COMPSCI 688: Probabilistic Graphical Models

Lecture 20: Learning with Stochastic Variational Inference

Dan Sheldon

Manning College of Information and Computer Sciences
University of Massachusetts Amherst

Partially based on materials by Benjamin M. Marlin (marlin@cs.umass.edu) and Justin Domke (domke@cs.umass.edu)

Review

Variational Auto-Encoder

Factor analysis model with non-linear mapping

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, I)$$
$$p(x_j|\mathbf{z}) = \text{Bernoulli}(x_j; (f_\theta(\mathbf{z}))_j), \qquad j = 1, \dots, d$$

Example non-linear mapping:

$$f_\theta(\mathbf{z}) = h_2(\mathbf{b}_2 + \mathbf{W}_2 \cdot h_1(\mathbf{b}_1 + \mathbf{W}_1\mathbf{z}))$$

Exact inference and learning are *intractable*.

Stochastic VI

Choose variational family, e.g., diagonal Gaussian, to approximate posterior:

$$q_\phi(\mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2)), \quad \phi = (\boldsymbol{\mu}, \boldsymbol{\sigma})$$

Stochastic optimization: repeatedly get **unbiased gradient estimate** $\hat{\nabla}_\phi$, update ϕ:

$$\hat{\nabla}_\phi \approx \nabla_\phi \text{ELBO}(\phi)$$

$$\phi \leftarrow \phi + \alpha \hat{\nabla}_\phi$$

How to get $\hat{\nabla}_\phi$?

Gradient Estimation: Reparameterization

	Without reparameterization	With reparameterization
Variational distribution	$q_\phi(z)$	$q_\phi(z)$
Sampling	$z \sim q_\phi(z)$	$\epsilon \sim q(\epsilon), z = \mathcal{T}_\phi(\epsilon)$
ELBO	$\mathbb{E}_{q_\phi(Z)}\left[\log \frac{p(Z,x)}{q_\phi(Z)}\right]$	$\mathbb{E}_{q(\epsilon)}\left[\log \frac{p(\mathcal{T}_\phi(\epsilon),x)}{q_\phi(\mathcal{T}_\phi(\epsilon))}\right]$
ELBO estimate	$\log \frac{p(z,x)}{q_\phi(z)}, z \sim q_\phi(z)$	$\log \frac{p(\mathcal{T}_\phi(\epsilon),x)}{q_\phi(\mathcal{T}_\phi(\epsilon))}, \epsilon \sim q(\epsilon)$
Gradient estimate	$\nabla_\phi \log \frac{p(z,x)}{q_\phi(z)}, z \sim q_\phi(z)$ (wrong/biased)	$\nabla_\phi \log \frac{p(\mathcal{T}_\phi(\epsilon),x)}{q_\phi(\mathcal{T}_\phi(\epsilon))}, \epsilon \sim q(\epsilon)$ (unbiased)

Reparameterization with Diagonal Gaussians

Suppose the variational family is a diagonal Gaussian

$$q_\phi(\mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$$

This can be reparameterized as:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

($\odot$ = elementwise multiplication)

Aside: Reparameterization with Arbitrary Gaussians

Another choice would be to use a general Gaussian distribution:

$$\epsilon \sim \mathcal{N}(0, I) \Longrightarrow \mu + L\epsilon \sim \mathcal{N}(\mu, LL^\top).$$

This is a reparameterization with

$$q(\epsilon) = \mathcal{N}(\epsilon | 0, I), \qquad \mathcal{T}_\phi(\epsilon) = \mu + L\epsilon \qquad \phi = (L, \mu)$$

It covers any multivariate Gaussian, since an arbitrary covariance matrix Σ can be written as $\Sigma = LL^\top$ for some L (e.g., a Cholesky factor)

Example: Bernoulli VAE

Let's return to our Bernoulli VAE factor analysis model and use a diagonal Gaussian approximation:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, I)$$
$$p(x_j|\mathbf{z}) = \mathrm{Bernoulli}(x_j; (f_\theta(\mathbf{z}))_j), \qquad j = 1, \ldots, d$$

$$q_\phi(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$$

BBSVI would repeat the following steps:

$$\epsilon \sim \mathcal{N}(0, I) \qquad\qquad (\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon)$$

$$\begin{aligned}\hat{\nabla}_{\mu,\sigma} = \nabla_{\mu,\sigma}\Big\{ & \log \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon;\ 0, I) \\ & + \sum_{j=1}^{d} \log \text{Bernoulli}\Big(x_j; \left(f_\theta(\boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon)\right)_j\Big) \\ & - \log \mathcal{N}\Big(\boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon;\ \boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2)\Big) \Big\}\end{aligned}$$

$$(\boldsymbol{\mu}, \boldsymbol{\sigma}) \leftarrow (\boldsymbol{\mu}, \boldsymbol{\sigma}) + \alpha \cdot \hat{\nabla}_{\mu,\sigma}$$

With the optimized parameters we could approximate $p(\mathbf{z}|\mathbf{x}) \approx q_\phi(\mathbf{z})$ and lower bound the log-marginal likelihood $\log p(\mathbf{x}) \geq \text{ELBO}(\phi)$. **What about learning?**

Learning with Stochastic Variational Inference

Learning with Stochastic Variational Inference

The basic idea is to jointly maximize the ELBO with respect to model parameters θ and variational parameters ϕ by getting unbiased gradient estimates for both:

$$\log p_\theta(\mathbf{x}) \geq \text{ELBO}(\theta, \phi) = \mathbb{E}_{q_\phi}\left[\log \frac{p_\theta(Z, x)}{q_\phi(Z)}\right]$$

$$\hat{\nabla}_\theta \approx \nabla_\theta \text{ELBO}(\theta, \phi)$$

$$\hat{\nabla}_\phi \approx \nabla_\phi \text{ELBO}(\theta, \phi)$$

$$(\theta, \phi) \leftarrow (\theta, \phi) + \alpha \cdot (\hat{\nabla}_\theta, \hat{\nabla}_\phi)$$

Learning with IID Data

How do we learn a latent variable model $p_\theta(\mathbf{z}, \mathbf{x})$ when we have iid data $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$?

Each datum $\mathbf{x}^{(n)}$ has its own:

- marginal likelihood $p_\theta(\mathbf{x}^{(n)})$
- posterior $p_\theta(\mathbf{z}^{(n)} \mid \mathbf{x}^{(n)})$
- **variational distribution** $q_{\phi^{(n)}}(\mathbf{z}^{(n)})$

Learning with IID Data

Basic approach: introduce variational parameters $\phi^{(n)}$ for each datum and construct an overall lower bound:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{n=1}^{N} \log p_\theta(\mathbf{x}^{(n)}) \geq \frac{1}{N}\sum_{n=1}^{N} \mathsf{ELBO}(\theta, \phi^{(n)}, \mathbf{x}^{(n)})$$

$$\mathsf{ELBO}(\theta, \phi^{(n)}, \mathbf{x}^{(n)}) = \mathbb{E}_{q_{\phi^{(n)}}}\left[\log p_\theta(\mathbf{Z}^{(n)}, \mathbf{x}^{(n)}) - \log q_{\phi^{(n)}}(\mathbf{Z}^{(n)})\right]$$

Then optimize the lower bound with respect to all parameters. Compute:

$$\hat{\nabla}_{\phi^{(n)}} \approx \nabla_{\phi^{(n)}} \mathsf{ELBO}(\theta, \phi^{(n)}, \mathbf{x}^{(n)}), \quad n = 1, \ldots, N,$$

$$\hat{\nabla}_\theta \approx \nabla_\theta \frac{1}{N}\sum_{n=1}^{N} \mathsf{ELBO}(\theta, \phi^{(n)}, \mathbf{x}^{(n)})$$

Then update $\theta, \phi^{(1)}, \ldots, \phi^{(N)}$ using stochastic gradients.

Amortized Inference

The basic approach described above introduces a very large number of variational parameters and can be very slow for large data sets.

Amortized inference proposes to use a neural net to *predict* the variational parameters $\phi^{(n)}$ for datum $\mathbf{x}^{(n)}$, e.g.

$$q_\phi(\mathbf{z}^{(n)} \,|\, \mathbf{x}^{(n)}) = \mathcal{N}(\mathbf{z}^{(n)};\ g_\phi(\mathbf{x}^{(n)}), \tau^2 I)$$

- The function g_ϕ predicts the mean of the variational posterior approximation for datum $\mathbf{x}^{(n)}$. (We could also model the (co)variance as some function of $\mathbf{x}^{(n)}$.)
- This is called *amortization* because it shares information across data points for learning the variational approximations.

Amortized Inferences: VAEs

A common choice for g_ϕ is a multi-layer neural network, similar to f_θ, e.g.:

$$f_\theta(\mathbf{z}) = h_2(\mathbf{b}_2 + \mathbf{W}_2 \cdot h_1(\mathbf{b}_1 + \mathbf{W}_1 \mathbf{z}))$$

$$g_\phi(\mathbf{x}) = \mathbf{b}_4 + \mathbf{W}_4 \cdot h_3(\mathbf{b}_3 + \mathbf{W}_3 \mathbf{x})$$

(h_1, h_2, h_3 are elementwise non-linear functions)

Illustration: p and q graphical models

Illustration: “Auto-Encoder”

Example: Inference and Learning in Bernoulli VAE

Putting all the pieces together, stochastic variational inference and learning for a Bernoulli VAE would repeat the following for all n in some order:

$$\epsilon \sim \mathcal{N}(0, I) \qquad (\mathbf{z}^{(n)} = g_\phi(\mathbf{x}^{(n)}) + \tau\epsilon)$$

$$\begin{aligned}\hat{\nabla}_{\theta,\phi} = \nabla_{\theta,\phi}\Bigg\{ & \log \mathcal{N}(g_\phi(\mathbf{x}^{(n)}) + \tau\epsilon;\ 0, I) \\ & + \sum_{j=1}^{d} \log \text{Bernoulli}\Big(x_j^{(n)}; \big(f_\theta(g_\phi(\mathbf{x}^{(n)}) + \tau\epsilon\big)_j\Big) \\ & - \log \mathcal{N}\Big(g_\phi(\mathbf{x}^{(n)}) + \tau\epsilon;\ g_\phi(\mathbf{x}^{(n)}), \tau^2 I\Big) \Bigg\}\end{aligned}$$

$$(\theta, \phi) \leftarrow (\theta, \phi) + \alpha \cdot \hat{\nabla}_{\theta,\phi}$$

Bonus: Closed Form Entropy, Etc.

Bonus: Handling Some Terms in Closed Form

The ELBO can be decomposed into several terms with different computational properties:

$$\mathsf{ELBO}(\phi) = \mathbb{E}_{q_\phi}\left[\log \frac{p(Z,x)}{q_\phi(Z)}\right]$$
$$= \underbrace{\mathbb{E}_{q_\phi}[\log p(Z)]}_{\text{"cross entropy"}} + \underbrace{\mathbb{E}_{q_\phi}[\log p(x|Z)]}_{\text{"energy"}} - \underbrace{\mathbb{E}_{q_\phi}[\log q_\phi(Z)]}_{\text{"entropy"}}$$

With simple distributions (esp. Gaussians) the cross entropy and entropy terms can often be computed in closed form.

Example: Closed-Form Cross-Entropy

Example: $p(\mathbf{z})$ is a standard normal and q_ϕ is a diagonal Gaussian:

$$\begin{aligned} p(\mathbf{z}) &= \mathcal{N}(\mathbf{z}; 0, I) \\ q_\phi(\mathbf{z}) &= \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \mathsf{diag}(\boldsymbol{\sigma}^2)) \end{aligned} \implies \int q_\phi(\mathbf{z}) \log p(\mathbf{z}) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\sum_{j=1}^{d}(\mu_j^2 + \sigma_j^2)$$

When possible, it's usually (but not always) best to compute these terms and their gradients analytically, and only use Monte Carlo estimation for the energy term.

This is because lower variance gradient estimates will make the stochastic optimization converge faster.