CPSC 550: Machine Learning II	2008/9 Term 2
Lecture 13 — Mar 3, 2009	
Lecturer: Nando de Freitas	*Scribe: Bo Chen*

This lecture draws a theoretical comparison between two parameter learning algorithms, namely the **Maximum Likelihood (ML)** and the **Contrastive Divergence (CD)**, in the context of **Restricted Boltzmann Machines (RBMs)**. Then, a multi-layer model called the **Deep Belief Net** is introduced.

13.1 Comparison of Learning Methods

This section compares ML with CD for RBMs. A generic ML learning algorithm for Boltzmann Machines is first presented, followed by a simplified version for RBMs.

13.1.1 Notation

An RBM contains a layer of visible units v, a layer of hidden units h, and connections between the two with weights W. The training data set is $\{d_t\}_{t=1}^{T}$. For notational convenience the instance number t is dropped when the context is clear.

13.1.2 ML for Boltzmann Machines

In the previous lecture, the gradient of the log likelihood was derived. ML basically tries to be as loyal to the exact gradient as possible and performs gradient descent to update the weights W.

The algorithm starts with some initial guess of W. It then iteratively updates W until some stopping criterion is met. In the kth iteration:

1. Sample the hidden units from the conditional using Markov chain Monte Carlo (MCMC):

$$\overset{\sim(k)}{h} \sim P(h|v=d)$$

2. Sample both the hidden and visible units by running a Gibbs Chain until equilibrium:

$$\overset{\approx}{v}^{(k)} \sim P(v|\tilde{h}^{(k-1)})$$
$$\overset{\approx}{h}^{(k)} \sim P(h|\tilde{v}^{(k-1)})$$

3. Update the connection weights. Each w_{ij} associated with the ith visible unit and the jth hidden unit is updated as follows:

$$w_{ij}^{(k)} = w_{ij}^{(k-1)} + \frac{\eta_k}{T}\sum_{t=1}^{T}\left(d_{it}\tilde{h}_{jt}^{(k)} - \overset{\approx}{v}_{it}^{(k)}\overset{\approx}{h}_{jt}^{(k)}\right) \tag{13.1}$$

where η_k is the learning rate or step length, which usually decays as the number of iterations grows.

This is called Controlled Markov Chain.

In case of RBMs, the updating formula (13.1) can be changed to:

$$w_{ij}^{(k)} = w_{ij}^{(k-1)} + \frac{\eta_k}{T}\sum_{t=1}^{T}\left(d_{it}p(h_{jt}=1|d_t) - p(v_{it}=1|\widetilde{h}_t)p(h_{jt}=1|\overset{\approx}{v}_t^{(k)})\right) \tag{13.2}$$

where the probabilities can all be computed exactly:

$$p(h_{jt}=1|d_t) = \sigma(\sum_i w_{ij}d_{it})$$

$$p(v_{it}=1|\widetilde{h}_t) = \sigma(\sum_j w_{ij}\widetilde{h}_{tj})$$

$$p(h_{jt}=1|\overset{\approx}{v}_t^{(k)}) = \sigma(\sum_i w_{ij}\overset{\approx}{v}_{it}^{(k)})$$

This is process is called Rao-Blackwellization. By computing the exact posterior instead of sampling, Eq(13.2) can be shown to give an estimator with less variance than Eq(13.1). This claim is proven in the next lecture.

13.1.3 CD-RBM

Unlike ML, which runs a long Gibbs Chain to acquire samples from the RBM, CD learning draws one sample close to the training data. The kth iteration of CD goes as follows:

1. Compute $P(h_j|d)$, the conditional distribution
2. Sample hidden units conditioning on the data:
$$\widetilde{h}_j \sim P(h_j|d)$$
3. Sample the visible units conditioning on the samples of hidden units:
$$\widetilde{v}_i \sim P(v_i|\widetilde{h}_j)$$
4. Compute $P(h_j|\widetilde{v}_i)$, the distribution over hidden units given the samples of visible units
5. The CD updating equation is:
$$w_{ij}^{(k)} = w_{ij}^{(k-1)} + \frac{\eta_k}{T}\sum_{t=1}^{T}\Big(d_{it}P(h_{jt}=1|d_t) - \widetilde{v}_{it}P(h_{jt}=1|v_{it})\Big)$$

13.1.4 Comparison Between ML-RBM and CD-RBM

As mentioned above, ML is computationally inhibitive since it runs a Gibbs Chain until equilibrium in every gradient descent iteration. Nonetheless, it is an unbiased estimator. On the contrary, CD is extremely efficient because only $O(1)$ samples are needed for each update. The disadvantage is that its samples are not drawn from the distribution implied by the RBM and hence the estimate of the gradient is not necessarily unbiased.

To further appreciate the difference between the two, it is helpful to examine the changes CD makes to the energy landscape (See figure 13.1). The data form four clusters in the feature space. CD modifies the energy surface near the data such that simulated data near the

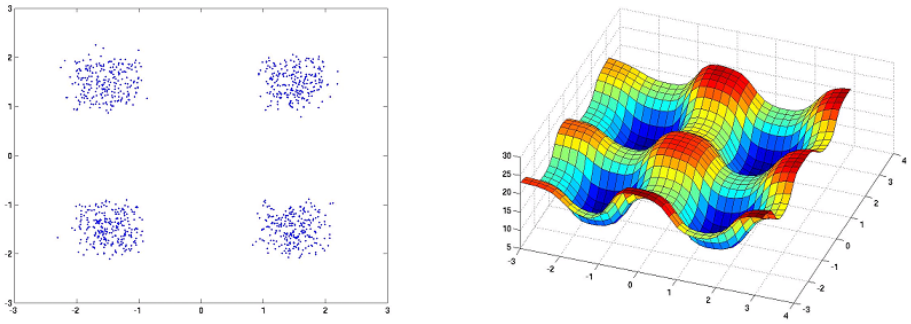

Figure 13.1. (left)Training data in 2D. (right)Energy surface after CD learning. Source: *CIAR Second Summer School Tutorial Lecture on Contrastive Divergence and Deterministic Energy-Based Models by Geoffrey Hinton*

clusters give low energy, but it does not model the area far away from the data. ML, on the other hand, will attempt to model the energy landscape everywhere in the feature space, regardless of the amount of support received from the data. Consequently, CD usually works better when the test data are close to the training data, whereas ML is more generalizable to completely novel data.

13.2 Deep Belief Nets

The RBMs can be stacked to create multi-layer networks called **Deep Belief Nets (DBNs)**.

Parameter estimation for DBNs is a simple extension of learning RBMs. Ignoring the higher levels, the visible layer and the lowest hidden layer constitute an RBM, of which the weights can be trained using CD. Then the activities of the hidden layer are fixed (hence treated as data) and used to learn higher-level weights. This procedure is repeated until the top layer is reached, where various tasks, including dimensionality reduction and discrimination, take place. See figure 13.2 for an example.

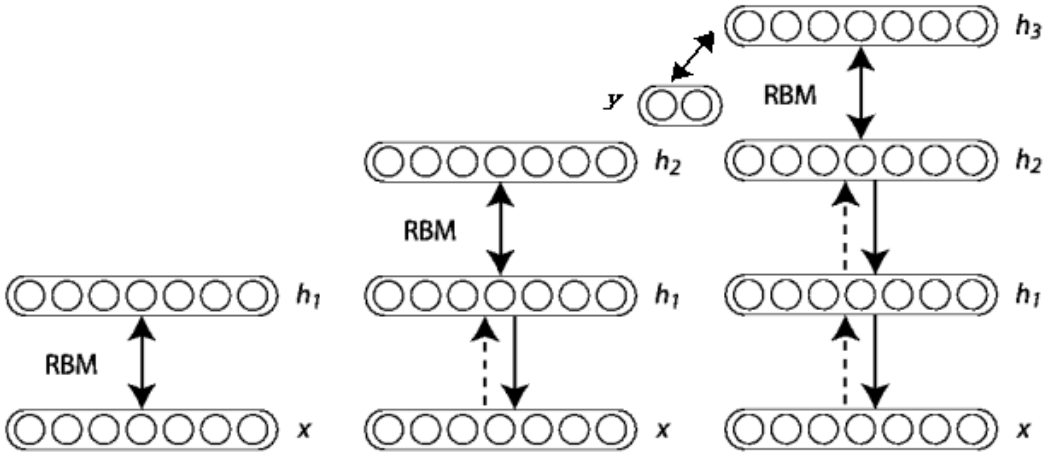

Figure 13.2. Illustration of learning a Deep Belief Net with three hidden layers. In step 1, an RBM consisting of (h_1, x) is learned, then the weights between h_1 and x as well as the activations of h_1 are fixed. Similarly in step 2, weights between h_2 and h_1 (now treated as data) are learned. Finally in step 3, an RBM for classification is trained on the data obtained by concatenating the labels y (observed) and the hidden units h_2 (fixed after step 2). Adapted from *http://www.iro.umontreal.ca/ lisa/twiki/bin/view.cgi/Public/DeepBeliefNetworks*