RBMs

Oliver K. Ernst

September 30, 2020

1 Theory

The objective function is the KL divergence:

$$\mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \int dx\; p(x) \ln \frac{p(x)}{\tilde{p}(x,\nu)} \tag{1}$$

where $p(x)$ is the true data distribution and $\tilde{p}(x,\nu)$ is the model distribution:

$$\begin{aligned} \tilde{p}(x,\nu) &= \frac{1}{Z(\nu)} \exp[-E(x,\nu)] \\ Z(\nu) &= \int dy\; \exp[-E(y,\nu)] \end{aligned} \tag{2}$$

for some energy function $E(x,\nu)$ with interactions ν. The gradients are

$$\begin{aligned} \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \nu} &= -\int dx\; p(x) \left(\frac{\tilde{p}(x,\nu)}{p(x)}\right) p(x) \left(\tilde{p}(x,\nu)\right)^{-2} \frac{\partial \tilde{p}(x,\nu)}{\partial \nu} = -\int dx\; \frac{p(x)}{\tilde{p}(x,\nu)} \frac{\partial \tilde{p}(x,\nu)}{\partial \nu} \\ \frac{\partial \tilde{p}(x,\nu)}{\partial \nu} &= -\frac{\partial E(x,\nu)}{\partial \nu} \tilde{p}(x,\nu) - \frac{1}{Z(\nu)^2} \exp[-E(x,\nu)] \frac{\partial Z(\nu)}{\partial \nu} \\ &= -\frac{\partial E(x,\nu)}{\partial \nu} \tilde{p}(x,\nu) + \frac{1}{Z(\nu)} \tilde{p}(x,\nu) \int dy\; \frac{\partial E(y,\nu)}{\partial \nu} \exp[-E(y,\nu)] \\ &= -\frac{\partial E(x,\nu)}{\partial \nu} \tilde{p}(x,\nu) + \tilde{p}(x,\nu) \int dy\; \frac{\partial E(y,\nu)}{\partial \nu} \tilde{p}(y,\nu) \\ &= -\frac{\partial E(x,\nu)}{\partial \nu} \tilde{p}(x,\nu) + \tilde{p}(x,\nu) \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{\tilde{p}} \\ \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \nu} &= \int dx\; \frac{p(x)}{\tilde{p}(x,\nu)} \frac{\partial E(x,\nu)}{\partial \nu} \tilde{p}(x,\nu) - \int dx\; \frac{p(x)}{\tilde{p}(x,\nu)} \tilde{p}(x,\nu) \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{\tilde{p}} \\ &= \int dx\; p(x) \frac{\partial E(x,\nu)}{\partial \nu} - \int dx\; p(x) \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{\tilde{p}} \\ &= \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{p} - \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{\tilde{p}} \end{aligned} \tag{3}$$

where the second integral is unity because p is normalized by definition.

$$\boxed{\frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \nu} = \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{p} - \left\langle \frac{\partial E}{\partial \nu} \right\rangle_{\tilde{p}}} \tag{4}$$

The first term is often called the *awake phase* moment, or moment under the *data distribution* p; the second term is often called the *asleep phase* moment, or moment under the *model distribution* $\tilde{p}$.

For the discrete case on a lattice with visible units $\boldsymbol{v}$ and hidden units $\boldsymbol{h}$:

$$\mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h}) \ln \frac{p(\boldsymbol{v}, \boldsymbol{h})}{\tilde{p}(\boldsymbol{v}, \boldsymbol{h})} \tag{5}$$

and a common energy function is:

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\boldsymbol{a}^\intercal \boldsymbol{v} - \boldsymbol{b}^\intercal \boldsymbol{h} - \boldsymbol{v}^\intercal W \boldsymbol{h} \tag{6}$$

for biases $\boldsymbol{a}, \boldsymbol{b}$ and weight matrix W, which play the role of ν. The gradients are:

$$\begin{aligned} \frac{\partial E}{\partial \boldsymbol{a}} &= -\boldsymbol{v} \\ \frac{\partial E}{\partial \boldsymbol{b}} &= -\boldsymbol{h} \\ \frac{\partial E}{\partial W} &= -\boldsymbol{v} \otimes \boldsymbol{h} \end{aligned} \tag{7}$$

leading to the gradients:

$$\boxed{\begin{aligned} \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \boldsymbol{a}} &= \langle \boldsymbol{v} \rangle_{\tilde{p}} - \langle \boldsymbol{v} \rangle_{p} \\ \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \boldsymbol{b}} &= \langle \boldsymbol{h} \rangle_{\tilde{p}} - \langle \boldsymbol{h} \rangle_{p} \\ \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial W} &= \langle \boldsymbol{v} \otimes \boldsymbol{h} \rangle_{\tilde{p}} - \langle \boldsymbol{v} \otimes \boldsymbol{h} \rangle_{p} \end{aligned}} \tag{8}$$

2 Implementation

2.1 Gradients

In practice, concerning the moments $\langle \dots \rangle$:

- In the continuous case, we usually cannot analytically perform the integral $\int dx$.
- In the discrete case, we usually cannot enumerate all possible states appearing in the sum $\sum_x$.

Therefore, these moments are estimated using batches. In the continuous case, let the batch be $X_{\tilde{p}}$ of size N, typically small ($N \sim 5 - 10$). For some observable $\chi(x)$:

$$\langle \chi(x) \rangle_{\tilde{p}} = \int dx\ \tilde{p}(x, \nu) \chi(x) \sim \frac{1}{N} \sum_{i=1}^{N} \chi(X_{\tilde{p},i}) \tag{9}$$

and similarly for moments with respect to p. In the discrete case, with the batch represented as $\boldsymbol{V}_{\tilde{p}}, \boldsymbol{H}_{\tilde{p}}$:

$$\langle \chi(\boldsymbol{v}, \boldsymbol{h}) \rangle_{\tilde{p}} = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}}\ \tilde{p}(\boldsymbol{v}, \boldsymbol{h}) \chi(\boldsymbol{v}, \boldsymbol{h}) \sim \frac{1}{N} \sum_{i=1}^{N} \chi(\boldsymbol{V}_{\tilde{p},i}, \boldsymbol{H}_{\tilde{p},i}) \tag{10}$$

The gradients are therefore estimated as:

$$\boxed{\begin{aligned} \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \boldsymbol{a}} &= \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{\tilde{p},i} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{p,i} \\ \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \boldsymbol{b}} &= \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{H}_{\tilde{p},i} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{H}_{p,i} \\ \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial W} &= \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{\tilde{p},i} \otimes \boldsymbol{H}_{\tilde{p},i} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{p,i} \otimes \boldsymbol{H}_{p,i} \end{aligned}} \tag{11}$$

2.2 Sampling

Where do such batches come from? They should be samples of the distributions p or $\tilde{p}$ as appropriate.

- For sampling the model distribution $\tilde{p}$, we have several options:
  - In the continuous case, without knowing further about $\tilde{p}(x)$, we can always perform Markov Chain Monte Carlo (MCMC) to sample the distributions (other sampling methods *may* be possible).
  - In the discrete case, we can use *Gibbs sampling*, which works by iteratively sampling $\tilde{p}(v_i|\boldsymbol{h})$ and $\tilde{p}(h_i|\boldsymbol{v})$. These are derived as follows:

$$\begin{aligned} \tilde{p}(h_i|\boldsymbol{v}) &= \frac{\tilde{p}(h_i, \boldsymbol{v})}{\tilde{p}(\boldsymbol{v})} \\ \tilde{p}(\boldsymbol{v}) &= \sum_{\boldsymbol{h}} \tilde{p}(\boldsymbol{v}, \boldsymbol{h}) \propto \exp[\boldsymbol{a}^\intercal \boldsymbol{v}] \\ \tilde{p}(\boldsymbol{h}, \boldsymbol{v}) &= \prod_i \tilde{p}(h_i, \boldsymbol{v}) \end{aligned} \tag{12}$$

    where the last line follows because in an RBM, the hidden variables are conditionally independent of all other hidden variables, and similarly for visible-visibles, then:

$$\tilde{p}(h_i, \boldsymbol{v}) \propto \exp[\boldsymbol{a}^\intercal \boldsymbol{v} + b_i h_i + \boldsymbol{v}^\intercal \mathrm{col}_i(W) h_i] \tag{13}$$

    then it follows:

$$\boxed{\tilde{p}(h_i|\boldsymbol{v}) \propto \exp[b_i h_i + \boldsymbol{v}^\intercal \mathrm{col}_i(W) h_i]} \tag{14}$$

    and similarly

$$\boxed{\tilde{p}(v_i|\boldsymbol{h}) \propto \exp[a_i v_i + v_i \mathrm{row}_i(W) \boldsymbol{h}]} \tag{15}$$

    In *contrastive divergence*, this procedure of iteratively sampling $\tilde{p}(v_i|\boldsymbol{h})$ and $\tilde{p}(h_i|\boldsymbol{v})$ is performed only a few times (or even only once!), starting from an initial data vector $\boldsymbol{v}$. This greatly improves computational efficiency; alternatively, you can run this sampler for a long time to let the *chain converge*.

    After sampling, we obtain the desired samples for the batch $\boldsymbol{V}_{\tilde{p},i}, \boldsymbol{H}_{\tilde{p},i}$. The sampling can be performed in *parallel* for higher efficiency, evaluating all N items in the batch $\boldsymbol{V}_{\tilde{p}}, \boldsymbol{H}_{\tilde{p}}$.

    In *persistent contrastive divergence*, we do not throw out the hidden states $\boldsymbol{V}_{\tilde{p}}, \boldsymbol{H}_{\tilde{p}}$ after one gradient step and restart from new data vectors $\boldsymbol{v}$. Instead, we keep these states, and use them in the next gradient step again after sampling $\tilde{p}(v_i|\boldsymbol{h})$ and $\tilde{p}(h_i|\boldsymbol{v})$ for a few more steps.
  - The samples $\boldsymbol{V}_p$ from the data distribution p are obvious; they are provided as training data. The samples $\boldsymbol{H}_p$ are obtained by *clamping the visible units to the data vectors* $\boldsymbol{V}_p$ and sampling *only* $\tilde{p}(h_i|\boldsymbol{v})$. Note that this is similarly possible with MCMC for the general continuous case; assuming we partition x into observed and latent variables, we can sample only the latent variables, keeping the observed variables clamped.

2.3 Objective function

It would be great if we could code up the objective function $\mathcal{D}_{\mathcal{KL}}$ onto a computer, but this is **not trivial**. Instead, we can make an important restriction:

Assume that our **optimizer only uses first-order gradients.**

If we make this restriction, we can consider the following objective function

$$\begin{aligned} S =& \boldsymbol{a}^\intercal \left(\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{\tilde{p},i} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{p,i} \right) \\ &+ \boldsymbol{b}^\intercal \left(\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{H}_{\tilde{p},i} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{H}_{p,i} \right) \\ &+ \left(\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{\tilde{p},i}^\intercal W \boldsymbol{H}_{\tilde{p},i} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{V}_{p,i}^\intercal W \boldsymbol{H}_{p,i} \right) \end{aligned} \tag{16}$$

which has the same **first-order** gradients as (11).

Note that the second-order gradients will obviously be incorrect!

This trick allows us to easily implement an RBM in TensorFlow.