On the Quantitative Analysis of Deep Belief Networks

Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Iain Murray MURRAY@CS.TORONTO.EDU
Department of Computer Science, University of Toronto, Toronto, Ontario M5S 3G4, Canada

Abstract

Deep Belief Networks (DBN's) are generative models that contain many layers of hidden variables. Efficient greedy algorithms for learning and approximate inference have allowed these models to be applied successfully in many application domains. The main building block of a DBN is a bipartite undirected graphical model called a restricted Boltzmann machine (RBM). Due to the presence of the partition function, model selection, complexity control, and exact maximum likelihood learning in RBM's are intractable. We show that Annealed Importance Sampling (AIS) can be used to efficiently estimate the partition function of an RBM, and we present a novel AIS scheme for comparing RBM's with different architectures. We further show how an AIS estimator, along with approximate inference, can be used to estimate a lower bound on the log-probability that a DBN model with multiple hidden layers assigns to the *test data*. This is, to our knowledge, the first step towards obtaining quantitative results that would allow us to directly assess the performance of Deep Belief Networks as generative models of data.

1. Introduction

Deep Belief Networks (DBN's), recently introduced by Hinton et al. (2006) are probabilistic generative models that contain many layers of hidden variables, in which each layer captures strong high-order correlations between the activities of hidden features in the layer below. The main breakthrough introduced by Hinton et al. was a greedy, layer-by-layer unsupervised learning algorithm that allows efficient training of these deep, hierarchical models. The learning procedure also provides an efficient way of performing approximate inference, which makes the values of the latent variables in the deepest layer easy to infer. These deep generative models have been successfully applied in many application domains (Hinton & Salakhutdinov, 2006; Bengio & LeCun, 2007).

The main building block of a DBN is a bipartite undirected graphical model called the Restricted Boltzmann Machine (RBM). RBM's, and their generalizations to exponential family models, have been successfully applied in collaborative filtering (Salakhutdinov et al., 2007), information and image retrieval (Gehler et al., 2006), and time series modeling (Taylor et al., 2006). A key feature of RBM's is that inference in these models is easy. An unfortunate limitation is that the probability of data under the model is known only up to a computationally intractable normalizing constant, known as the partition function. A good estimate of the partition function would be extremely helpful for model selection and for controlling model complexity, which are important for making RBM's generalize well.

There has been extensive research on obtaining deterministic approximations (Yedidia et al., 2005) or deterministic upper bounds (Wainwright et al., 2005) on the log-partition function of arbitrary discrete Markov random fields (MRF's). These variational methods rely critically on an ability to approximate the entropy of the undirected graphical model. However, for densely connected MRF's, such as RBM's, these methods are unlikely to perform well. There have also been many developments in the use of Monte Carlo methods for estimating the partition function, including Annealed Importance Sampling (AIS) (Neal, 2001), Nested Sampling (Skilling, 2004), and many others (see e.g. Neal (1993)). In this paper we show how one such method, AIS, by taking advantage of the bipartite structure of an RBM, can be used to efficiently estimate its partition function. We further show that this estimator, along with approximate inference, can be used to estimate a lower bound on the log-probability that a DBN model with multiple hidden layers assigns to training or test data. This result allows us to assess the performance of DBN's as generative models and to compare them to other probabilistic models, such as plain mixture models.

Appearing in *Proceedings of the 25^{th} International Conference on Machine Learning*, Helsinki, Finland, 2008.

2. Restricted Boltzmann Machines

A Restricted Boltzmann Machine is a particular type of MRF that has a two-layer architecture in which the visible, binary stochastic units $\mathbf{v} \in \{0,1\}^D$ are connected to hidden binary stochastic units $\mathbf{h} \in \{0,1\}^M$. The energy of the state $\{\mathbf{v}, \mathbf{h}\}$ is:

$$E(\mathbf{v},\mathbf{h};\theta) = -\sum_{i=1}^{D}\sum_{j=1}^{M} W_{ij} v_i h_j - \sum_{i=1}^{D} b_i v_i - \sum_{j=1}^{M} a_j h_j, \quad (1)$$

where $\theta = \{W, \mathbf{b}, \mathbf{a}\}$ are the model parameters: W_{ij} represents the symmetric interaction term between visible unit i and hidden unit j; b_i and a_j are bias terms. The probability that the model assigns to a visible vector $\mathbf{v}$ is:

$$p(\mathbf{v};\theta) = \frac{p^*(\mathbf{v};\theta)}{Z(\theta)} = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} \exp\left(-E(\mathbf{v},\mathbf{h};\theta)\right), \quad (2)$$

$$Z(\theta) = \sum_{\mathbf{v}}\sum_{\mathbf{h}} \exp\left(-E(\mathbf{v},\mathbf{h};\theta)\right), \quad (3)$$

where p^* denotes unnormalized probability, and $Z(\theta)$ is the partition function or normalizing constant. The conditional distributions over hidden units $\mathbf{h}$ and visible vector $\mathbf{v}$ are given by logistic functions:

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}), \quad p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}) \quad (4)$$

$$p(h_j = 1|\mathbf{v}) = \sigma\Big(\sum_i W_{ij} v_i + a_j\Big) \quad (5)$$

$$p(v_i = 1|\mathbf{h}) = \sigma\Big(\sum_j W_{ij} h_j + b_i\Big), \quad (6)$$

where $\sigma(x) = 1/(1+\exp(-x))$. The derivative of the log-likelihood with respect to the model parameter W can be obtained from Eq. 2:

$$\frac{\partial \ln p(\mathbf{v})}{\partial W_{ij}} = \mathrm{E}_{P_0}[v_i h_j] - \mathrm{E}_{P_{\mathrm{Model}}}[v_i h_j],$$

where $\mathrm{E}_{P_0}[\cdot]$ denotes an expectation with respect to the data distribution and $\mathrm{E}_{P_{\mathrm{Model}}}[\cdot]$ is an expectation with respect to the distribution defined by the model. The expectation $\mathrm{E}_{P_{\mathrm{Model}}}[\cdot]$ cannot be computed analytically. In practice learning is done by following an approximation to the gradient of a different objective function, called the "Contrastive Divergence" (CD) (Hinton, 2002):

$$\Delta W_{ij} = \epsilon\big(\mathrm{E}_{P_0}[v_i h_j] - \mathrm{E}_{P_T}[v_i h_j]\big). \quad (7)$$

The expectation $\mathrm{E}_{P_T}[\cdot]$ represents a distribution of samples from running the Gibbs sampler (Eqs. 5, 6), initialized at the data, for T full steps. Setting $T = \infty$ recovers maximum likelihood learning, although T is typically set to one.

Even though CD learning may work well in practice, the problem of model selection and complexity control still remains. Suppose we have two RBM's with parameter values θ_A and θ_B. Suppose that each RBM has different number of hidden units and was trained using different learning rates and different numbers of CD steps. On the validation set, we are interested in calculating the ratio:

$$\frac{p(\mathbf{v};\theta_A)}{p(\mathbf{v};\theta_B)} = \frac{p^*(\mathbf{v};\theta_A)}{p^*(\mathbf{v};\theta_B)} \frac{Z(\theta_B)}{Z(\theta_A)},$$

which requires knowing the ratio of partition functions.

3. Estimating Ratios of Partition Functions

Suppose we have two distributions defined on some space $\mathcal{V}$ with probability density functions: $p_A(\mathbf{v}) = p^*_A(\mathbf{v})/Z_A$ and $p_B(\mathbf{v}) = p^*_B(\mathbf{v})/Z_B$. One way to estimate the ratio of normalizing constants is to use a simple importance sampling (IS) method. Suppose that $p_A(\mathbf{v}) \neq 0$ whenever $p_B(\mathbf{v}) \neq 0$:

$$\frac{Z_B}{Z_A} = \frac{\int p^*_B(\mathbf{v})\mathrm{d}\mathbf{v}}{Z_A} = \int \frac{p^*_B(\mathbf{v})}{p^*_A(\mathbf{v})} p_A(\mathbf{v})\mathrm{d}\mathbf{v} = \mathrm{E}_{p_A}\left[\frac{p^*_B(\mathbf{v})}{p^*_A(\mathbf{v})}\right].$$

Assuming we can draw independent samples from p_A, the unbiased estimate of the ratio of partition functions can be obtained by using a simple Monte Carlo approximation:

$$\frac{Z_B}{Z_A} \approx \frac{1}{M}\sum_{i=1}^{M} \frac{p^*_B(\mathbf{v}^{(i)})}{p^*_A(\mathbf{v}^{(i)})} \equiv \frac{1}{M}\sum_{i=1}^{M} w^{(i)} = \hat{r}_{\mathrm{IS}}, \quad (8)$$

where $\mathbf{v}^{(i)} \sim p_A$. If p_A and p_B are not close enough, the estimator $\hat{r}_{\mathrm{IS}}$ will be very poor. In high-dimensional spaces, the variance of $\hat{r}_{\mathrm{IS}}$ will be very large (or possibly infinite), unless p_A is a near-perfect approximation to p_B.

3.1. Annealed Importance Sampling (AIS)

Suppose that we can define a sequence of intermediate probability distributions: $p_0, ..., p_K$, with $p_0 = p_A$ and $p_K = p_B$, which satisfy the following conditions:

- C1 $p_k(\mathbf{v}) \neq 0$ whenever $p_{k+1}(\mathbf{v}) \neq 0$.
- C2 We must be able to easily evaluate the unnormalized probability $p^*_k(\mathbf{v})$, $\forall \mathbf{v} \in \mathcal{V}$, $k = 0, ..., K$.
- C3 For each $k = 0, ..., K-1$, we must be able to draw a sample $\mathbf{v}'$ given $\mathbf{v}$ using a Markov chain transition operator $T_k(\mathbf{v}';\mathbf{v})$ that leaves $p_k(\mathbf{v})$ invariant:

$$\int T_k(\mathbf{v}';\mathbf{v}) p_k(\mathbf{v})\mathrm{d}\mathbf{v} = p_k(\mathbf{v}'). \quad (9)$$

- C4 We must be able to draw (preferably independent) samples from p_A.

The transition operators $T_k(\mathbf{v}';\mathbf{v})$ represent the probability density of transitioning from state $\mathbf{v}$ to $\mathbf{v}'$. Constructing a suitable sequence of intermediate probability distributions will depend on the problem. One general way to define this sequence is to set:

$$p_k(\mathbf{v}) \propto p_A^*(\mathbf{v})^{1-\beta_k} p_B^*(\mathbf{v})^{\beta_k}, \tag{10}$$

with $0 = \beta_0 < \beta_1 < ... < \beta_K = 1$ chosen by the user. Once the sequence of intermediate distributions has been defined we have:

Annealed Importance Sampling (AIS) run:

1. Generate $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_K$ as follows:
   - Sample $\mathbf{v}_1$ from $p_A = p_0$
   - Sample $\mathbf{v}_2$ given $\mathbf{v}_1$ using T_1
   - ...
   - Sample $\mathbf{v}_K$ given $\mathbf{v}_{K-1}$ using T_{K-1}
2. Set

$$w^{(i)} = \frac{p_1^*(\mathbf{v}_1)}{p_0^*(\mathbf{v}_1)} \frac{p_2^*(\mathbf{v}_2)}{p_1^*(\mathbf{v}_2)} ... \frac{p_{K-1}^*(\mathbf{v}_{K-1})}{p_{K-2}^*(\mathbf{v}_{K-1})} \frac{p_K^*(\mathbf{v}_K)}{p_{K-1}^*(\mathbf{v}_K)}$$

Note that there is no need to compute the normalizing constants of any intermediate distributions. After performing M runs of AIS, the importance weights $w^{(i)}$ can be substituted into Eq. 8 to obtain an estimate of the ratio of partition functions:

$$\frac{Z_B}{Z_A} \approx \frac{1}{M}\sum_{i=1}^{M} w^{(i)} = \hat{r}_{\text{AIS}}. \tag{11}$$

Neal (2005) shows that for sufficiently large number of intermediate distributions K, the variance of $\hat{r}_{\text{AIS}}$ will be proportional to $1/MK$. Provided K is kept large, the total amount of computation can be split in any way between the number of intermediate distributions K and the number of annealing runs M without adversely affecting the accuracy of the estimator. If samples drawn from p_A are independent, the number of AIS runs can be used to control the variance in the estimate of $\hat{r}_{\text{AIS}}$:

$$\text{Var}(\hat{r}_{\text{AIS}}) = \frac{1}{M}\text{Var}(w^{(i)}) \approx \frac{\hat{s}^2}{M} = \hat{\sigma}^2, \tag{12}$$

where $\hat{s}^2$ is estimated simply from the sample variance of the importance weights.

3.2. Ratios of Partition Functions of two RBM's

Suppose we have two RBM's with parameter values $\theta_A = \{W^A, \mathbf{b}^A, \mathbf{a}^A\}$ and $\theta_B = \{W^B, \mathbf{b}^B, \mathbf{a}^B\}$ that define probability distributions p_A and p_B over $\mathcal{V} \in \{0,1\}^D$. Each RBM can have a different number of hidden units $\mathbf{h}^A \in \{0,1\}^{M_A}$ and $\mathbf{h}^B \in \{0,1\}^{M_B}$. The generic AIS intermediate distributions (Eq. 10) would be harder to sample from than an RBM. Instead we introduce the following sequence of distributions for $k = 0, ..., K$:

$$p_k(\mathbf{v}) = \frac{p_k^*(\mathbf{v})}{Z_k} = \frac{1}{Z_k}\sum_{\mathbf{h}} \exp\left(-E_k(\mathbf{v}, \mathbf{h})\right), \tag{13}$$

where the energy function is given by:

$$E_k(\mathbf{v}, \mathbf{h}) = (1-\beta_k)E(\mathbf{v}, \mathbf{h}^A; \theta_A) + \beta_k E(\mathbf{v}, \mathbf{h}^B; \theta_B), \tag{14}$$

with $0 = \beta_0 < \beta_1 < ... < \beta_K = 1$. For $i = 0$, we have $\beta_0 = 0$ and so $p_0 = p_A$. Similarly, for $i = K$, we have $p_K = p_B$. For the intermediate values of k, we will have some interpolation between p_A and p_B.

Let us now define a Markov chain transition operator $T_k(\mathbf{v}'; \mathbf{v})$ that leaves $p_k(\mathbf{v})$ invariant. Using Eqs. 13, 14, it is straightforward to derive a block Gibbs sampler. The conditional distributions are given by logistic functions:

$$p(h_j^A = 1|\mathbf{v}) = \sigma\Big((1-\beta_k)(\sum_i W_{ij}^A v_i + a_j^A)\Big) \tag{15}$$

$$p(h_j^B = 1|\mathbf{v}) = \sigma\Big(\beta_k(\sum_i W_{ij}^B v_i + a_j^B)\Big) \tag{16}$$

$$p(v_i' = 1|\mathbf{h}) = \sigma\Big((1-\beta_k)(\sum_j W_{ij}^A h_j^A + b_i^A) + \beta_k(\sum_j W_{ij}^B h_j^B + b_i^B)\Big). \tag{17}$$

Given $\mathbf{v}$, Eqs. 15, 16 are used to stochastically activate hidden units $\mathbf{h}^A$ and $\mathbf{h}^B$. Eq. 17 is then used to draw a new sample $\mathbf{v}'$ as shown in Fig. 1 (left panel). Due to the special structure of RBM's, the cost of summing out $\mathbf{h}$ is linear in the number of hidden units. We can therefore easily evaluate:

$$\begin{aligned} p_k^*(\mathbf{v}) &= \sum_{\mathbf{h}^A, \mathbf{h}^B} e^{(1-\beta_k)E(\mathbf{v}, \mathbf{h}^A; \theta_A) + \beta_k E(\mathbf{v}, \mathbf{h}^B; \theta_B)} \\ &= e^{(1-\beta_k)\sum_i b_i^A v_i} \prod_{j=1}^{M_A} (1 + e^{(1-\beta_k)(\sum_i W_{ij}^A v_i + a_j^A)}) \\ &\times \; e^{\beta_k \sum_i b_i^B v_i} \prod_{j=1}^{M_B} (1 + e^{\beta_k(\sum_i W_{ij}^B v_i + a_j^B)}). \end{aligned}$$

We will assume that the parameter values of each RBM satisfy $|\theta| < \infty$, in which case $p(\mathbf{v}) > 0$ for all $\mathbf{v} \in \mathcal{V}$. This will ensure that condition C1 of the AIS procedure is always satisfied. We have already shown that conditions C2 and C3 are satisfied. For condition C4, we can run a blocked Gibbs sampler (Eqs. 5, 6) to generate samples from p_A. These sample points will not be independent, but the AIS estimator will still converge to the correct value, provided our Markov chain is ergodic (Neal, 2001). However, assessing the accuracy of this estimator can be difficult, as it depends on both the variance of the importance weights and on autocorrelations in the Gibbs sampler.

3.3. Estimating Partition Functions of RBM's

The partition function of an RBM can be found by finding the ratio to the normalizer for $\theta_A = \{0, \mathbf{b}^A, \mathbf{a}^A\}$, an RBM

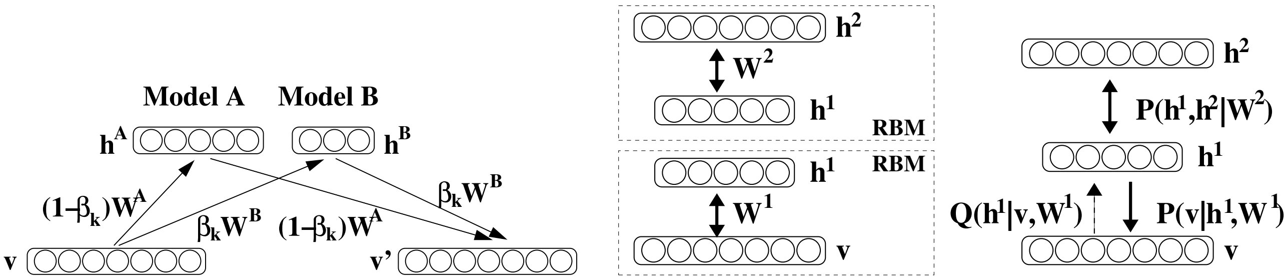

Figure 1. **Left:** The Gibbs transition operator $T_k(\mathbf{v}';\mathbf{v})$ leaves $p_k(\mathbf{v})$ invariant when estimating the ratio of partition functions Z_B/Z_A. **Middle:** Recursive greedy learning consists of learning a stack of RBMs. **Right:** Two-layer DBN as a generative model.

with a zero weight matrix. From Eq. 3, we know:

$$Z_A = 2^{M_A}\prod_i(1+e^{b_i}). \tag{18}$$

Moreover,

$$p_A(\mathbf{v}) = \prod_i p_A(v_i) = \prod_i 1/(1+e^{-b_i}),$$

so we can draw exact independent samples from this "base-rate" RBM. AIS in this case allows us to obtain an *unbiased* estimate of the partition function Z_B. This approach closely resembles simulated annealing, since the intermediate distributions of Eq. 13 take form:

$$p_k(\mathbf{v}) = \frac{\exp((1-\beta_k)\mathbf{v}^T\mathbf{b}^A)}{Z_k}\sum_{\mathbf{h}^B}\exp(-\beta_k E(\mathbf{v},\mathbf{h}^B;\theta_B)).$$

We gradually change β_k (or inverse temperature) from 0 to 1, annealing from a simple "base-rate" model to the final complex model. The importance weights $w^{(i)}$ ensure that AIS produces correct estimates.

4. Deep Belief Networks (DBN's)

In this section we briefly review a greedy learning algorithm for training Deep Belief Networks. We then show how to obtain an estimate of the lower bound on the log-probability that the DBN assigns to the data.

4.1. Greedy Learning of DBN's

Consider learning a DBN with two layers of hidden features as shown in Fig. 1 (right panel). The greedy strategy developed by Hinton et al. (2006) uses a stack of RBM's (Fig. 1, middle panel). We first train the bottom RBM with parameters W^1, as described in section 2.

A key observation is that the RBM's joint distribution $p(\mathbf{v},\mathbf{h}^1|W^1)$ is identical to that of a DBN with second-layer weights tied to $W^2 = {W^1}^\top$. We now consider untying and refining W^2, improving the fit to the training data.

For any approximating distribution $Q(\mathbf{h}^1|\mathbf{v})$, the DBN's log-likelihood has the following variational lower bound:

$$\ln p(\mathbf{v}|W^1,W^2) \;\geq\; \sum_{\mathbf{h}^1} Q(\mathbf{h}^1|\mathbf{v})\big[\ln p(\mathbf{h}^1|W^2) \;+\; \ln p(\mathbf{v}|\mathbf{h}^1,W^1)\big] + \mathcal{H}(Q(\mathbf{h}^1|\mathbf{v})), \tag{19}$$

where $\mathcal{H}(\cdot)$ is the entropy functional. We set $Q(\mathbf{h}^1|\mathbf{v}) = p(\mathbf{h}^1|\mathbf{v},W^1)$ defined by the RBM (Eq. 5). Initially, when $W^2 = {W^1}^\top$, Q is the DBN's true factorial posterior over $\mathbf{h}^1$, and the bound is tight. Therefore, any increase in the bound will lead to an increase in the true likelihood of the model. Maximizing the bound of Eq. 19 with frozen W^1 is equivalent to maximizing:

$$\sum_{\mathbf{h}^1} Q(\mathbf{h}^1|\mathbf{v})\ln p(\mathbf{h}^1|W^2). \tag{20}$$

This is equivalent to training the second layer RBM with vectors drawn from $Q(\mathbf{h}^1|\mathbf{v})$ as data.

This scheme can be extended by training a third RBM on $\mathbf{h}^2$ vectors drawn from the second RBM. If we initialize $W^3 = {W^2}^\top$, we are guaranteed to improve the lower bound on the log-likelihood, though the log-likelihood itself can fall (Hinton et al., 2006). Repeating this greedy, layer-by-layer training several times results in a deep, hierarchical model.

Recursive Greedy Learning Procedure for the DBN.

1. Fit parameters W^1 of a 1-layer RBM to data.
2. Freeze the parameter vector W^1 and use samples from $p(\mathbf{h}^1|\mathbf{v},W^1)$ as the data for training the next layer of binary features with an RBM.
3. Proceed recursively for as many layers as desired.

In practice, when adding a new layer l, we typically do not initialize $W^l = {W^{l-1}}^\top$, so the number of hidden units of the new RBM does not need to be the same as the number of the visible units of the lower-level RBM.

4.2. Estimating Lower Bounds for DBN's

Consider the same DBN model with two layers of hidden features shown in Fig. 1. The model's joint distribution is:

$$p(\mathbf{v},\mathbf{h}^1,\mathbf{h}^2) = p(\mathbf{v}|\mathbf{h}^1)\,p(\mathbf{h}^2,\mathbf{h}^1), \tag{21}$$

where $p(\mathbf{v}|\mathbf{h}^1)$ is defined by Eq. 6), and $p(\mathbf{h}^1,\mathbf{h}^2)$ is the joint distribution defined by the second layer RBM. Note that $p(\mathbf{v}|\mathbf{h}^1)$ is normalized.

By explicitly summing out $\mathbf{h}^2$, we can easily evaluate an unnormalized probability $p^*(\mathbf{v},\mathbf{h}^1)=Zp(\mathbf{v},\mathbf{h}^1)$. Using the approximating factorial distribution Q, which we get as a byproduct of the greedy learning procedure, and the variational lower bound of Eq. 19, we obtain:

$$\ln \sum_{\mathbf{h}^1} p(\mathbf{v},\mathbf{h}^1) \geq \sum_{\mathbf{h}^1} Q(\mathbf{h}^1|\mathbf{v}) \ln p^*(\mathbf{v},\mathbf{h}^1) \\ - \ln Z + \mathcal{H}(Q(\mathbf{h}^1|\mathbf{v})) = B(\mathbf{v}). \quad (22)$$

The entropy term $\mathcal{H}(\cdot)$ can be computed analytically, since Q is factorial. The partition function Z is estimated by running AIS on the top-level RBM. And the expectation term can be estimated by a simple Monte Carlo approximation:

$$\sum_{\mathbf{h}^1} Q(\mathbf{h}^1|\mathbf{v}) \ln p^*(\mathbf{v},\mathbf{h}^1) \approx \frac{1}{M}\sum_{i=1}^{M} \ln p^*(\mathbf{v},\mathbf{h}^{1(i)}), \quad (23)$$

where $\mathbf{h}^{1(i)} \sim Q(\mathbf{h}^1|\mathbf{v})$. The variance of this Monte Carlo estimator will be proportional to $1/M$ provided the variance of $\ln p^*(\mathbf{v},\mathbf{h}^{1(i)})$ is finite. In general, we will be interested in calculating the lower bound averaged over the test set containing N_t samples, so

$$\frac{1}{N_t}\sum_{n=1}^{N_t} B(\mathbf{v}^n) \approx \frac{1}{N_t}\sum_{n=1}^{N_t}\left[\frac{1}{M}\sum_{i=1}^{M} \ln p^*(\mathbf{v}^n,\mathbf{h}^{1(i)}) \; + \right. \\ \left. \mathcal{H}(Q(\mathbf{h}^1|\mathbf{v}^n))\right] - \ln \hat{Z} = \hat{r}_B - \ln \hat{Z} = \hat{r}_{\text{Bound}}. \; (24)$$

In this case the variance of the estimator induced by the Monte Carlo approximation will asymptotically scale as $1/(N_t M)$. We will show in the experimental results section that the value of M can be small provided N_t is large.

The error of the overall estimator $\hat{r}_{\text{Bound}}$ in Eq. 24 will be mostly dominated by the error in the estimate of $\ln Z$. In our experiments, we obtained unbiased estimates of $\hat{Z}$ and its standard deviation $\hat{\sigma}$ using Eqs. 11, 12. We report $\ln \hat{Z}$ and $\ln(\hat{Z} \pm \hat{\sigma})$.

Estimating this lower bound for Deep Belief Networks with more layers is now straightforward. Consider a DBN with L hidden layers. The model's joint distribution and its approximate posterior distribution Q are given by:

$$p^(\mathbf{v},\mathbf{h}^1,...,\mathbf{h}^L) = p(\mathbf{v}|\mathbf{h}^1)...p(\mathbf{h}^{L-2}|\mathbf{h}^{L-1})p(\mathbf{h}^{L-1},\mathbf{h}^L)$$
$$Q(\mathbf{h}^1,...,\mathbf{h}^L|\mathbf{v}) = Q(\mathbf{h}^1|\mathbf{v})Q(\mathbf{h}^2|\mathbf{h}^1)...Q(\mathbf{h}^L|\mathbf{h}^{L-1}).$$

The bound can now be obtained by using Eq. 22. Note that most of the computation resources will be spent on estimating the partition function Z of the top level RBM.

5. Experimental Results

In our experiments we used the MNIST digit dataset, which contains 60,000 training and 10,000 test images of ten handwritten digits (0 to 9), with 28×28 pixels. The dataset was binarized: each pixel value was stochastically set to 1 in proportion to its pixel intensity. Samples from the training set are shown in Fig. 2 (top left panel). Annealed importance sampling requires the β_k that define a sequence of intermediate distributions. In all of our experiments this sequence was chosen by quickly running a few preliminary experiments and picking the spacing of β_k so as to minimize the log variance of the final importance weights. The biases $\mathbf{b}^A$ of a base-rate model (see Eq. 18) were set by maximum likelihood, then smoothed to ensure that $p(\mathbf{v}) > 0, \forall\, \mathbf{v} \in \mathcal{V}$. Code that can be used to reproduce experimental results is available at www.cs.toronto.edu/∼rsalakhu.

5.1. Estimating partition functions of RBM's

In our first experiment we trained three RBM's on the MNIST digits. The first two RBM's had 25 hidden units and were learned using CD (section 2) with T=1 and T=3 respectively. We call these models CD1(25) and CD3(25). The third RBM had 20 hidden units and was learned using CD with T=1. For all three models we can calculate the exact value of the partition function simply by summing out the 784 visible units for each configuration of the hiddens. For all three models we used 500 β_k spaced uniformly from 0 to 0.5, 4,000 β_k spaced uniformly from 0.5 to 0.9, and 10,000 β_k spaced uniformly from 0.9 to 1.0, with a total of 14,500 intermediate distributions.

Table 1 shows that for all three models, using only 10 AIS runs, we were able to obtain good estimates of partition functions in just 20 seconds on a Pentium Xeon 3.00GHz machine. For model CD1(25), however, the variance of the estimator was high, even with 100 AIS runs. However, figure 3 (top row) reveals that as the number of annealing runs is increased, AIS can almost exactly recover the true value of the partition function across all three models.

We also estimated the ratio of normalizing constants of two RBM's that have different numbers of hidden units: CD1(20) and CD1(25). This estimator could be used to do complexity control. In detail, using 100 AIS runs with uniform spacing of 10,000 β_k, we obtained $\ln \hat{r}_{\text{AIS}} = \ln(Z_{\text{CD1(20)}}/Z_{\text{CD1(25)}}) = -24.49$ with an error estimate $\ln(\hat{r}_{\text{AIS}} \pm 3\hat{\sigma}) = (-24.19, -24.93)$. Each sample from CD1(25) was generated by starting a Markov chain at the previous sample and running it for 10,000 steps. Compared to the true value of -24.18, this result suggests that our estimates may have a small systematic error due to the Markov chain failing to visit some modes.

Our second experiment consisted of training two more realistic models: CD1(500) and CD3(500). We used exactly the same spacing of β_k as before and exactly the same base-rate model. Results are shown in table 1 (bottom row). For each model we were able to get what appears to be a rather

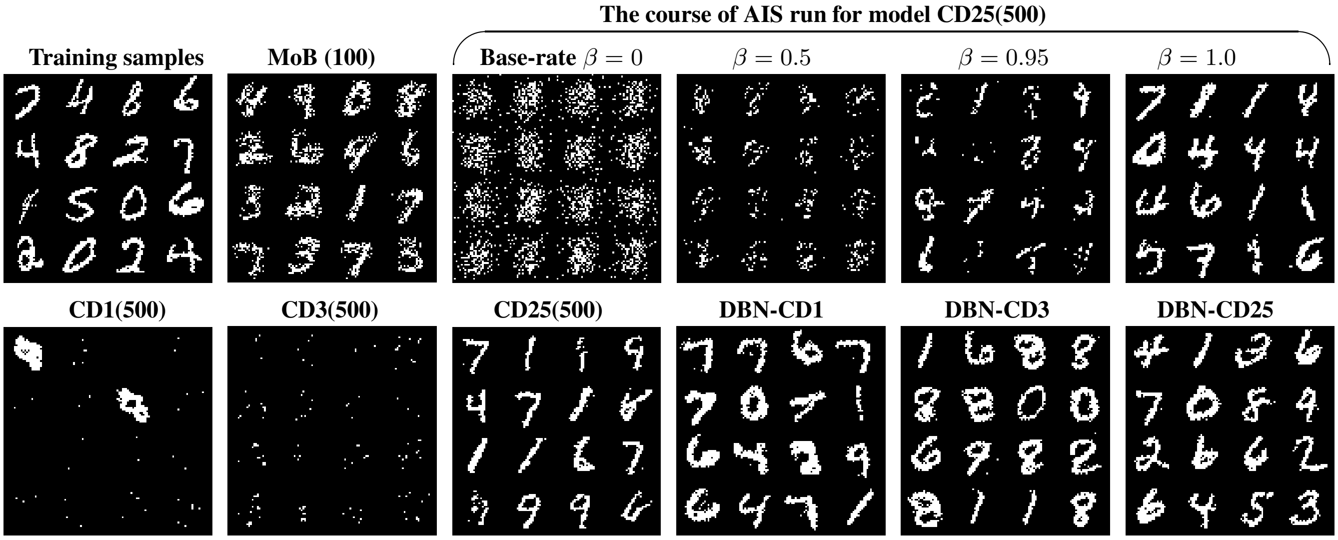

Figure 2. **Top row:** First two panels show random samples from the training set and a mixture of Bernoullis model with 100 components. The last 4 panels display the course of 16 AIS runs for CD25(500) model by starting from a simple base-rate model and annealing to the final complex model. **Bottom row:** Random samples generated from three RBM's and corresponding three DBN's models.

Table 1. Results of estimating partition functions of RBM's along with the estimates of the average training and test log-probabilities. For all models we used 14,500 intermediate distributions.

AIS Runs		True $\ln Z$	Estimates $\ln \hat{Z}$	$\ln(\hat{Z} \pm \hat{\sigma})$	$\ln(\hat{Z} \pm 3\hat{\sigma})$	Time (mins)	Avg. Test log-prob. true	estimate	Avg. Train log-prob. true	estimate
100	CD1(25)	255.41	256.52	255.00, 257.10	0.0000, 257.73	3.3	−151.57	−152.68	−152.35	−153.46
	CD3(25)	307.47	307.63	307.44, 307.79	306.91, 308.05	3.3	−143.03	−143.20	−143.94	−144.11
	CD1(20)	279.59	279.57	279.43, 279.68	279.12, 279.87	3.1	−164.52	−164.50	−164.89	−164.87
100	CD1(500)	—	350.15	350.04, 350.25	349.77, 350.42	10.4	—	−125.53	—	−122.86
	CD3(500)	—	280.09	279.99, 280.17	279.76, 280.33	10.4	—	−105.50	—	−102.81
	CD25(500)	—	451.28	451.19, 451.37	450.97, 451.52	10.4	—	−86.34	—	−83.10

accurate estimate of Z. Of course, we are relying on an empirical estimate of AIS's accuracy, which could potentially be misleading. Nonetheless, Fig. 3 (bottom row) shows that as we increase the number of annealing runs, the value of the estimator does not oscillate drastically.

While performing these tests, we observed that contrastive divergence learning with T=3 results in considerably better generative model than CD learning with T=1: the difference of 20 nats is striking! Clearly, the widely used practice of CD learning with T=1 is a rather poor "substitute" for maximum likelihood learning. Inspired by this result, we trained a model by starting with T=1, and gradually increasing T to 25 during the course of CD training, as suggested by (Carreira-Perpinan & Hinton, 2005). We call this model CD25(500). Training this model was computationally much more demanding. However, the estimate of the average test log-probability for this model was about -86, which is 39 and 19 nats better than the CD1(500) and CD3(500) models respectively. Fig. 2 (bottom row) shows samples generated from all three models by randomly initializing binary states of the visible units and running alternating Gibbs for 100,000 steps. Certainly, samples generated by CD25(500) look much more like the real handwritten digits, than either CD1(500) or CD3(500).

We also obtained an estimate of the log ratio of two partition functions $\hat{r}_{\text{AIS}} = \ln Z_{\text{CD25(500)}}/Z_{\text{CD3(500)}} = 169.96$, using 10,000 β_k and 100 annealing runs. The estimates of the individual log-partition functions were $\ln \hat{Z}_{\text{CD25(500)}} = 451.28$ and $\ln \hat{Z}_{\text{CD3(500)}} = 280.09$, in which case the log ratio is $451.28 - 280.09 = 171.19$. This is in agreement (to within three standard deviations) with the direct estimate of the ratio, $\hat{r}_{\text{AIS}} = 169.96$.

For a simple comparison we also trained several mixture of Bernoullis models (see Fig. 2, top left panel) with 10, 100, and 500 components. The corresponding average test log-probabilities were -168.95, -142.63, and -137.64. The data generated from the mixture model looks better than CD3(500), although our quantitive results reveal this is due to over-fitting. The RBM's make much better predictions.

5.2. Estimating lower bounds for DBN's

We greedily trained three DBN models with two hidden layers. The first model, called DBN-CD1, was greedily

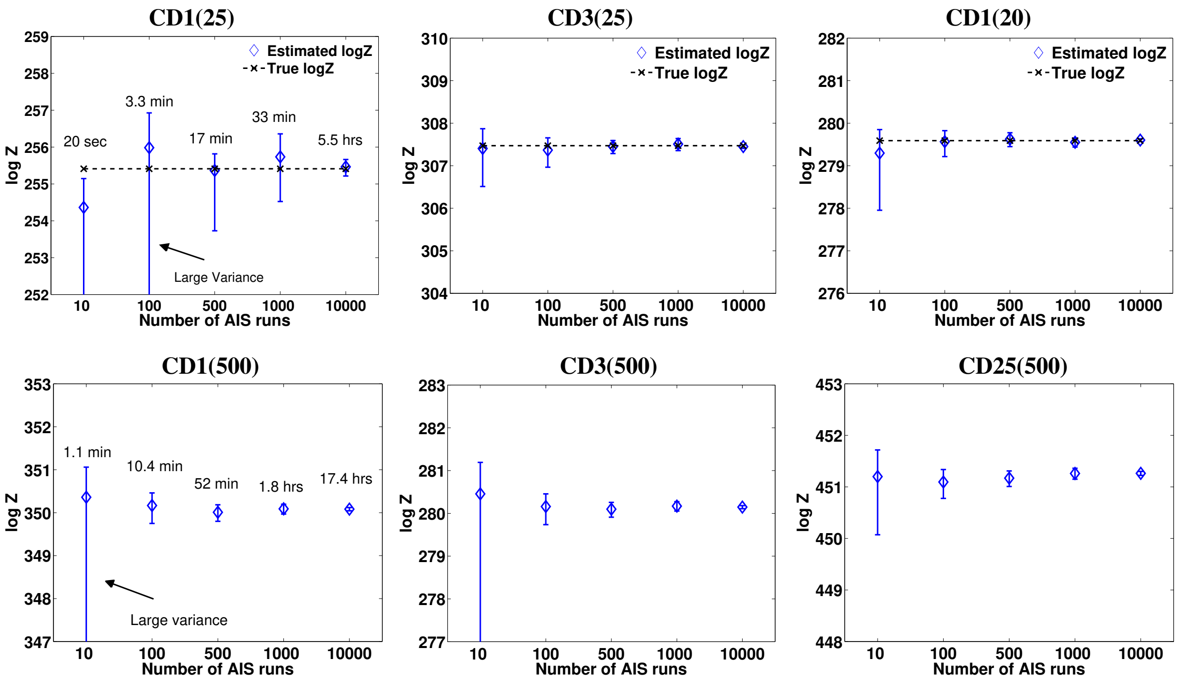

Figure 3. Estimates of the log-partition functions $\ln \hat{Z}$ as we increase the number of annealing runs. The error bars show $\ln(\hat{Z} \pm 3\hat{\sigma})$.

learned by freezing the parameter vector of the CD1(500) model and learning the 2^{nd} layer RBM with 2000 hidden units using CD with T=1. Similarly, the other two models, DBN-CD3 and DBN-CD25, added 2000 hidden units on top of CD3(500) and CD25(500), using CD with T=3 and T=25 respectively. Training the DBN's took roughly three times longer than the RBM's.

Table 2 shows the results. We used 15,000 intermediate distributions and 500 annealing runs to estimate the partition function of the 2^{nd} layer RBM. This took 2.3 hours. Further sampling was required for the simple Monte Carlo approximation of Eq. 23. We used M=5 samples from the approximating distribution $Q(\mathbf{h}|\mathbf{v})$ for each data vector $\mathbf{v}$. Setting M=100 did not make much difference. Table 2 also reports the empirical error in the estimate of the lower bound $\hat{r}_{\text{Bound}}$. From Eq. 24, we have $\text{Var}(\hat{r}_{\text{Bound}}) = \text{Var}(\hat{r}_B) + \text{Var}(\ln \hat{Z})$, both of which are shown in table 2. Note that models DBN-CD1 and DBN-CD3 significantly outperform their single layer counterparts: CD1(500) and CD3(500). Adding a second layer for those two models improves model performance by at least 25 and 7 nats. This corresponds to a dramatic improvement in the quality of samples generated from the models (Fig. 2, bottom row).

Observe that greedy learning of DBN's does not appear to suffer severely from overfitting. For single layer models, the difference between the estimates of training and test log-probabilities was about 3 nats. For DBN's, the corresponding difference in the estimates of the lower bounds was about 4 nats, even though adding a second layer introduced over twice as many (or one million) new parameters.

Table 2. Results of estimating lower bounds $\hat{r}_{\text{Bound}}$ (Eq. 24) on the average training and test log-probabilities for DBN's. On average, the total error of the estimator is about $\pm$ 2 nats.

	Model	Avg. bound log-prob	Error $\hat{r}_B$ ± 3 std	AIS error $\ln(\hat{Z} \pm 3\hat{\sigma}) - \ln \hat{Z}$
Test	DBN-CD1	-100.64	± 0.77	$-1.43, +0.57$
	DBN-CD3	-98.29	± 0.75	$-0.91, +0.31$
	DBN-CD25	-86.22	± 0.67	$-0.84, +0.65$
Train	DBN-CD1	-97.67	± 0.30	$-1.43, +0.57$
	DBN-CD3	-94.86	± 0.29	$-0.91, +0.31$
	DBN-CD25	-82.47	± 0.25	$-0.84, +0.65$

The result of our experiments for DBN-CD25, however, was very different. For this model, on the test data we obtained $\hat{r}_{\text{Bound}} = -86.22$. This is comparable to the estimate of -86.34 for the average test log-probability of the CD25(500) model. Clearly, we cannot confidently assert that DBN-CD25 is a better generative model compared to the carefully trained single layer RBM. This peculiar result also supports previous claims that if the first level RBM already models data well, adding extra layers will not help (LeRoux & Bengio, 2008; Hinton et al., 2006). As an additional test, instead of randomly initializing parameters of the 2^{nd} layer RBM, we initialized it by using the same parameters as the 1^{st} layer RBM but with hidden and visible units switched (see Fig. 1). This initialization ensures that the distribution over the visible units $\mathbf{v}$ defined by the two-layer DBN is *exactly the same* as the distribution over $\mathbf{v}$ defined by the 1^{st} layer RBM. Therefore, after learning parameters of the 2^{nd} layer RBM, the lower bound on the training data log-likelihood can only improve. After carefully training the second level RBM, our estimate of the lower bound on the test log-probability was only -85.97. Once again, we cannot confidently claim that adding an extra layer in this case yields better generalization.

6. Discussions

The original paper of Hinton et al. (2006) showed that for DBN's, each additional layer increases a lower bound (see Eq. 19) on the log-probability of the *training* data, provided the number of hidden units per layer does not decrease. However, assessing generalization performance of these generative models is quite difficult, since it requires enumeration over an exponential number of terms. In this paper we developed an annealed importance sampling procedure that takes advantage of the bipartite structure of the RBM. This can provide a good estimate of the partition function in a reasonable amount of computer time. Furthermore, we showed that this estimator, along with approximate inference, can be used to obtain an estimate of the lower bound on the log-probability of the *test* data, thus allowing us to obtain some quantitative evaluation of the generalization performance of these deep hierarchical models.

There are some disadvantages to using AIS. There is a need to specify the β_k that define a sequence of intermediate distributions. The number and the spacing of β_k will be problem dependent and will affect the variance of the estimator. We also have to rely on the empirical estimate of AIS accuracy, which could potentially be very misleading (Neal, 2001; Neal, 2005). Even though AIS provides an unbiased estimator of Z, it occasionally gives large overestimates and usually gives small underestimates, so in practice, it is more likely to underestimate of the true value of the partition function, which will result in an overestimate of the log-probability. But these drawbacks should not result in disfavoring the use of AIS for RBM's and DBN's: it is much better to have a slightly unreliable estimate than no estimate at all, or an extremely indirect estimate, such as discriminative performance (Hinton et al., 2006).

We find AIS and other stochastic methods attractive as they can just as easily be applied to undirected graphical models that generalize RBM's and DBN's to exponential family distributions. This will allow future application to models of real-valued data, such as image patches (Osindero & Hinton, 2008), or count data (Gehler et al., 2006).

Another alternative would be to employ deterministic approximations (Yedidia et al., 2005) or deterministic upper bounds (Wainwright et al., 2005) on the log-partition function. However, for densely connected MRF's, we would not expect these methods to work well. Indeed, preliminary results suggest that these methods provide quite inaccurate estimates of (or very loose upper bounds on) the partition function, even for small RBM's when *trained on real data*.

Acknowledgments

We thank Geoffrey Hinton and Radford Neal for many helpful suggestions. This research was supported by NSERC and CFI. Iain Murray is supported by the government of Canada.

References

Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. *Large-Scale Kernel Machines*. MIT Press.

Carreira-Perpinan, M., & Hinton, G. (2005). On contrastive divergence learning. *10th Int. Workshop on Artificial Intelligence and Statistics (AISTATS'2005)*.

Gehler, P., Holub, A., & Welling, M. (2006). The Rate Adapting Poisson (RAP) model for information retrieval and object recognition. *Proceedings of the 23rd International Conference on Machine Learning*.

Hinton, & Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*, 504 – 507.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*, 1711–1800.

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*, 1527–1554.

LeRoux, N., & Bengio, Y. (2008). Representational power of restricted Boltzmann machines and deep belief networks. *To appear in Neural Computation*.

Neal, R. M. (1993). *Probabilistic inference using Markov chain Monte Carlo methods* (Technical Report CRG-TR-93-1). Department of Computer Science, University of Toronto.

Neal, R. M. (2001). Annealed importance sampling. *Statistics and Computing*, *11*, 125–139.

Neal, R. M. (2005). *Estimating ratios of normalizing constants using linked importance sampling* (Technical Report 0511). Department of Statistics, University of Toronto.

Osindero, S., & Hinton, G. (2008). Modeling image patches with a directed hierarchy of Markov random fields. *NIPS 20*. Cambridge, MA: MIT Press.

Salakhutdinov, R., Mnih, A., & Hinton, G. (2007). Restricted Boltzmann machines for collaborative filtering. *Proceedings of the Twenty-fourth International Conference (ICML 2004)*.

Skilling, J. (2004). Nested sampling. *Bayesian inference and maximum entropy methods in science and engineering, AIP Conference Proceeedings*, *735*, 395–405.

Taylor, G. W., Hinton, G. E., & Roweis, S. T. (2006). Modeling human motion using binary latent variables. *Advances in Neural Information Processing Systems*. MIT Press.

Wainwright, M. J., Jaakkola, T., & Willsky, A. S. (2005). A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, *51*, 2313–2335.

Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2005). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, *51*, 2282–2312.