Normalising Flows

Philip Schulz and Wilker Aziz
https://github.com/philschulz/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

Posterior Approximations

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

Counter example

Gaussian mixture model

- The problem with Standard Likelihoods
- **Normalising Flows**
- Use Case 1: Density Estimation
- Use Case 2: Inference (sampling)
- Summary

Recap: Reparametrisation

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

$$h(x) = y$$

Recap: Reparametrisation

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.

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

The Challenge

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

Normalising Flows

Approach

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

Problem

If we want $p(y)$, 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 $h : \mathcal{X} \rightarrow \mathcal{Y}$ into

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

Now we can learn K mappings with simple Jacobians.

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

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

$$p(y) = p(x) \left|\det J_{h_1^{-1}}\left(y^{(K-1)}\right)\right| \left|\det J_{h_2^{-1}}\left(y^{(K-2)}\right)\right| \ldots \left|\det J_{h_1^{-1}}\left(y\right)\right|$$

Normalising Flows: Density Estimation

Setting

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

Goal

Transform known variable x into $\epsilon = h(x)$ and express the likelihood as

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

2-step Flow

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

2-step Flow

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

2-step Flow

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

The transformations h_1^{-1} and h_2^{-1} are learned by backprop. The determinants need to be computed analytically.

Designing a Transformation

Assume: $x_i = (x_{i1}, x_{i2}, \dots x_{iM})$. Then factorise the density according to the chain rule.

$$\log p(x_i|\theta) = \sum_{j=1}^{M} \log p(x_{ij}|x_{i,<j}\theta)$$

Next assume an invertible mapping $h(x_{ij}) = \epsilon_{ij}$.

Simple Mapping

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

Designing a Transformation

Assume: $x_i = (x_{i1}, x_{i2}, \ldots x_{iM})$. Then factorise the density according to the chain rule.

$$\log p(x_i|\theta) = \sum_{j=1}^{M} \log p(x_{ij}|x_{i,<j}\theta)$$

Next assume a mapping $h(x_{ij}) = \epsilon_{ij}$.

Flow Mapping

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

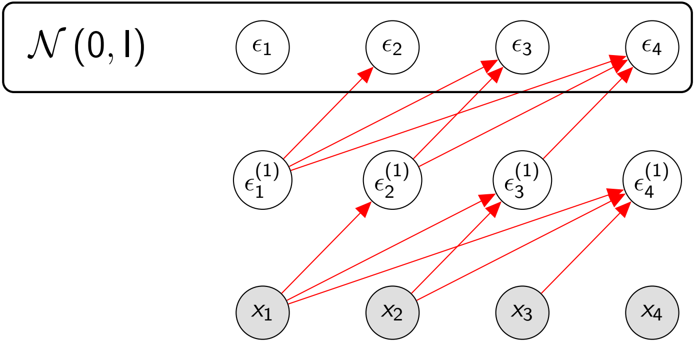
Designing a Transformation

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

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.

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

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.

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

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.

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

The Jacobian is

$$J_{h_2^{-1}}(x) =$$

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.

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

The Jacobian is

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

Designing a Transformation

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

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

Designing a Transformation

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

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

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

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

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

Designing a Transformation

Simple Jacobian Determinant

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

In practice we work with the log-likelihood.

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

2-step Flow

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

2-step Flow

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

$$
\log p(x) = \log p(h_1^{-1}(h_2^{-1}(x)))
$$

2-step Flow

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

2-step Flow

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

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

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

2-step Flow

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

$$= p(h_1^{-1}(h_2^{-1}(x)))\left|\det J_{h_1^{-1}}\left(h_2^{-1}(x)\right)\right|\left|\det J_{h_2^{-1}}(x)\right|$$

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

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

$$\epsilon = h_1^{-1} = \frac{\epsilon^{(1)} - \mu^{(2)}}{\sigma^{(2)}} \text{ where } \left[\mu^{(2)}, \sigma^{(2)}\right] = g(\epsilon^{(1)})$$

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

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 $q(z|x)$.

Normalising Flows: Inference

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

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

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

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

2-step Flow

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

2-step Flow

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

2-step Flow

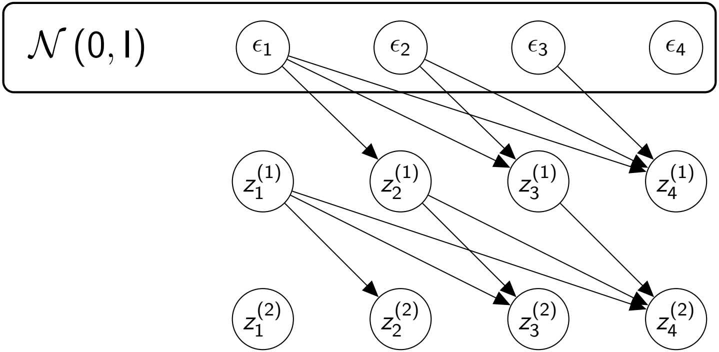
$$\begin{aligned} q(z^{(2)}) &= q(\epsilon)\left|\det J_{h_1}\left(z^{(1)}\right)\right|\left|\det J_{h_2}\left(z^{(2)}\right)\right| \\ &= q(h_1(h_2(z^{(2)})))\left|\det J_{h_1}\left(h_2(z^{(2)})\right)\right|\left|\det J_{h_2}\left(z^{(2)}\right)\right| \end{aligned}$$

The transformations h_1^{-1} and h_2^{-1} are learned by backprop. 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



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.

$$z_j^{(1)} = h_1(\epsilon)_j \; = \mu_j + \sigma_j \epsilon$$

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.

$$\begin{aligned} z_j^{(1)} &= h_1(\epsilon)_j &&= \mu_j + \sigma_j \epsilon \\ z^{(1)} &= h_1(\epsilon) &&= \mu + \sigma \epsilon \end{aligned}$$

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.

$$\begin{aligned} z_j^{(1)} = h_1(\epsilon)_j \quad &= \mu_j + \sigma_j \epsilon \\ z^{(1)} = h_1(\epsilon) \quad &= \mu + \sigma \epsilon \end{aligned}$$

$$J_{h_1}(\epsilon) = \mathrm{I}\,\sigma + J_\mu(\epsilon) + J_{\sigma\epsilon}(\epsilon)$$

Designing a Transformation

Simple Jacobian Determinant

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

In practice we work with the log-likelihood.

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

2-step Flow

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

$$\log q(z^{(2)}) = \log q(h_1^{-1}(h_2^{-1}(z^{(2)}))) + \sum_{j=1}^{M} \log \sigma_j^{(1)} + \sum_{j=1}^{M} \log \sigma_j^{(2)}$$

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

ELBO

$$\mathrm{ELBO} = \mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \mathrm{KL}\left(q(z^{(2)}|\lambda)) \,||\, p(z^{(2)})\right) =$$
$$\mathbb{E}_{q(z|\lambda)}\left[\log p(x|z)\right] - \mathrm{KL}\left(q(\epsilon)\left|\det J_h\left(z^{(2)}\right)\right| \,||\, p(z)\right)$$

ELBO

KL-term

$$\mathrm{KL}\left(q(\epsilon)\left|\det J_h\left(z^{(2)}\right)\right| \;||\; p(z)\right) =$$
$$\mathbb{E}_{q(z^{(2)}|\lambda))}\left[\frac{q(\epsilon)\left|\det J_h\left(z^{(2)}\right)\right|}{p\left(z^{(2)}\right)}\right] \overset{\text{MC}}{\approx} \frac{1}{S}\sum_{s=1}^{S}\frac{q(\epsilon)\left|\det J_h\left(z^{(2,s)}\right)\right|}{p\left(z^{(2,s)}\right)}$$

Jacobian

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

Other Appliations of Normalizing Flows

- As a prior
- Modeling of dynamic systems

Summary

- NFs model arbitrary continuous distributions
- They allow for density computation
- Need to have simple Jacobian
- 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.