1^{-1}(\epsilon) &&= \mu + \sigma\epsilon
\end{aligned}
$$

$$
J_{h_1^{-1}}(\epsilon) = \mathrm{I}\sigma + J_\mu(\epsilon) + J_{\sigma\epsilon}(\epsilon)
$$

Designing a Transformation

Simple Jacobian Determinant

$$\left|\det J_{h_1^{-1}}(\epsilon)\right| = \prod_{j=1}^{M} \sigma_j$$

In practice we work with the log-likelihood.

$$\log \left|\det J_{h_1^{-1}}(\epsilon)\right| = \sum_{j=1}^{M} \log \sigma_j$$

2-step Flow

$$
\begin{aligned}
q_Z(z^{(2)}) &= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(z^{(1)} \right) \right| \left| \det J_{h_2^{-1}} \left(z^{(2)} \right) \right| \\
&= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(h_1^{-1}(\epsilon) \right) \right| \left| \det J_{h_2^{-1}} \left(h_2^{-1}(h_1^{-1}(\epsilon)) \right) \right|
\end{aligned}
$$

2-step Flow

$$
\begin{aligned}
q_Z(z^{(2)}) &= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(z^{(1)} \right) \right| \left| \det J_{h_2^{-1}} \left(z^{(2)} \right) \right| \\
&= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(h_1^{-1}(\epsilon) \right) \right| \left| \det J_{h_2^{-1}} \left(h_2^{-1}(h_1^{-1}(\epsilon)) \right) \right|
\end{aligned}
$$

$$
\log q_Z(z^{(2)}) = \log q_\epsilon(\epsilon)
$$

2-step Flow

$$
\begin{aligned}
q_Z(z^{(2)}) &= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(z^{(1)} \right) \right| \left| \det J_{h_2^{-1}} \left(z^{(2)} \right) \right| \\
&= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(h_1^{-1}(\epsilon) \right) \right| \left| \det J_{h_2^{-1}} \left(h_2^{-1}(h_1^{-1}(\epsilon)) \right) \right| \\
\log q_Z(z^{(2)}) &= \log q_\epsilon(\epsilon) + \sum_{j=1}^{M} \log \sigma_j^{(1)} + \sum_{j=1}^{M} \log \sigma_j^{(2)}
\end{aligned}
$$

2-step Flow

$$
\begin{aligned}
q_Z(z^{(2)}) &= q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(z^{(1)}\right)\right|\left|\det J_{h_2^{-1}}\left(z^{(2)}\right)\right| \\
&= q_\epsilon(\epsilon)\left|\det J_{h_1^{-1}}\left(h_1^{-1}(\epsilon)\right)\right|\left|\det J_{h_2^{-1}}\left(h_2^{-1}(h_1^{-1}(\epsilon))\right)\right|
\end{aligned}
$$

$$
\log q_Z(z^{(2)}) = \log q_\epsilon(\epsilon) + \sum_{j=1}^{M} \log \sigma_j^{(1)} + \sum_{j=1}^{M} \log \sigma_j^{(2)}
$$

$$
z^{(1)} = \mu^{(1)} + \sigma^{(1)}\epsilon \text{ where } \left[\mu^{(1)}, \sigma^{(1)}\right] = f_\lambda^{(1)}(\epsilon)
$$

2-step Flow

$$
\begin{aligned}
q_Z(z^{(2)}) &= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(z^{(1)} \right) \right| \left| \det J_{h_2^{-1}} \left(z^{(2)} \right) \right| \\
&= q_\epsilon(\epsilon) \left| \det J_{h_1^{-1}} \left(h_1^{-1}(\epsilon) \right) \right| \left| \det J_{h_2^{-1}} \left(h_2^{-1}(h_1^{-1}(\epsilon)) \right) \right|
\end{aligned}
$$

$$
\log q_Z(z^{(2)}) = \log q_\epsilon(\epsilon) + \sum_{j=1}^{M} \log \sigma_j^{(1)} + \sum_{j=1}^{M} \log \sigma_j^{(2)}
$$

$$
z^{(1)} = \mu^{(1)} + \sigma^{(1)}\epsilon \text{ where } \left[\mu^{(1)}, \sigma^{(1)}\right] = f_\lambda^{(1)}(\epsilon)
$$

$$
z^{(2)} = \mu^{(2)} + \sigma^{(2)} z^{(1)} \text{ where } \left[\mu^{(2)}, \sigma^{(2)}\right] = f_\lambda^{(2)}(z^{(1)})
$$

ELBO

Recall: $z = z^{(2)}$

$$\text{ELBO} = \mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \text{KL}\left(q(z)|\lambda)\right) \mid\mid p(z)) =$$

ELBO

Recall: $z = z^{(2)}$

$$\text{ELBO} = \mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \text{KL}\left(q(z)|\lambda)\right) \,||\, p(z)) =$$
$$\mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \text{KL}\left(q(\epsilon)|\det J_{h^{-1}}\,(z)|\,||\, p(z)\right)$$

ELBO

KL-term

$$\mathrm{KL}\left(q(\epsilon)|\det J_h\left(z\right)| \;||\; p(z)\right) =$$

ELBO

KL-term

$$\mathrm{KL}\left(q(\epsilon)|\det J_h\left(z\right)| \;||\; p(z)\right) =$$

$$\mathbb{E}_{q(z|\lambda))}\left[\log \frac{q(\epsilon)|\det J_{h^{-1}}\left(z\right)|}{p(z)}\right]$$

ELBO

KL-term

$$\mathrm{KL}\left(q(\epsilon)|\det J_h\left(z\right)| \mid\mid p(z)\right) =$$

$$\mathbb{E}_{q(z|\lambda))}\left[\log \frac{q(\epsilon)|\det J_{h^{-1}}\left(z\right)|}{p(z)}\right] \overset{\mathrm{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S} \log \frac{q(\epsilon)\left|\det J_{h^{-1}}\left(z^{(s)}\right)\right|}{p\left(z^{(s)}\right)}$$

ELBO

KL-term

$$\mathrm{KL}\left(q(\epsilon)|\det J_h(z)| \;||\; p(z)\right) =$$

$$\mathbb{E}_{q(z|\lambda))}\left[\log \frac{q(\epsilon)|\det J_{h^{-1}}(z)|}{p(z)}\right] \overset{\text{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S} \log \frac{q(\epsilon)\left|\det J_{h^{-1}}\left(z^{(s)}\right)\right|}{p(z^{(s)})}$$

Jacobian

ELBO

KL-term

$$\mathrm{KL}\left(q(\epsilon)|\det J_h(z)| \,||\, p(z)\right) =$$

$$\mathbb{E}_{q(z|\lambda))}\left[\log \frac{q(\epsilon)|\det J_{h^{-1}}(z)|}{p(z)}\right] \overset{\mathrm{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S} \log \frac{q(\epsilon)\left|\det J_{h^{-1}}\left(z^{(s)}\right)\right|}{p\left(z^{(s)}\right)}$$

Jacobian

$$\left|\det J_{h^{-1}}\left(z^{(s)}\right)\right| = \sum_{j=1}^{M} \log \sigma_j^{(1)} + \sum_{j=1}^{M} \log \sigma_j^{(2)}$$

Summary

- NFs model arbitrary continuous distributions
- They allow for density computation
- Need to have simple Jacobian determinants
- Depending on direction, they are good at either sampling or density computation (not both)

References I

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, pages 881–889, 2015.

Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron C. Courville. Neural autoregressive flows. *CoRR*, 2018. URL http://arxiv.org/abs/1804.00779.

References II

Diederik P. Kingma, Tim Salimans, and Max Welling. Improving variational inference with inverse autoregressive flow. 2016. URL http://arxiv.org/abs/1606.04934.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, pages 1530–1538, 2015. URL http://proceedings.mlr.press/v37/rezende15.pdf.