Restricted Boltzmann Machines

Boltzmann Machine(BM)

- A Boltzmann machine extends a stochastic Hopfield network to include hidden units. It has binary (0 or 1) visible vector unit x and hidden (latent) vector unit h that detects features in the visible vector x.
- The model is parametrised in matrix form by U, V, W, b, c, where the visible-visible weights are U, the hidden-hidden weights are V, and W are the visible-hidden weights, all symmetric without self-connections. The visible units have biases b, and the hidden units have biases c.
- Each joint configuration of the visible and hidden units has an associated energy, defined in matrix form by:

$$E(v,h) = -\frac{1}{2}v^\top Uv - \frac{1}{2}h^\top Vh - v^\top Wh - c^\top v - b^\top h$$

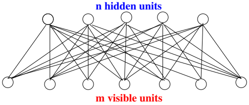
- Without hidden units, E is as in a Hopfield network.
- Learning with BM is extremely difficult and impractical.

Restricted Boltzmann Machines (RBM)

- An RBM is a BM with a bi-partite graph of m visible and n hidden units, i.e., no connections between visible units or between hidden units. What are the maximal cliques?
- The energy has, by Hammersley-Clifford theorem, parameters $\theta \in \Theta := \{w_{ij}, b_j, c_i : 1 \leq j \leq m, 1 \leq i \leq n\}$:

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

- Assume each training item $x_k \in D$, $(k = 1, \ldots, \ell)$, gives a B/W pixel image and items are i.i.d random variables drawn from a distribution q on m nodes.
- **Unsupervised learning:** An RBM can learn a distribution p to approximate q on $D \subset S = \{0, 1\}^m$.



Maximising log likelihood

- An asymmetric measure of difference between q and p is given by **Kullback-Leiber divergence** or the **relative entropy** of q wrt p given for a finite state space S by:

$$\mathrm{KL}(q\|p) = \sum_{x\in S} q(x)\ln\frac{q(x)}{p(x)} = \sum_{x\in S} q(x)\ln q(x) - \sum_{x\in S} q(x)\ln p(x)$$

- $\mathrm{KL}(q\|p)$ is non-negative and is zero iff $p = q$.
- Only the last term depends on p, thus on the parameters.
- Therefore, minimising $\mathrm{KL}(q\|p)$ corresponds to maximising the likelihood of p for training items.
- Thus, learning aims to determine all parameters $\theta \in \Theta$ to maximise the likelihood wrt D defined by:

$$L(\theta|D) = \prod_{k=1}^{\ell} p(x_k|\theta), \quad \text{or maximising its log likelihood:}$$

$$\ln L(\theta|D) = \ln \prod_{k=1}^{\ell} p(x_k|\theta) = \sum_{k=1}^{\ell} \ln p(x_k|\theta)$$

Gradient Ascent

- Since we cannot analytically solve the maximisation for an RBM, we use the method of gradient ascent.
- **Idea.** Find $(\theta_1, \ldots, \theta_p)$ for the maximum value of $f : \mathbb{R}^p \to \mathbb{R} : (\theta_1, \ldots, \theta_p) \mapsto f(\theta_1, \ldots, \theta_p)$, as follows:
- Start with some $\theta_i^{(0)}$ and for each i obtain increasingly better approximations to the θ_i value for the maximum of f:

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \alpha \frac{\partial f}{\partial \theta_i}(\theta_i^{(t)}), \qquad \text{with } \alpha > 0 \text{ a constant}$$

- For RBM, start from an initial value $\theta^{(0)}$ for $\theta \in \Theta$. Let

$$\theta^{(t+1)} = \theta^{(t)} + \alpha \frac{\partial}{\partial \theta} \left(\sum_{k=1}^{\ell} \ln p(x_k | \theta^{(t)}) \right) - \lambda \theta^{(t)} + \nu \Delta \theta^{(t-1)} \quad (1)$$

  where $\Delta\theta^{(t)} = \theta^{(t+1)} - \theta^{(t)}$ and $\alpha > 0$ is the learning rate.
- The last two terms are added to optimise the algorithm:
- $-\lambda\theta^{(t)}$ is **the decay weight**, with $\lambda > 0$ a constant.
- $\nu\Delta\theta^{(t-1)}$ is the **momentum**, with $\nu > 0$ a constant.

RBM probability distribution

- To use gradient ascent, we need to compute $p(v)$ and $\partial \ln p(v)/\partial\theta$, where v is any state of the visible units.
- As in any energy based model, the joint distribution of visible and hidden units (v, h) is given by

$$p(v, h) = \frac{e^{-E(v,h)}}{Z}, \text{ with } Z = \sum_{v\in\{0,1\}^m} \sum_{h\in\{0,1\}^n} e^{-E(v,h)}$$

- Since the only connections are between a visible and a hidden unit, the conditional probability distributions are:

$$p(h|v) = \prod_{i=1}^{n} p(h_i|v), \qquad p(v|h) = \prod_{j=1}^{m} p(v_j|h).$$

- The marginal distribution of visible units is given by

$$p(v) = \sum_h p(v, h) = \frac{1}{Z}\sum_h e^{-E(v,h)}$$

- This distribution can be computed as product of factors.

Computation of log-likelihood

- Therefore, the log-likelihood is computed as:

$$\begin{aligned} \ln\; p(x|\theta) &= \ln \frac{1}{Z}\sum_h e^{-E(x,h)} \\ &= \ln \sum_h e^{-E(x,h)} - \ln \sum_{x,h} e^{-E(x,h)}, \end{aligned} \tag{2}$$

  where θ is assumed to be one of the parameters, i.e., w_{ij}, b_j, c_i, of the model.
- To compute the derivative of log likelihood we need the following:
- $$p(h|v) = \frac{p(v,h)}{p(v)} = \frac{\frac{1}{Z}e^{-E(v,h)}}{\frac{1}{Z}\sum_h e^{-E(v,h)}} = \frac{e^{-E(v,h)}}{\sum_h e^{-E(v,h)}}$$
- We can now proceed as follows.

Computation of log-likelihood gradient (I)

$$
\begin{aligned}
&\frac{\partial}{\partial\theta}\left(\ln\ p(v|\theta)\right)\\
&= \frac{\partial}{\partial\theta}\left(\ln \textstyle\sum_h e^{-E(v,h)}\right) - \frac{\partial}{\partial\theta}\left(\ln \textstyle\sum_{v,h} e^{-E(v,h)}\right)\\
&= -\frac{1}{\sum_h e^{-E(v,h)}} \textstyle\sum_h e^{-E(v,h)} \frac{\partial E(v,h)}{\partial\theta} + \frac{1}{\sum_{v,h} e^{-E(v,h)}} \textstyle\sum_{v,h} e^{-E(v,h)} \frac{\partial E(v,h)}{\partial\theta}\\
&= -\sum_h p(h|v)\frac{\partial E(v,h)}{\partial\theta} + \sum_{v,h} p(v,h)\frac{\partial E(v,h)}{\partial\theta}, \qquad (3)
\end{aligned}
$$

where in deriving the first term in (3) we have used Equation (2).

- By $E(v,h) = -\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}h_i v_j - \sum_{j=1}^{m} b_j v_j - \sum_{i=1}^{n} c_i h_i$, the partial derivatives $\partial E(v,h)/\partial\theta$ can be easily computed for each $\theta = w_{ij}, b_j, c_i$.
- Let $\theta = w_{ij}$, thus $\partial E(v,h)/\partial w_{ij} = -h_i v_j$ for computation.
- The cases of $\theta = b_j, c_i$ are entirely similar.

Average log-likelihood gradient

- Taking average of the log-likelihood gradient of all training vectors for $\theta = w_{ij}$ we have:

$$\frac{1}{\ell} \sum_{v \in D} \frac{\partial \ln p(v|w_{ij})}{\partial w_{ij}}$$

$$= \frac{1}{\ell} \sum_{v \in D} \left[-\sum_h p(h|v) \frac{\partial E(v,h)}{\partial w_{ij}} + \sum_{v,h} p(v,h) \frac{\partial E(v,h)}{\partial w_{ij}} \right]$$

$$= \frac{1}{\ell} \sum_{v \in D} \left[\sum_h p(h|v) h_i v_j - \sum_h p(v,h) h_i v_j \right]$$

$$= \frac{1}{\ell} \sum_{v \in D} \left[\mathbb{E}_{p(h|v)}(h_i v_j) - \mathbb{E}_{p(v,h)}(h_i v_j) \right]$$

$$= \langle h_i v_j \rangle_{p(h|v)q(v)} - \langle h_i v_j \rangle_{p(v,h)} = \langle h_i v_j \rangle_{\text{data}} - \langle h_i v_j \rangle_{\text{model}} \quad (4)$$

  where q denotes the distribution of the data set and $\mathbb{E}_p$ denotes expectation value wrt the probability distribution p.

- Need to compute the averages in (4). The first term, called the **positive phase**, is easy to deal with by computing $p(h|v)$ (similar to $p(v|h)$). The second one, called the **negative phase**, can only be approximated.

Logistic transition probability $\sigma(x) = 1/(1 + e^{-x})$

- To compute $p(v_k = 1|h)$ let v_{-k} denote the state of all visible units other than the kth visible unit V_k.
- Put $\eta_k(h) := -\sum_{i=1}^{n} w_{ij} h_i - b_k$, and

$$\gamma(v_{-k}, h) := -\sum_i \sum_{j \neq k} w_{ij} h_i v_j - \sum_{j \neq k} b_j v_j - \sum_i c_i h_i.$$

- Then $E(v, h) = E(v_k, v_{-k}, h) = \gamma(v_{-k}, h) + v_k \eta_k(h)$. Thus, by independence of visible units:

$$p(v_k = 1|h) = p(v_k = 1|v_{-k}, h) = \frac{p(v_k = 1, v_{-k}, h)}{p(v_{-k}, h)}$$

$$= \frac{e^{-E(v_k=1, v_{-k}, h)}}{e^{-E(v_k=1, v_{-k}, h)} + e^{-E(v_k=0, v_{-k}, h)}}$$

$$= \frac{e^{-\gamma(v_{-k}, h) - 1 \cdot \eta_k(h)}}{e^{-\gamma(v_{-k}, h) - 1 \cdot \eta_k(h)} + e^{-\gamma(v_{-k}, h) - 0 \cdot \eta_k(h)}}$$

PTO

Logistic transition probability & Block Gibbs sampling

$$= \frac{e^{-\gamma(v_{-k},h)} \cdot e^{-\eta_k(h)}}{e^{-\gamma(v_{-k},h)} \cdot e^{-\eta_k(h)} + e^{-\gamma(v_{-k},h)}} = \frac{e^{-\gamma(v_{-k},h)} \cdot e^{-\eta_k(h)}}{e^{-\gamma(v_{-k},h)} \cdot (e^{-\eta_k(h)} + 1)}$$

$$= \frac{e^{-\eta_k(h)}}{e^{-\eta_k(h)} + 1} = \frac{1}{1 + e^{\eta_k(h)}} = \sigma(-\eta_k(h)) = \sigma\left(\sum_{i=1}^{n} w_{ik} h_i + b_k\right)$$

- Similarly, by symmetry, we have:

$$p(h_k = 1|v) = \sigma\left(\sum_{j=1}^{m} w_{kj} v_j + c_k\right)$$

- Since on each level the variables are independent, we can do **Block Gibbs sampling** in two steps in each stage:
(i) sample h based on $p(h|v) = \prod_{i=1}^{n} p(h_i|v)$, and,
(ii) sample v based on $p(v|h) = \prod_{j=1}^{m} p(v_j|h)$.

Computation of log-likelihood gradient (II)

- The first term in Equation (3), for $\theta = w_{ij}$, can now be calculated as follows. Recall that h_{-i} denotes the values of all hidden units except i.

$$-\sum_h p(h|v)\frac{\partial E(v,h)}{\partial \theta} = \sum_h p(h|v) h_i v_j$$

$$= \sum_{h_i}\sum_{h_{-i}} p(h_i|v)p(h_{-i}|v)h_i v_j = \sum_{h_{-i}} p(h_{-i}|v)\sum_{h_i} p(h_i|v)h_i v_j$$

$$= 1 \cdot \sum_{h_i} p(h_i|v)h_i v_j = p(h_i = 1|v)v_j = \sigma(\sum_{\ell=1}^{m} w_{i\ell}v_\ell + c_i)v_j$$

  since $\sum_{h_{-i}} p(h_{-i}|v) = 1$.
- This can thus be easily computed for any given state v of the visible vector, including training vectors.

Computation of log-likelihood gradient (III)

- For the second term in Equation (3) with $\theta = w_{ij}$, use $p(v,h) = p(v)p(h|v)$ and the result in the derivation of the first term to get:

$$\sum_{v,h} p(v,h) \frac{\partial E(v,h)}{\partial w_{ij}} = \sum_{v,h} p(v)p(h|v) \frac{\partial E(v,h)}{\partial w_{ij}}$$

$$= \sum_v p(v) \sum_h p(h|v) \frac{\partial E(v,h)}{\partial w_{ij}} = -\sum_v p(v) \sum_h p(h|v) h_i v_j$$

$$= -\sum_v p(v)p(h_i = 1|v)v_j \qquad (5)$$

- This has to be summed over all possible visible vectors, with an exponential complexity of 2^m.
- Instead, we can run MCMC by approximating this average using samples from model distribution as we computed averages for the stochastic Hopfield network.
- Unfortunately, this has to be done until the stationary distribution is reached and is itself intractable.

Block Gibbs Sampling and MCMC for RBM

- **Exercise.**

$$p(H_i = h_i|v) = \frac{e^{\sum_{j=1}^{m} w_{ij} v_j h_i + c_i h_i}}{1 + e^{\sum_{j=1}^{m} w_{ij} v_j + c_i}}$$

$$p(V_j = v_j|h) = \frac{e^{\sum_{i=1}^{n} w_{ij} v_j h_i + b_j v_j}}{1 + e^{\sum_{i=1}^{n} w_{ij} h_i + b_j}}$$

- Obtain transitional probabilities for block Gibbs sampling:

$$p(h|v) \quad \text{and} \quad p(v|h)$$

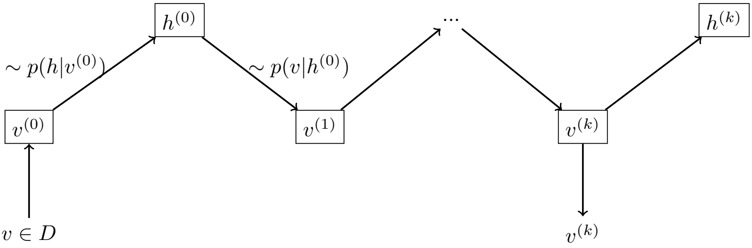
- We can then show that

$$p(v, h) = \frac{e^{-E(v,h)}}{Z}, \quad \text{where } Z = \sum_{v \in \{0,1\}^m, h \in \{0,1\}^n} e^{-E(v,h)}$$

  satisfies the detailed balance condition and is thus the stationary distribution of the RBM.
- Thus we can use MCMC to find averages wrt the stationary distribution.

Contrastive divergence CD-k

- **CD-k** is an algorithm to approximate MCMC for an RBM.
- We simply run Gibbs block sampling for only k steps:
- Start with a training vector $v^{(0)}$ and at step $0 \leq s \leq k-1$:
- Sample $h^{(s)} \sim p(h|v^{(s)})$;
- Sample $v^{(s+1)} \sim p(v|h^{(s)})$.
- Replace each term in (5) with $-p(h_i = 1|v^{(k)})v_j^{(k)}$.
- We usually take $k = 1$.



Overall algorithm for unsupervised training of RBM

init $\Delta w'_{ij} = \Delta b'_j = \Delta c'_i = 0$ for $i = 1, \ldots, n$, $j = 1, \ldots, m$

for all training mini-batches $T \subset D$ **do**

  init $\Delta w_{ij} = \Delta b_j = \Delta c_i = 0$ for $i = 1, \ldots, n$, $j = 1, \ldots, m$

  for all $v \in T$ **do**

    $v^{(0)} \leftarrow v$

    $v^{(k)} \leftarrow$ generate k-steps Gibbs sampling from $v^{(0)}$

    $\Delta w_{ij} \Leftarrow \Delta w_{ij} + p(h_i = 1 | v^{(0)}) \cdot v_j^{(0)} - p(h_i = 1 | v^{(k)}) \cdot v_j^{(k)}$

    $\Delta b_j \Leftarrow \Delta b_j + v_j^{(0)} - v_j^{(k)}$

    $\Delta c_i \Leftarrow \Delta c_i + p(h_i = 1 | v^{(0)}) - p(h_i = 1 | v^{(k)})$

  end for

  $w_{ij} \leftarrow w_{ij} + \frac{\alpha}{|T|} \cdot \Delta w_{ij} + \nu \Delta w'_{ij} - \lambda w_{ij}$

  $b_j \leftarrow b_j + \frac{\alpha}{|T|} \cdot \Delta b_j + \nu \Delta b'_j - \lambda b_j$

  $c_i \leftarrow c_i + \frac{\alpha}{|T|} \cdot \Delta c_i + \nu \Delta c'_i - \lambda c_i$

  $\Delta w'_{ij} \leftarrow \Delta w_{ij}$

  $\Delta b'_j \leftarrow \Delta b_j$

  $\Delta c'_i \leftarrow \Delta c_i$

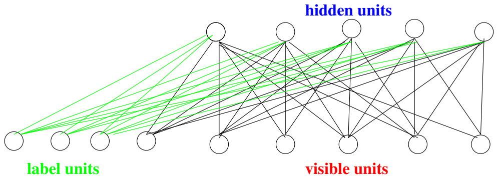
end for

Some comments about the overall algorithm

- We usually use $k = 1$, i.e., we implement CD-1.
- In terms of the gradient ascent algorithm described in the recursive Equation (1), the overall algorithm uses $\theta = w_{ij}, b_j, c_i$.
- The explicit time dependence $\theta^{(t)}$ has been suppressed to avoid cluttering the formulas.
- In fact, w_{ij}, b_j and c_i stand for $w_{ij}^{(t)}$, $b_j^{(t)}$ and $c_i^{(t)}$, while w'_{ij}, b'_j and c'_i stand for $w_{ij}^{(t-1)}$, $b_j^{(t-1)}$ and $c_i^{(t-1)}$.
- The overall algorithm thus includes one loop of Equation (1) for updating values of $w_{ij}^{(t)}$, $b_j^{(t)}$ and $c_i^{(t)}$.
- For practical information on how to choose the parameters such α, λ, ν, batch size, or the initial values of weights and biases, see G. Hinton's: A practical guide to training restricted Boltzmann machines.

RBM as a Generative Model

- An RBM can be used to generate new data similar to those it has been trained with.
- Suppose we have a labelled data set, e.g., the MNIST handwritten digits with ten classes, one for each digit.
- There are in general a number of classes or labels and each item in the data set has a unique label.
- For each class include a visible unit, which would be turned on when the RBM is trained for any item in that class.
- After training, if we clamp the unit for a given class to “on” and the rest of class units to “off”, the RBM generates patterns that it classifies in the given class.



Softmax function

- For a single binary node with value $v = 0$ or $v = 1$, the energy is $E = -bv$ and thus the probability of $v = 1$ is given by the Logistic sigmoid function:

$$\frac{e^{-E(1)}}{e^{-E(1)} + e^{-E(0)}} = \frac{e^{-E(1)}}{e^{-E(1)} + e^{-E(0)}} = \frac{1}{1 + e^{E(1)}}$$

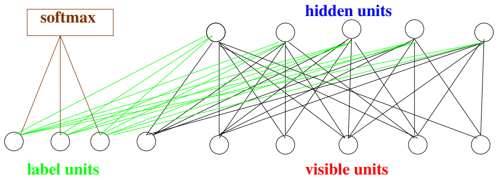
- Suppose we have L labels or classes, each having a weight $z_k \in \mathbb{R}$ for $1 \leq k \leq L$. Then we can generalise the Logistic sigmoid map to L states.
- The **softmax** function takes a vector in $z \in \mathbb{R}^L$ of L real numbers and provides a probability vector with L components:

$$\frac{e^{z_k}}{\sum_{l=1}^{L} e^{z_l}}$$

- From this probability vector, we can sample a value of k with $1 \leq k \leq L$.

RBM as a Discriminative Model

- We include a softmax unit which finds the probability of the labels, given the number of times each label unit is activated during a specific period provides.
- We train the RBM as in the generative model.
- For classification, we clamp the visible units to the values for the pattern we like to classify.
- We run Gibbs sampling for a specified number of times and each time one label becomes activated by the softmax unit.
- The active label will become stable at the end of Gibbs sampling, thus classifying our pattern.



Basic properties of RBM

- Given a probability distribution q on our data set, the RBM marginal probability distribution p for visible units can actually coincide with q if enough hidden units are used: In fact, if $k+1$ hidden units are used where k is the number of different configurations in $\{0,1\}^m$ with non-zero q value.
- In general though the marginal distribution p is only an approximation to q.
- An upper bound for the average error in k-step contrastive divergence (CD-k) is given by

$$\frac{1}{2}\|q-p\|\left(1-e^{-(m+n)\Delta}\right)^k$$

  where m and n are the number of visible and hidden units, and Δ is a positive number which can be obtained from the final values of w_{ij}, b_j and c_i.
- Therefore as $k\to\infty$ the average error converges to zero.

Averaging and Sampling: Justifying CD-k

- For an irreducible and aperiodic transition matrix P on a finite state space S with stationary distribution π, recall that $qP^n \to \pi$ for any initial probability vector q and also that $\lim_{n\to\infty} \mathbb{E}_{qP^n}(f) = \mathbb{E}_\pi(f)$ for any function $f : S \to \mathbb{R}$.
- Now, if $x^{(0)} \in S$ is any sample $x^{(0)} \sim q$, and we recursively construct a sequence of samples $x^{(k+1)} \sim P(x|x^{(k)})$, i.e., $x^{(k+1)} \sim x^{(k)}P$, then for large k we have: $x^{(k)} \sim \pi$
- If $x_j^{(0)} \sim q$, with $1 \leq j \leq \ell$ for large ℓ, is a set of initial samples, by Central Limit Theorem, we have for large k:

$$\frac{1}{\ell}\sum_{j=1}^{\ell} f(x_j^{(k)}) \approx \mathbb{E}_\pi(f).$$

which justifies CD-k, with q as the probability distribution over the data set D and $\pi(v,h) = \Pr(v,h) = e^{-E(v,h)}/Z$.