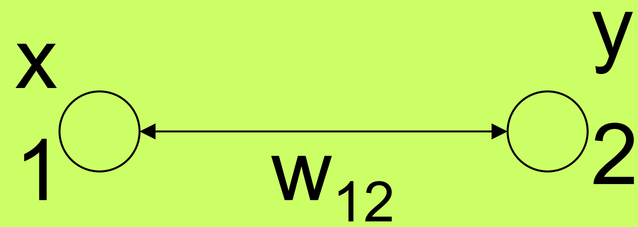
CS623: Introduction to Computing with Neural Nets *(lecture-19)*

Pushpak Bhattacharyya
Computer Science and Engineering Department
IIT Bombay

Illustration of the basic idea of Boltzmann Machine

- To learn the identity function
- The setting is probabilistic, x = 1 or

  x = -1, with uniform probability, *i.e.*,
  - P(x=1) = 0.5, P(x=-1) = 0.5
- For, x=1, y=1 with P=0.9
- For, x=-1, y=-1 with P=0.9



x	y
1	1
-1	-1

Illustration of the basic idea of Boltzmann Machine (contd.)

- Let α = output neuron states

  β = input neuron states

  $P_{\alpha|\beta}$ = observed probability distribution

  $Q_{\alpha|\beta}$ = desired probability distribution

  Q_{β} = probability distribution on input states β

Illustration of the basic idea of Boltzmann Machine (contd.)

- The divergence D is given as:

$D = \sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta \ln Q_{\alpha|\beta} / P_{\alpha|\beta}$

called KL divergence formula

$D = \sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta \ln Q_{\alpha|\beta} / P_{\alpha|\beta}$

$>= \sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta (1 - P_{\alpha|\beta} / Q_{\alpha|\beta})$

$>= \sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta - \sum_\alpha \sum_\beta P_{\alpha|\beta} Q_\beta$

$>= \sum_\alpha \sum_\beta Q_{\alpha\beta} - \sum_\alpha \sum_\beta P_{\alpha\beta}$

{$Q_{\alpha\beta}$ and $P_{\alpha\beta}$ are joint distributions}

$>= 1 – 1 = 0$

Gradient descent for finding the weight change rule

$P(S_\alpha)\ \alpha\ exp(-E(S_\alpha)/T)$

$P(S_\alpha) = (exp(-E(S_\alpha)/T)) / (\sum_{\beta\ \epsilon\ all\ states} exp(-E(S_\beta)/T)$

$ln(P(S_\alpha)) = (-E(S_\alpha)/T) - ln\ Z$

$D = \sum_\alpha \sum_\beta Q_{\alpha|\beta}\ Q_\beta\ ln\ (Q_{\alpha|\beta} / P_{\alpha|\beta})$

$\Delta w_{ij} = \eta\ (\delta D/\delta w_{ij})$; gradient descent

Calculating gradient: 1/2

$$\delta D / \delta w_{ij} = \delta/\delta w_{ij} [\sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta \ln (Q_{\alpha|\beta} / P_{\alpha|\beta})]$$

$$= \delta/\delta w_{ij} [\sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta \ln Q_{\alpha|\beta}$$

$$- \sum_\alpha \sum_\beta Q_{\alpha|\beta} Q_\beta \ln P_{\alpha|\beta}]$$

Constant
With respect
To w_{ij}

$$\delta(\ln P_{\alpha|\beta}) / \delta w_{ij} = \delta/\delta w_{ij} [-E(S_\alpha)/T - \ln Z]$$

$$Z = \sum_\beta exp(-E(S_\beta))/T$$

Calculating gradient: 2/2

$$\delta\, [-E(S_\alpha)/T]\, /\delta w_{ij} = (-1/T)\ \delta/\delta w_{ij}\, [\, -\sum_i \sum_{j>i} w_{ij}\, s_i\, s_j\,]$$
$$= (-1/T)[-s_i s_j|_\alpha]$$
$$= (1/T)[s_i s_j|_\alpha]$$

$$\delta\ (\ln Z)/\delta w_{ij}=(1/Z)(\delta Z/\delta w_{ij})$$

$$Z= \sum_\beta exp(-E(S_\beta)/T)$$

$$\delta Z/\delta w_{ij}= \sum_\beta[exp(-E(S_\beta)/T)(\delta(-E(S_\beta/T)/\delta w_{ij}\,))]$$
$$= (1/T)\ \sum_\beta exp(-E(S_\beta)/T).s_i s_j|_\beta$$

Final formula for Δw_{ij}

$$\Delta w_{ij} = [1/T/\ [s_i s_j|_\alpha - (1/Z) \sum_\beta exp(-E(S_\beta)/T).s_i s_j|_\beta$$

$$= [1/T][s_i s_j|_\alpha - \underbrace{\sum_\beta P(S_\beta).s_i s_j|_\beta]}$$

Expectation of i[th] and j[th]
Neurons being on together

Issue of Hidden Neurons

- Boltzmann machines
  - can come with hidden neurons
  - are equivalent to a Markov Random field
  - with hidden neurons are like a Hidden Markov Machines
- Training a Boltzmann machine is equivalent to running the Expectation Maximization Algorithm

Use of Boltzmann machine

- Computer Vision
  - Understanding scene involves what is called "Relaxation Search" which gradually minimizes a cost function with progressive relaxation on constraints
- Boltzmann machine has been found to be slow in the training
  - Boltzmann training is NP-hard.

Questions

- Does the Boltzmann machine reach the global minimum? What ensures it?
- Why is simulated annealing applied to Boltzmann machine?
  - local minimum → increase T → n/w runs →gradually reduce T → reach global minimum.
- Understand the effect of varying T
  - Higher T → small difference in energy states ignored, convergence to local minimum fast.