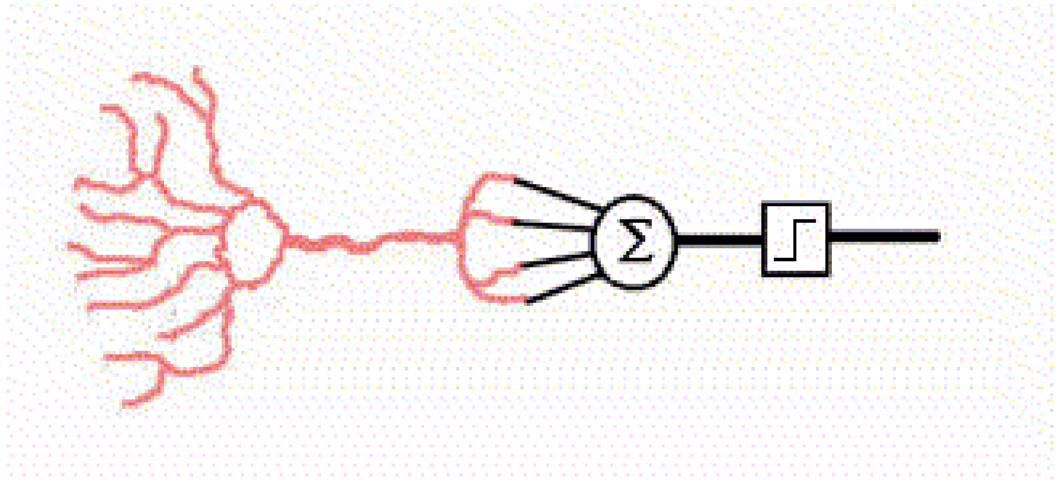


NEURAL NETWORKS (CS010 805G02)

Mod 3 - Stochastic networks

Boltzmann Machines

Shiney Thomas
AP,CSE,AJCE

INTRODUCTION

- Unsupervised learning- Statistical Mechanics
  - Statistical mechanics is a branch of theoretical physics that uses probability theory to study the average behaviour of a mechanical system whose exact state is uncertain.
  - Commonly used to explain the thermodynamic behaviour of large systems.
  - The benefit of using statistical mechanics is that it provides exact methods to connect thermodynamic quantities (such as temperature, heat, or entropy) to microscopic behaviour(of atoms, electrons),
- Boltzmann Machine(Hinton and Sejnowski,1983,1986) – first multilayer learning machine inspired by statistical mechanics

Stochastic networks

- In probability theory and related fields, a **stochastic** or **random process** is a mathematical object usually defined as a collection of random variables.
- **Stochastic neural networks** are a type of artificial neural networks built by introducing random variations into the network, either by giving the network's neurons stochastic transfer functions, or by giving them stochastic weights.
- This makes them useful tools for optimization problems, since the random fluctuations help it escape from local minima.
- An example of a neural network using stochastic transfer functions is a Boltzmann machine. Each neuron is binary valued, and the chance of it firing depends on the other neurons in the network.

STOCHASTIC NETWORKS

- Basically BM is a device for modelling the underlying probability distribution of a given data set, from which conditional distributions for use in tasks like pattern completion and pattern classification can be derived.
- But learning process in BM is extremely slow. New stochastic machines such as
  - Sigmoid belief networks
  - Helmholtz machines

Basics of Statistical Mechanics

- Consider a physical system with many degrees of freedom,that can reside in any one of a large number of possible states.
- Let $\boldsymbol{p_i}$ be probability of occurrence of state $\boldsymbol{i}$, with properties

$$p_i \geq 0 \quad \text{for all } i$$

$$\sum_i p_i = 1$$

- Let $\boldsymbol{E_i}$ denote the energy of the system when it is in state $\boldsymbol{i}$. In SM when the system is in thermal equilibrium with its surrounding environment, the state $\boldsymbol{i}$ occurs with a probability

$$p_i = \frac{1}{Z}\exp\left(-\frac{E_i}{k_B T}\right)$$

T- abs temp. in Kelvins
k_B - Boltzmann's constant ($1.38 \text{ X } 10^{-23}$ joules/kelvin)
Z – a constant independent of all states

Basics of Statistical Mechanics

- From equations 2(condition for normalization of probabilities) ,3 we get

$$Z = \sum_i \exp\left(-\frac{E_i}{k_B T}\right)$$

- Normalizing quantity Z is called sum over states or the partition function
- In context of neural networks, parameter T can be viewed as pseudotemperature that controls thermal fluctuations representing the effect of "synaptic noise" in a neuron.
  - Precise scale irrelevant
  - Set k_B = unity , thus $\boldsymbol{p_i}$ and Z

$$p_i = \frac{1}{Z}\exp\left(-\frac{E_i}{T}\right)$$

$$Z = \sum_i \exp\left(-\frac{E_i}{T}\right)$$

Boltzmann Machine

- The Boltzmann machine (named in honour of the 19th-century physicist and inventor of statistical mechanics by its inventors) has similarities to and differences from the Hopfield net.
- Boltzmann machine is a stochastic machine whose composition consists of stochastic neurons
- Two possible states ,+1,-1 (or 1,0) = ‘on’ and ‘off’ states resp.
- The stochastic neurons of a Boltzmann machine are partitioned in two groups: visible and hidden
  - *visible neurons* provide an interface between the net and its environment during the training phase, the visible neurons are clamped on to specific states determined by env;
  - *the hidden neurons* always operate freely, they are used to explain underlying constraints in the environmental input vectors.

Boltzmann Machine

- The stochastic neurons of a Boltzmann machine are partitioned in two groups: visible and hidden

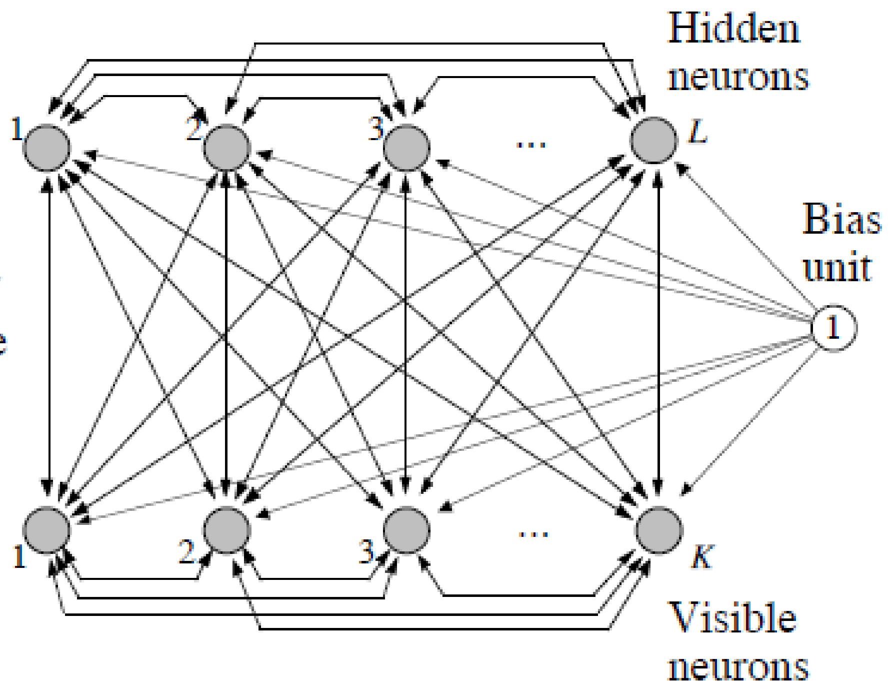


Figure 11.4: Architectural graph of Boltzmann machine; K is the number of visible neurons and L is the number of hidden neurons.

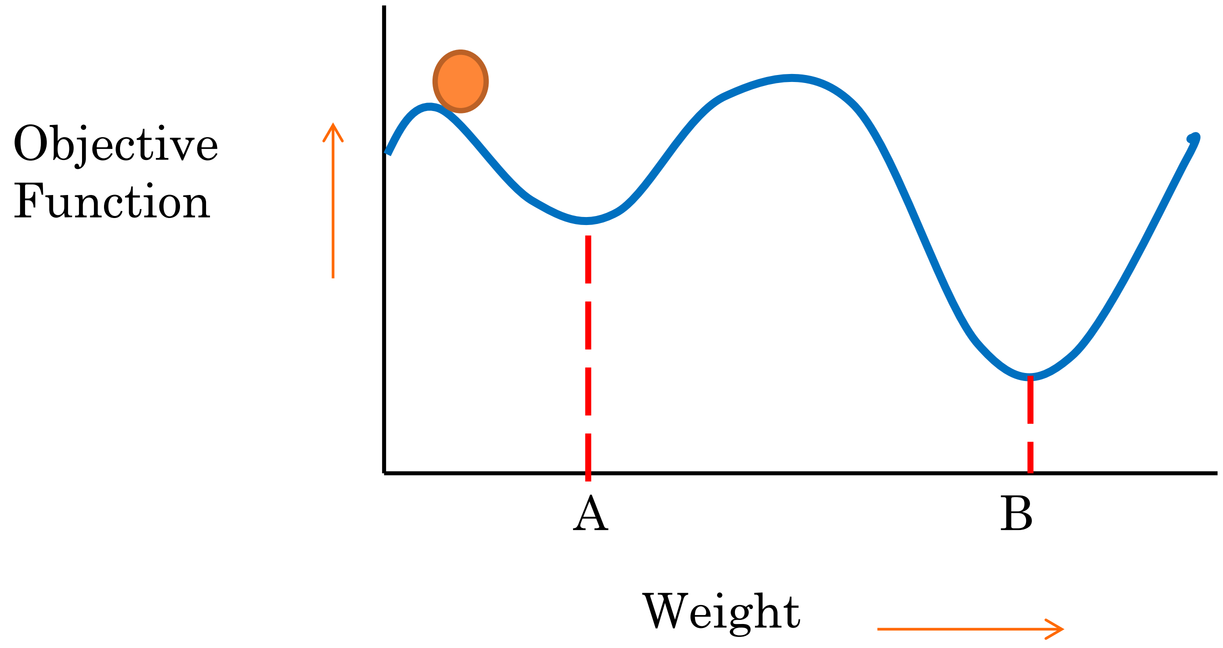
Boltzmann Machine

- The hidden units "explain" underlying constraints by capturing higher-order correlations between the clamping vectors.
- Boltzmann machine learning may be viewed as an unsupervised learning procedure for modelling a probability distribution that is specified by the clamping patterns onto visible neurons.
- The network can perform pattern completion:
  - when a vector bearing part of the information is clamped onto a subset of the visible neurons, the network performs completion of the pattern on the remaining visible neurons (if it has learnt properly).

- **Deterministic training method-** step by step procedure to adjust the n/w weights based on the values of inputs, actual outputs and desired outputs. Eg Perceptron training
- **Statistical training methods** – makes pseudorandom changes in the weight values, retaining those changes that result in improvements.
- Our aim while training the network is to minimize the difference btw actual output and desired output(objective function),but we can get trapped in a poor solution.

Ref: Philip D Wasserman

- Local minima problem



- Marble on a surface of a block
- If box is shaken violently in a horizontal direction, the marble will move rapidly from side to side. Never settling at any one point, at any instant the marble may be at any point on the surface with equal probability
- If the violence of shaking is gradually reduced, a condition will be reached in which the marble “sticks” briefly at point B
  - Lower level of shaking, marble will stay at both points A and B for short times
  - If it is continually reduced ,a critical point will be reached where shaking is just strong enough to move the marble from point A to point B but not from B to A(climb hill)
  - Thus the marble will end up in a global minimum as shaking amplitude is reduced to zero.

- ANN can be trained in same way through random adjustments of weights
  - First Large random adjustments are made, retaining only those weight changes that reduce the objective function
  - Average step size is then gradually reduced and a global minimum will eventually be reached
- Similar to Annealing of metals- Hence known as Simulated Annealing
- Distribution of energy states is

$$P(E) \propto \exp(-E/kT)$$

Boltzmann Training

1) Define a variable T that represents an artificial temperature. Start with T at a large value
2) Apply a set of inputs to the network, and calculate the outputs and objective function
3) Make a random weight change, and recalculate the output and the change in objective function(error minimization) due to the weight change
4) If the objective function is reduced(improved), retain the weight change

- If the weight change results in increase in the objective function, calculate the probability of accepting that change from the Boltzmann distribution **$P(c) = \exp(-c / kT)$**

  where P(c) is the probability of a change of c in the objective function

BOLTZMANN TRAINING

- Select a random number **r** from a uniform distribution between 0 and 1

  If P(c) > r , retain the change;

  Return the weight to previous value , otherwise;
- Repeat steps 3 and 4 over each of the weights in the network, gradually reducing the temperature T until an acceptable low value for the objective function is achieved
- A different input vector is applied and training process is repeated, until the objective function is acceptable
- Random weight change -> gaussian distribution

Boltzmann Learning Rule

- Two phases of operation:
- Positive phase- The network operates in its clamped condition(i.e, under the direct influence of training set, $\mathscr{T}$)
- Negative phase: The network is allowed to run freely and therefore with no environmental input

How it Works

- Step 1. Pick an example
- Step 2. Run network in *positive phase*
- Step 3. Run network in *negative phase*
- Step 4. Compare the statistics of the two phases
- Step 5. Update the weights based on statistics
- Step 6. Go to step 1 and repeat.

Step 1: Pick Example

- Pretty simple. Just select an example at random.

Step 2. The Positive Phase

- *Clamp* our visible units with the pattern specified by our current example
- Let network *settle* using the simulated annealing method
- Record the outputs of the units
- Start again with our example, settling again and recording units again.

Step 3. The Negative Phase

- Here, we don’t clamp the network units. We just let it settle to some state as before.
- Do this several times, again recording the unit outputs.

Step 4. Compare Statistics

- For each pair of units, we compute the odds that both units are coactive (both on) for the positive phase. Do it also for the negative phase.
- If we have n units, this gives us two n x n matrices of probabilities
- $p_{i,j}$ is probability that both unit i and j are both on.

Step 5: Update Weights

$$\Delta w_{i,j} = k(p_{i,j}^{+} - p_{i,j}^{-})$$

- Change each weight according to the difference of the probabilities for the positive and negative phase
- Here, *k (or η)* is a learning rate

Boltzmann Learning

- Stochastic learning process with a recurrent structure
- State of a neuron is +1 or –1 and some neurons are free (adaptive state) and others are clamped (frozen state)
- Boltzmann machine is characterized by an energy function

$$E = -\tfrac{1}{2}\sum_{j}\sum_{k\neq j} w_{kj} x_k x_j$$

- Free neurons change state with probability:

$$P(+x_k \rightarrow -x_k) = \frac{1}{1+\exp(-\Delta E_k / T)}$$

- The learning rule is given by:

$$\Delta w_{kj} = \eta\left(p_{kj}^{+} - p_{kj}^{-}\right) \quad j \neq k$$

Where ρ^{+}_{kj} is the correlation with neurons in clamped states and ρ^{-}_{kj} is the correlation with the neurons in a frozen state

Why it Works

- This reduces the difference between what the network settles to when the inputs are clamped, and what it settles to when its allowed to free-run.
- So, the weights learn about what kinds of visible units go together.
- Recruits hidden units to help learn higher order relationships

Can Be Used For Mappings Too

- Here, the positive phase involves clamping both the input and output units and letting the network settle.
- The negative phase involves clamping just the input units
- Network learns that given the input, it should settle to a state where the output units are what they should be