# Restricted Boltzmann machine

***Pradip Das, Tata Digital Pvt Ltd***
pradipdas_2021@iitkalumni.org

---

A restricted Boltzmann machine (RBM), also known as a restricted Sherrington–Kirkpatrick model with an external field or restricted stochastic Ising–Lenz–Little model, is an artificial neural network of the stochastic type that has the capability to acquire knowledge about a probability distribution across its input data.

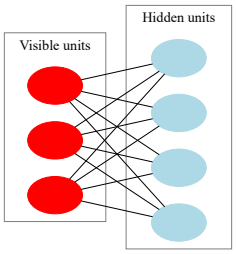


**Figure 1:** Restricted Boltzmann Machine with three visible units and four hidden units (no bias units)

Let $v$ and $h$ are the visible and hidden units, then we can calculate $h$ from $v$ $(v \rightarrow h)$ and also $v$ from $h$ $(h \rightarrow v)$.

Here are key features and concepts associated with Restricted Boltzmann Machines:

- **Architecture:** An RBM consists of two layers of nodes: a visible layer and a hidden layer. Nodes within each layer are fully connected, but there are no connections between nodes within the same layer. This "restricted" connectivity simplifies the training algorithm.
- **Stochastic Nodes:** RBM nodes are stochastic, meaning they have probabilistic activation states. The states of the visible and hidden nodes are binary, typically representing on/off or $0/1$ states. The probability of activation is determined by the sigmoid function.
- **Energy-Based Model:** RBM is an energy-based model where the model is trained to assign lower energy to observed data and higher energy to unobserved or generated data. The energy of a particular configuration of visible and hidden nodes is determined by the weights and biases in the network.
- **Learning Algorithm:** The training of an RBM involves adjusting the weights and biases to minimize the difference between the observed data's energy and the generated data's energy. Contrastive Divergence (CD) is a commonly used algorithm for training RBMs. CD approximates the gradient of the log-likelihood function, making the learning process more tractable.

- **Generative Model:** RBMs are generative models, meaning they can generate new samples that resemble the training data. By sampling from the joint distribution of visible and hidden units, RBMs can generate new instances.
- **Pre-training for Deep Learning:** RBMs have been used as a pre-training step for deep neural networks. The idea is to pre-train each layer of a deep network as a stack of RBMs before fine-tuning the entire network on a specific task. This pre-training helps initialize the network with useful features.
- **Sparse Representations:** RBMs tend to learn sparse representations of data, which can be beneficial for capturing hierarchical and abstract features.

## Bernoulli RBM

In Bernoulli RBM all the nodes in visible and hidden unit takes only binary value, i.e., $v_i,\ h_j \in \{0, 1\}$, where $v_i$ is the $i^{th}$ node in visible unit and $h_j$ is the $j^{th}$ node in hidden unit.
Let, visible unit have $D$ nodes, hidden unit have $M$ nodes and $W = [w_{ij}]$ is the weight between visible and hidden unit. Then to calculate $h$ from $v$, $\mathcal{P}(h_j = 1|v) = \sigma(\sum_{i=1}^{D} w_{ij}v_i + c_j)$ and to calculate $v$ from $h$, $\mathcal{P}(v_i = 1|h) = \sigma(\sum_{j=1}^{M} w_{ij}h_j + b_i)$. In vector form those equation can be written as,

$$\begin{aligned} \mathcal{P}(h = 1|v) &= \sigma(W^t v + c) \\ \mathcal{P}(v = 1|h) &= \sigma(Wh + b) \end{aligned}$$

We only get the probabilities, what if we want the actual $h$? $h$ is random variable so it has no explicit fixed value. We only have probability distribution fro $h$ given $v$.

*Relaxing the Bernoulli constrain:* When going from $h$ to $v$ we simply use $\mathcal{P}(h = 1|v)$ itself as a input. i.e.,

$$\begin{aligned} \tilde{h} &= \mathcal{P}(h = 1|v) = \sigma(W^t v + c) \\ \tilde{v} &= \mathcal{P}(v = 1|\tilde{h}) = \sigma(W\tilde{h} + b) \end{aligned}$$

## Boltzmann Machine

What if we create a neural network without imposing any structure on it? "Everything connected to everything" - Boltzmann Machines.
A Boltzmann machine, named after Ludwig Boltzmann, is a stochastic spin-glass model that includes an external field, specifically the Sherrington–Kirkpatrick model, and is a form of the stochastic Ising model. It is a statistical physics concept commonly employed in the realm of cognitive science and is also categorized as a Markov random field.
The theoretical appeal of Boltzmann machines lies in their training algorithm, which is both localized and Hebbian in nature (trained via Hebb's rule). Additionally, their parallelism and the similarity of their dynamics to basic physical processes add to their intrigue. Although Boltzmann machines with unconstrained connections haven't proven useful in practical machine learning or inference tasks, when connections are properly constrained, efficient learning can render them practical for various problems.

These machines derive their name from the Boltzmann distribution utilized within their sampling function. Geoffrey Hinton, Terry Sejnowski, and Yann LeCun notably popularized and advocated for Boltzmann machines within cognitive science and machine learning communities. Within machine learning, this category of models is more broadly termed "energy-based models" (EBM), as the learning task is defined based on Hamiltonians of spin glasses.

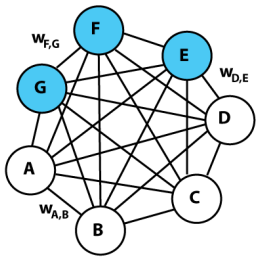


**Figure 2:** A graphical representation of a Boltzmann machine with a few weights labeled. Each undirected edge represents dependency and is weighted with weight $w_{ij}$. In this example there are 3 hidden units (blue) and 4 visible units (white).

Energy of a Boltzmann machine is given by

$$E = -\left(\sum_{i,j} w_{i,j} s_i s_j + \sum_i b_i s_i\right)$$

$w_{ij}$ is the connection strength between unit $j$ and unit $i$, $s_i$ is the state and $s_i \in \{0, 1\}$, of unit $i$, $b_i$ is the bias of unit $i$ in the global energy function. Goal of Boltzmann machine is to find some equilibrium. The concept of "equilibrium of energy" refers to a state where the network has settled into a stable configuration that represents a balance between the visible and hidden units. The term "energy" in this context is borrowed from statistical physics, and it's used as a metaphor to describe the state of the Boltzmann Machine.

***Training of Boltzmann machine***: The units within the Boltzmann machine are divided into 'visible' units referred to as $v$ and 'hidden' units designated as $h$. The visible units receive input from the 'environment', i.e. the training set constitutes binary vectors from the $v$ set. The distribution across this training set is denoted $P^+(v)$. The distribution over global states converges as the Boltzmann machine reaches thermal equilibrium. We denote this distribution, after we marginalize it over the hidden units, as $P^-(v)$.
Our goal is to approximate the "real" distribution $P^+(v)$ using the $P^-(v)$ produced by the machine. The similarity of the two distributions is measured by the KL divergence, $G$,

$$G = \sum_v P^+(v) \ln\left(\frac{P^+(v)}{P^-(v)}\right)$$

It is very difficult to train, this brings us to Restricted Boltzmann Machine.

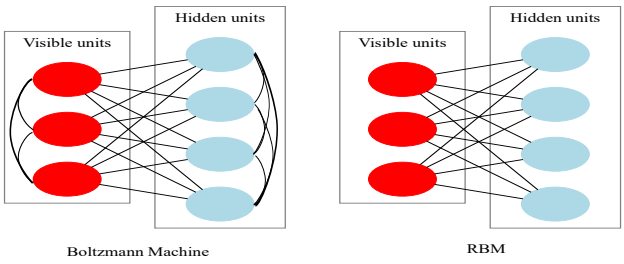


**Figure 3:** Boltzmann Machine and Restricted Boltzmann Mechine

## Energy of RBM

In scalar form the energy of RBM is defined by

$$E(v,h) = -\left(\sum_{i=1}^{D}\sum_{j=1}^{M} w_{ij}v_ih_j + \sum_{i=1}^{D} b_iv_i + \sum_{j=1}^{M} c_jh_j\right)$$

where $w_ij$ is weight between $v_i$ and $h_j$ and $b_i$, $c_j$ are bias. In vector form

$$E(v,h) = -\left(v^TWh + b^Tv + c^Th\right)$$

Just like a original Boltzmann Machine's energy function. Our goal is to find some equilibrium energy given our data set. This energy function leads us to a probability model and this probability model leads us to the simple neural network.

**Notion of energy equilibrium:**

In the context of Restricted Boltzmann Machines (RBMs), the notion of energy equilibrium relates to the equilibrium state of the system with respect to its energy function. RBMs are probabilistic graphical models that aim to capture complex relationships in data by learning a joint probability distribution over observed and hidden variables.
In RBMs, the equilibrium state is reached when the system's energy function reaches a stable point. This stable point typically corresponds to a state where the energy of the RBM is minimized or, equivalently, the probability distribution assigned by the RBM to observed data is maximally accurate.
During training, RBMs typically employ techniques such as Contrastive Divergence (CD) or Persistent Contrastive Divergence (PCD) to iteratively adjust the parameters (weights and biases) of the model to move towards this equilibrium state. These training algorithms aim to minimize the difference between the model's generated distribution and the true distribution of the observed data.
Once the RBM is trained, it can be used for tasks such as generating new samples from the learned distribution or performing inference on the hidden variables given observed data. At equilibrium, the RBM's energy function stabilizes, indicating that the model has learned to capture the underlying statistical dependencies in the data.
In summary, the notion of energy equilibrium in RBMs refers to the state where the energy function of the model stabilizes, indicating that the model has learned to accurately represent the data distribution.

Now using energy function we define a probability distribution.

$$\mathcal{P}(v,h) \propto e^{-E(v,h)}$$

$$\mathcal{P}(v,h) = \frac{1}{z}e^{-E(v,h)}$$

where $z = \sum_v \sum_h e^{-E(v,h)}$, is called partition function so that $\sum_v \sum_h \mathcal{P}(v,h) = 1$.

Why should we define probability in this way? –Answer: *Statistical mechanics*.
General outline is that $p_i$ is the probability that the system is in a microstate with energy $E_i$. Then,

$$p_i \propto e^{-e_i/kT}$$

$$p_i = \frac{1}{z}e^{-e_i/kT}$$

where $T$ is temperature, $k$ Boltzmann constant and $z = \sum e^{-E_i/kT}$. This distribution known as Boltzmann distribution or Gibbs distribution
– Same format as Boltzmann Machine.

***Energy function to probability distribution:***

By Bayes Rule

$$\mathcal{P}(v|h) = \mathcal{P}(v,h)/\mathcal{P}(h)$$
$$\mathcal{P}(h|v) = \mathcal{P}(v,h)/\mathcal{P}(v)$$

we know numerator but what s denominator?

$$\mathcal{P}(v) = \sum_h \mathcal{P}(v,h),\ \mathcal{P}(h) = \sum_v \mathcal{P}(v,h)$$

Now,

$$\mathcal{P}(v,h) = \frac{1}{z}\exp\left(vW^Th + b^Tv + c^Th\right)$$

So,

$$\mathcal{P}(h|v) = \frac{\exp\left(vW^Th + b^Tv + c^Th\right)}{\sum_h \exp\left(vW^Th + b^Tv + c^Th\right)}$$

Denominator here is nothing more than the normalizing constant. So let just ignore it and denote as $z'$.

$$\mathcal{P}(h|v) = \frac{1}{z'}\exp\left(vW^Th + b^Tv + c^Th\right)$$

Then,

$$
\begin{aligned}
\mathcal{P}(h|v) &= \frac{1}{z'} \exp\left(\sum_{i=1}^{D}\sum_{j=1}^{M} w_{ij}v_i h_j + \sum_{i=1}^{D} b_i v_i + \sum_{j=1}^{M} c_j h_j\right) \\
&= \frac{1}{z'} \exp\left(\sum_{i=1}^{D} b_i v_i\right) \prod_{j=1}^{M} \exp\left(\sum_{i=1}^{D} w_{ij}v_i h_j + c_j h_j\right) \\
&= \frac{1}{z''} \prod_{j=1}^{M} \exp\left(\sum_{i=1}^{D} w_{ij}v_i h_j + c_j h_j\right) \\
&= \frac{1}{z''} \prod_{j=1}^{M} \exp\left(h_j\left(\sum_{i=1}^{D} w_{ij}v_i + c_j\right)\right)
\end{aligned}
$$

Each of $j$ term is independent ($\mathcal{P}(A, B) = \mathcal{P}(A) \cdot \mathcal{P}(B)$) and since $z$ is just normalizing constant with respect to $h$ it doesn't matter. We can see that each $\mathcal{P}(h_j|v)$ is independent of other. So for a single one,

$$
\mathcal{P}(h_j|v) = \frac{1}{z'''} \exp\left(h_j\left(\sum_{i=1}^{D} w_{ij}v_i + c_j\right)\right)
$$

As $h_j \sim Bern(0, 1)$ so,

$$
\begin{aligned}
\mathcal{P}(h_j = 1|v) + \mathcal{P}(h_j = 0|v) &= 1 \\
\implies z''' &= 1 + \exp\left(\sum_{i=1}^{D} w_{ij}v_i + c_j\right)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
\mathcal{P}(h_j = 1|v) &= \frac{\exp\left(\sum_{i=1}^{D} w_{ij}v_i + c_j\right)}{1 + \exp\left(\sum_{i=1}^{D} w_{ij}v_i + c_j\right)} \\
&= \sigma\left(\sum_{i=1}^{D} w_{ij}v_i + c_j\right)
\end{aligned}
$$

Therefore,

$$
\mathcal{P}(h = 1|v) = \sigma\left(W^T v + c\right)
$$

## RBM Training

We want $W,\ b,\ c$ to maximize the likelihood $\mathcal{P}(v)$.
*maximize* $\mathcal{P}(v)$ w.r.t. $W,\ b,\ c$. Therefore,

$$
W, b, c = \arg\max_{W,b,c} \mathcal{P}(v; W, b, c)
$$

Maximizing $\mathcal{P}(v; W, b, c)$ is same as maximizing $\log \mathcal{P}(v; W, b, c)$ so,

$$W, b, c = \arg\max_{W,b,c} \log \mathcal{P}(v; W, b, c)$$

Here $\mathcal{P}(v) = \frac{1}{z} \sum_h e^{E(v,h)}$ is intractable. So we need to define something for which $\mathcal{P}(v)$ is to be tractable.

***Free Energy:***

$$F(v) = -\log \sum_h e^{E(v,h)}$$
$$\implies e^{-F(v)} = \sum_h e^{E(v,h)}$$

Thus

$$\mathcal{P}(v) = \frac{1}{z} e^{-F(v)},\ z = \sum_v e^{-F(v)}$$

***Maximum Likelihood Estimation:***
Let pretend $\mathcal{P}(v)$ is tractable and gradient descent is possible. What will the update look like? Find derivative w.r.t. $W_{ij}, b_i, c_j$.

$$\begin{aligned}
\frac{\partial \log \mathcal{P}(v)}{\partial \theta} &= \frac{\partial}{\partial \theta} \left( log \left( \frac{e^{-F(v)}}{z} \right) \right) \\
&= \frac{\partial}{\partial \theta} \left( -F(v) - \log(z) \right) \\
&= -\frac{\partial F(v)}{\partial \theta} - \frac{\partial \log z}{\partial \theta} \\
&= -\frac{\partial F(v)}{\partial \theta} - \frac{1}{z} \frac{\partial z}{\partial \theta} \\
&= -\frac{\partial F(v)}{\partial \theta} - \frac{1}{z} \frac{\partial}{\partial \theta} \left( \sum_{v'} e^{-F(v')} \right) \\
&= -\frac{\partial F(v)}{\partial \theta} + \sum_{v'} \frac{1}{z} e^{-F(v')} \frac{\partial F(v')}{\partial \theta} \\
&= -\frac{\partial F(v)}{\partial \theta} + \sum_{v'} \mathcal{P}(v') \frac{\partial F(v')}{\partial \theta}
\end{aligned}$$

So,

$$-\frac{\partial \log \mathcal{P}(v)}{\partial \theta} = \frac{\partial F(v)}{\partial \theta} - \mathbf{E} \left( \frac{\partial F(v')}{\partial \theta} \right)$$

How to sample? Markov chain Monte Carlo (MCMC) specifically, Gibbs sampling.

$$
\begin{aligned}
Given &: v_1 \\
Find &: \mathcal{P}(h|v_1) \\
Sample &: h_1 \sim \mathcal{P}(h|v_1) \\
Find &: \mathcal{P}(v|h_1) \\
sample &: v_2 \sim \mathcal{P}(v|h_1) \\
Find &: \mathcal{P}(h|v_2) \\
Sample &: h_2 \sim \mathcal{P}(h|v_2) \\
&.... \\
\end{aligned}
$$

$v_\infty$ is a sample from $\mathcal{P}(v)$.
So in practice we do $k\ (= 1)$ step. This is known as Contrastive Divergence (CD-k).
$v \to \mathcal{P}(h|v) \to h' \sim \mathcal{P}(h|v) \to \mathcal{P}(v|h') \to v' \sim \mathcal{P}(v|h')$.

$$
-\frac{\partial \log \mathcal{P}(v)}{\partial \theta} \approx \frac{\partial F(v)}{\partial \theta} - \frac{\partial F(v')}{\partial \theta}
$$

So loss $L = F(v) - F(v')$, as we don't actually have to calculate the gradient because Theano and Tensorflow have automatic differentiation.

***Fake Loss:***

$$
\frac{\partial(-\log \mathcal{P}(v))}{\partial \theta} \approx \frac{\partial L}{\partial \theta} = \frac{\partial F(v)}{\partial \theta} - \frac{\partial F(v')}{\partial \theta}
$$

We don't express gradient explicitly in Theano and Tensorflow, therefore we only want $L$ itself (the integral of derivative).

$$
\begin{aligned}
&\int \frac{\partial L}{\partial \theta} d\theta = \int \frac{\partial F(v)}{\partial \theta} d\theta - \int \frac{\partial F(v')}{\partial \theta} d\theta \\
&\implies L = F(v) - F(v')
\end{aligned}
$$

## Uses of RBM in e-commerce

RBMs can be utilized in various ways within ecommerce projects, particularly for tasks involving recommendation systems, customer segmentation, and anomaly detection. Here's how RBMs can be applied:

- **Recommendation Systems:**

  RBMs can be employed to model user-item interactions in ecommerce platforms. By learning the joint probability distribution of users and items, RBMs can generate recommendations for users based on their preferences and past interactions.

  RBMs can capture complex patterns in user behavior, such as sequential interactions and latent preferences, leading to more accurate recommendations compared to traditional collaborative filtering methods.

- **Customer Segmentation:**

  RBMs can be used for clustering and segmenting customers based on their browsing history, purchase behavior, demographic information, etc.

  By learning a distributed representation of customers and products, RBMs can identify meaningful clusters of customers with similar preferences or characteristics. These segments can then be used for targeted marketing campaigns, personalized recommendations, and product customization.

- **Anomaly Detection:**

  RBMs can detect anomalies or unusual patterns in user behavior, such as fraudulent transactions, abnormal browsing activities, or sudden changes in purchase behavior.

  By learning the normal distribution of user interactions, RBMs can flag instances that deviate significantly from the learned patterns, helping ecommerce platforms to detect and mitigate potential threats or issues.

- **Feature Learning and Representation:**

  RBMs can be used to learn low-dimensional representations of high-dimensional data, such as product images, user reviews, or textual descriptions.

  These learned representations can capture meaningful features and characteristics of the data, enabling more efficient processing, classification, and retrieval tasks within ecommerce systems.

- **Session-based Recommendation:**

  RBMs can be used to model sequential user behavior within ecommerce sessions, capturing the temporal dynamics of user interactions.

  By learning the transition probabilities between different items or actions within sessions, RBMs can make personalized recommendations based on the current context and user's historical behavior.

- **Dynamic Pricing:**

  RBMs can be applied to model demand patterns and price sensitivities of customers in ecommerce settings.

  By analyzing historical transaction data and external factors (e.g., competitor prices, seasonality), RBMs can help optimize pricing strategies to maximize revenue and profitability.

Incorporating RBMs into ecommerce projects requires data preprocessing, model training, evaluation, and deployment stages, along with considerations for scalability, interpretability, and privacy concerns. Collaborative filtering methods, hybrid models combining RBMs with other techniques (e.g., matrix factorization, deep learning), and ensemble approaches can further enhance the effectiveness of RBM-based solutions in ecommerce applications.

## Advantages and Disadvantages

RBMs have several advantages and disadvantages, which are important to consider when choosing them for a particular task. Here's a breakdown of the advantages and disadvantages:

- **Advantages:**
  - **Feature Learning:** RBMs are capable of automatically learning useful representations or features from raw data without requiring manual feature engineering. This can be particularly advantageous when dealing with high-dimensional and complex data such as images, text, or sequential data.
  - **Generative Modeling:** RBMs can generate new samples from the learned probability distribution, making them useful for tasks such as data generation, data augmentation, and denoising.
  - **Non-linearity:** RBMs can capture non-linear relationships between variables, allowing them to model complex dependencies in the data.
  - **Unsupervised Learning:** RBMs can be trained in an unsupervised manner, meaning they can learn from unlabeled data, making them suitable for tasks where labeled data is scarce or expensive to obtain.
  - **Ability to Handle Missing Values:** RBMs can handle missing values in the input data gracefully, allowing them to still learn meaningful representations even when some data points are incomplete.
- **Disadvantages:**
  - **Computational Complexity:** Training RBMs can be computationally intensive, especially for large datasets and deep architectures. This can lead to longer training times and higher computational resource requirements.
  - **Interpretability:** RBMs can learn complex and distributed representations of the data, which can make it difficult to interpret and understand the learned features.
  - **Sensitivity to Hyperparameters:** RBMs have several hyperparameters that need to be carefully tuned, such as learning rate, batch size, and the number of hidden units. Improper tuning of these hyperparameters can lead to suboptimal performance or slow convergence during training.
  - **Sample Generation Quality:** While RBMs can generate new samples from the learned distribution, the quality of generated samples may not always be as high as desired, especially for highly structured data such as images or text.
  - **Limited Scalability:** Despite being parallelizable, RBMs may not scale well to extremely large datasets or architectures due to memory and computational constraints.