Lab2 - Restricted Boltzmann Machine

Jiangnan Huang, You Zuo

November 12, 2020

Instructor: Caio Corro

1 Restricted Boltzmann Machines

1.1 Energy-based model

For an energy-based model with hidden states, we have its **joint distribution** written as:

$$p(x, z; \theta) = \frac{\exp(-e(x, z; \theta))}{\mathcal{Z}(\theta)} \tag{1}$$

where $\mathcal{Z}(\theta) = \sum_{x'} \sum_{z'} \exp(-e(x', z', \theta))$ is a normalizing factor called the partition function.

Besides, we introduce the **free energy**, which is defined as:

$$\mathcal{F}(x; \theta) = -\log \sum_{z} \exp(-e(x, z; \theta)) \tag{2}$$

we need this because this allows us to write the marginal probability of x as follows:

$$p(x; \theta) = \sum_{z} p(x, z; \theta) = \sum_{z} \frac{\exp(-e(x, z; \theta))}{\mathcal{Z}(\theta)} = \frac{\exp(-\mathcal{F}(x; \theta))}{\sum_{x'} \exp(-\mathcal{F}(x'; \theta))} \tag{3}$$

Usually, we learn the model by performing gradient descent algorithms on the empirical negative log-likelihood of the training data, but we can also write it as an expectation with the distribution of our observed data $\mathcal{D}$:

$$\begin{aligned} \hat{\theta} &= \arg\min_{\theta} \sum_{x \in \mathcal{D}} -\log p(x; \theta) \\ &= \arg\min_{\theta} \sum_{x \in \mathcal{D}} \frac{1}{|\mathcal{D}|} - \log p(x; \theta) \\ &= \arg\min_{\theta} \mathbb{E}_{\mathcal{D}}[-\log p(x; \theta)] \end{aligned} \tag{4}$$

From (3) and (4), for EBM with hidden variables we have the training goal:

$$\nabla_\theta \mathbb{E}_{\mathcal{D}}[-\log p(x;\theta)] = \mathbb{E}_{\mathcal{D}}[\nabla_\theta \mathcal{F}(x;\theta)] - \mathbb{E}_{p(x;\theta)}[\nabla_\theta \mathcal{F}(x;\theta)] \tag{5}$$

Usually, we implement methods of Monte Carlo sampling to approximate an expectation. But one last problem here is that, for the free energy $\mathcal{F}(x;\theta) = -\log \sum_z \exp(-e(x,z;\theta))$, it is intractable to compute for a complex form of latent variables z.

However, **Restricted Boltzmann Machine defines its energy function in a specific way which allows us to overcome the intractability for EBM with latent variables**.

1.2 Introduction of RBM

For a RBM, all the latent variables are supposed to be **binary**, and its energy function is defined as:

$$e(x,z;\theta) = -x^T u - z^T v - x^T w z \tag{6}$$

where w represents the weights connecting hidden and visible variables, and u, v are the offsets of the observed and hidden layers respectively. $\theta = \{w, u, v\}$ are parameters that we are going to learn.

Intuitively, it is a linear combination of functions of observed nodes x, hidden nodes z, and functions between x and z. For a RBM, it assumes that the functions among these nodes contribute linearly to the energy, and there is no quadratic term of x or z, because **the nodes (variables) of the same layer have no relation with each other (they are conditional independent, and that is also why it is called "restricted")**. Thus we can write the conditional probabilities as:

$$\begin{aligned} p(x|z) &= \prod_i P(x_i|z) \\ p(z|x) &= \prod_j p(z_j|x) \end{aligned} \tag{7}$$

It offers us the possibility to sample from $p(x|z)$ and $p(z|x)$ more efficiently from Gibbs sampling.

2 MCMC Space Exploration

In our lab exercise 2, we focused on a single Markov Chain when visualizing the sampling process and observed how it evolved.

For each fixed Markov Chain, it starts from a random point $x^{(0)} \sim p(x|z \sim \mathcal{B}(0.5))$, then continue sampling around this point. In the later process, every point is sampled based on the previous one:

$$\begin{aligned} x^{(t+1)} &\sim p(x|z^{(t)}) \\ z^{(t+1)} &\sim p(z|x^{(t+1)}) \end{aligned} \tag{8}$$

n

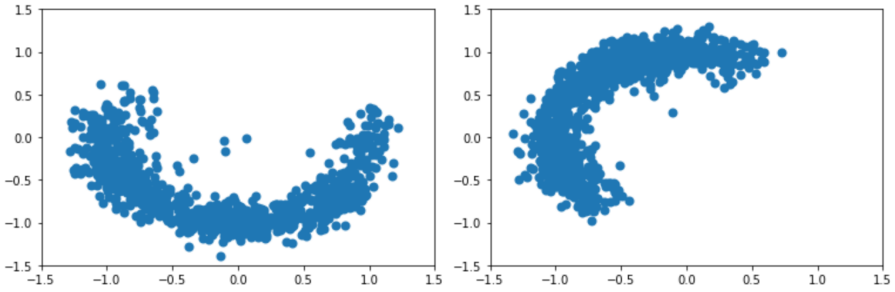

Figure 1: two MC sample spaces

We can find from those single Markov Chains that, usually the first sampling point is not good, and each point is highly related and close to the previous one.

3 Implementation

In practice, if we need a large number of samples from the RBM model, there would be some tricks as follows:

1. implement multiple independent Markov Chains and start sampling in parallel, then pick samples from different MCs.
2. burn-in: do not use the first samples.
3. skipping: use only one-in-s samples to ensure the Independence among points sampled next to each other.