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

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

Variational Auto-Encoder

Factor analysis model with non-linear mapping

\[
p(z) = \mathcal{N}(z; 0, I)
\]

\[
p(x_j | z) = \text{Bernoulli}(x_j; (f_\theta(z))_j), \quad j = 1, \ldots, d
\]

Example non-linear mapping:

\[
f_\theta(z) = h_2(b_2 + W_2 \cdot h_1(b_1 + W_1z))
\]

Exact inference and learning are \textit{intractable}.

Stochastic VI

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

\[
q_\phi(z) = \mathcal{N}(\mu, \text{diag}(\sigma^2)) \quad \phi = (\mu, \sigma)
\]

Stochastic optimization: repeatedly get \textbf{unbiased gradient estimate} \( \hat{\nabla}_\phi \), update \( \phi \):

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

\[
\phi \leftarrow \phi + \alpha \hat{\nabla}_\phi
\]

How to get \( \hat{\nabla}_\phi \)?
Gradient Estimation: Reparameterization

<table>
<thead>
<tr>
<th></th>
<th>Without reparameterization</th>
<th>With reparameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variational distribution</td>
<td>$q_\phi(z)$</td>
<td>$q_\phi(z)$</td>
</tr>
<tr>
<td>Sampling</td>
<td>$z \sim q_0(z)$</td>
<td>$\epsilon \sim q(\epsilon), z = T_\phi(\epsilon)$</td>
</tr>
<tr>
<td>ELBO</td>
<td>$\mathbb{E}_{q_0(z)} \left[ \log \frac{p(z,x)}{q_0(z</td>
<td>x)} \right]$</td>
</tr>
<tr>
<td>ELBO estimate</td>
<td>$\log \frac{p(z,x)}{q_0(z)}$, $z \sim q_0(z)$</td>
<td>$\log \frac{p(T_\phi(\epsilon),x)}{q_0(T_\phi(\epsilon),x)}$, $\epsilon \sim q(\epsilon)$</td>
</tr>
<tr>
<td>Gradient estimate</td>
<td>$\nabla_\phi \log \frac{p(z,x)}{q_0(z)}$, $z \sim q_0(z)$</td>
<td>$\nabla_\phi \log \frac{p(T_\phi(\epsilon),x)}{q_0(T_\phi(\epsilon),x)}$, $\epsilon \sim q(\epsilon)$</td>
</tr>
</tbody>
</table>

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

Aside: Reparameterization with Arbitrary Gaussians

Another choice would be to use a general Gaussian distribution:

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

This is a reparameterization with

$$q(\epsilon) = \mathcal{N}(\epsilon|0, I), \quad T_\phi(\epsilon) = \mu + L\epsilon \quad \phi = (L, \mu)$$

Example: Bernoulli VAE

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

$$p(z) = \mathcal{N}(z; 0, I)$$
$$p(x_j|z) = \text{Bernoulli}(x_j; (f_\theta(z))_j), \quad j = 1, \ldots, d$$
$$q_\phi(z) = \mathcal{N}(z; \mu, \text{diag}(\sigma^2))$$

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

Learning with Stochastic Variational Inference

The basic idea is to jointly maximize the ELBO with respect to model parameters $\theta$ and variational parameters $\phi$ by getting unbiased gradient estimates for both:

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

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

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

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

Learning with IID Data

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

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

- marginal likelihood $p_\theta(x^{(n)})$
- posterior $p_\theta(z^{(n)} | x^{(n)})$
- variational distribution $q_{\phi^{(n)}}(z^{(n)})$
Learning with IID Data

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

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

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

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

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

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

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 $x^{(n)}$, e.g.

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

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

Amortized Inferences: VAEs

A common choice for $g_{\phi}$ is a multi-layer neural network, similar to $f_{\theta}$, e.g.:

$$
f_{\theta}(z) = h_2(b_2 + W_2 \cdot h_1(b_1 + W_1z))
$$

$$
g_{\phi}(x) = b_4 + W_4 \cdot h_3(b_3 + W_3x)
$$

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

$$
\epsilon \sim \mathcal{N}(0, I) \\
\hat{\nabla}_{\theta,\phi} = \nabla_{\theta,\phi} \left\{ \log \mathcal{N}(g_{\phi}(x^{(n)}) + \tau \epsilon; 0, I) \\
+ \sum_{j=1}^{d} \log \text{Bernoulli}(x_{j}^{(n)}; (f_{\theta}(g_{\phi}(x^{(n)}) + \tau \epsilon))_{j}) \\
- \log \mathcal{N}(g_{\phi}(x^{(n)}) + \tau \epsilon; g_{\phi}(x^{(n)}), \tau^2 I) \right\} \\
(\theta, \phi) \leftarrow (\theta, \phi) + \alpha \cdot \hat{\nabla}_{\theta,\phi}
$$
The ELBO can be decomposed into several terms with different computational properties:

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

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

**Example: Closed-Form Cross-Entropy**

**Example:** $p(z)$ is a standard normal and $q_\phi$ is a diagonal Gaussian:

$$p(z) = \mathcal{N}(z; 0, I) \quad q_\phi(z) = \mathcal{N}(z; \mu, \text{diag}(\sigma^2))$$

$$\int q_\phi(z) \log p(z) = -\frac{d}{2} \log(2\pi) - \frac{1}{2} \sum_{j=1}^{d} (\mu_j^2 + \sigma_j^2)$$

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.