Deep Probabilistic Generative Models - Restricted Boltzmann Machine

M2 ANO BEHIDJ Ramzi
M2 AI JEMLI Nour

October 2021

Contents

1 Introduction

Invented in 1985 by the Godfather of Deep Learning, Boltzmann Machine is a generative unsupervised model used for making Inference about data never seen using a learning from a probability distribution of the original dataset. Restricted Boltzmann machine is an algorithm useful for dimensionality reduction, classification, regression, collaborative filtering, feature learning and topic modeling.
Boltzmann Machine present two layers with a bidirectional connection:

- Input layers: the visible layer
- Hidden layer

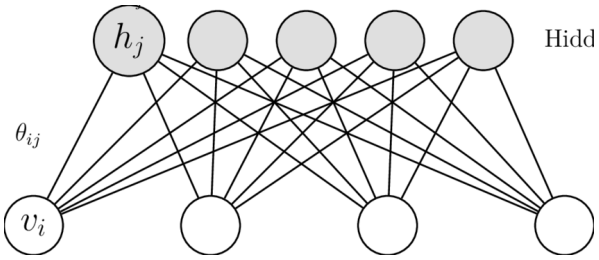

Figure 1 – Restricted Boltzman machine architecture

As we can see above, RBMs are shallow, two-layer neural nets that constitute the building blocks of deep-belief networks. The first layer of the RBM is called the visible, or input, layer, and the second is the hidden layer. All the layer are connected to each other's, not only to different types of layers but to the same ones. This means everything is connected to everything so it can learn from the inputs.

In this lab we are gonna see why are Restricted Boltzmann Machines interesting as Energy based models and observation when using a single Markov Chain to sample from the model.

2 Restricted Boltzmann Machines

2.1 Energy based model

First of all, to define energy, we can properly say that it is not associated with deep learning. Nonetheless, energy can be defined as a property of physics, a quantitive one. In order to measure one model's quality, we can use energy. in fact, some deep learning architectures actually rely on it. There is one purpose to deep learning models, it is to facilitate and encode dependencies between variables. This process surely helps as a measurement of one model's quality.

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 $\mathcal{Z}(\theta) = \sum_{x'} \sum_{z'} \exp\left(-e\left(x', z', \theta\right)\right)$ 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:

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

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 energy based models with hidden variables we have the training goal:

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

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 energy based models with latent variables.

2.2 Main idea

The relation between the energy-based model and the Bolzman Machine is The restricted Bolzman Machine which is a probabilistic unsupervised generative learning algorithm. RBM Is undirected and has just two layers and what makes it different to the Bolzman Machine is that it's visible nodes arent's connected to each other and same for the hidden nodes, It's restrict the intralayer connection and with it's also called an asymmetrical bipartite graph.
The Restricted Bolzman Machine consists of m visible units $v = (v1, ..., v_m)$ to represent observable data and n hidden units $h = (h1, ..., h_n)$ to capture dependencies between observed variables. The energy function of the RBM is :

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} h_i v_j - \sum_{j=1}^{m} b_j v_j - \sum_{i=1}^{n} c_i h_i \tag{6}$$

where w represents the weights connecting hidden and visible variables. Intuitively, it is a linear combination of functions of observed nodes x, hidden nodes z, and functions between x and z. In terms of probability this means that the hidden variables are independent given the state of the visible variables and vice versa :

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

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

RBM don't use gradient descent or backpropagation. They use a contrastive divergence. Parameters of the visible nodes are randomly generated first and used to generate the hidden nodes. Which use same parameters to reconstruct visible nodes. And the parameter used to reconstruct the visible nodes are the same. But the generated nodes aren't the same because they're not connected to each others

3 Markov Chain Monte Carlo space exploration

Our focus here is 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 $x^{(0)} \sim p(x \mid z \sim \mathcal{B}(0.5)\ )$, then continue sampling around this point. In the later process, every point is sampled based on the previous one.
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, some continuous sampling points may form a "cluster".

4 Advice when sampling from the model

In practice, when there is need to sample many times from a model, one can consider other algorithms. The first approaches for problems with large numbers of parameters is to use the Hamiltonian. In fact, these methods perform very well at large dimensions but require analytic derivatives with respect to the parameters for speed performance. Next, when the number of parameters is large, Gibbs sampling is worth considering. Another method is used quite a bit in hierarchical inferences, as well as in problems where the dimension is low. It consists in taking a sampling from the approximate function, which by assumption was easy and then re-weight the sampling using the ratio of the true function to the approximate function as a weight or probability.