Deep networks for robust visual recognition

Yichuan Tang Y3TANG@UWATERLOO.CA
Chris Eliasmith CELIASMITH@UWATERLOO.CA
Centre for Theoretical Neuroscience, University of Waterloo, Waterloo ON N2L 3G1 CANADA

Abstract

Deep Belief Networks (DBNs) are hierarchical generative models which have been used successfully to model high dimensional visual data. However, they are not robust to common variations such as occlusion and random noise. We explore two strategies for improving the robustness of DBNs. First, we show that a DBN with sparse connections in the first layer is more robust to variations that are not in the training set. Second, we develop a probabilistic denoising algorithm to determine a subset of the hidden layer nodes to unclamp. We show that this can be applied to any feedforward network classifier with localized first layer connections. Recognition results after denoising are significantly better over the standard DBN implementations for various sources of noise.

1. Introduction

Deep Belief Networks (DBNs) are hierarchical generative models with many latent variables that effectively model high dimensional visual image data (Hinton et al., 2006). A DBN is trained by a greedy layer-by-layer unsupervised learning algorithm on a series of bipartite Markov Random Field (MRF) known as a Restricted Boltzmann Machine (RBM). Fine tuning by the up-down algorithm or discriminative optimization results in a deep network capable of fast feed-forward classification (Hinton & Salakhutdinov, 2006; Salakhutdinov & Hinton, 2009).

DBNs model all the pixels in the visible layer probabilistically, and as a result, are not robust to images with "noise" which are not in the training set. We include occlusions, additive noise, and "salt" noise in our definition of noise in this paper.

To improve the robustness of the DBN, we introduce a modified version of the DBN termed a sparse DBN (sDBN) where the first layer is sparsely (and locally) connected. This is in part inspired by the properties of the human visual system. It is well-established that the lower cortical levels represent the visual input in a local, sparsely connected topographical manner (Hubel & Wiesel, 1959). We show that a sDBN is more robust to noise on the MNIST (LeCun et al., 1998) dataset with noise added to the test images. We then present a denoising algorithm which combines top-down and bottom-up inputs to "fill in" the subset of hidden layer nodes which are most affected by noise. (Lee & Mumford, 2003) proposed that the human visual cortex performs hierarchical Bayesian inference where "beliefs" are propagated up and down the hierarchy. Our attention-esque top-down feedback can be thought of as a type of "belief" that helps to identify object versus non-object (noise) elements in the visible layer.

2. Related Work

Sparsely connected weights have been widely used in visual recognition algorithms (Fukushima, 1983; LeCun et al., 1998; Serre et al., 2005). Most of these algorithms contain a max-pooling stage following a convolutional stage to provide a certain amount of translational and scale invariance. Recently there has been work combining the convolutional approach with the DBN (Lee et al., 2009; Norouzi et al., 2009). These efforts enforce sparse connections similar in spirit to those enforced here. However, unlike those methods, our main motivation is not to provide translational invariance and/or to reduce the number of model parameters, but rather to diminish the effect of noise on the activations of hidden layer nodes. In addition, our algorithm does not require weight sharing (applying the same filter across an image), which would increase the total number of hidden layer nodes and increase the computational complexity of our denoising algo-

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010. Copyright 2010 by the author(s)/owner(s).

rithm.

(Welling et al., 2002; Roth & Black, 2005) also learned MRFs to model the prior statistics of images for denoising and inpainting. Whereas those methods model at the pixel level and explicitly specify a noise likelihood, our proposed algorithm uses the prior over the first hidden layer to estimate the subset of nodes which are affected by noise. This allows the method to be agnostic about the noise likelihood distribution.

Finally, we evaluate our methods on the widely-used MNIST handwritten digit classification task, where the state-of-the-art performance is currently 0.53% (Jarrett et al., 2009) for domain knowledge based methods and 0.95% (Salakhutdinov & Hinton, 2009) for permutation invariant methods.

3. Sparsely Connected DBN

While the first layer weights of a standard DBN are somewhat spatially localized, they are not forced to be zero past a given radius. Consequently, the small but significant weight values affect a given hidden node's activation if any noise are present anywhere in the image. Classification results are likewise affected, making DBNs not robust to various types of noise. For instance, figure 1 gives examples of noisy images and their respective classification errors of a DBN. This DBN was trained according to (Hinton et al., 2006), followed by 30 epochs of discriminative optimization and achieves 1.03% test error on the clean images. However, error dramatically increased under various types of noise.

These particular kinds of noise were chosen to reflect various possible sources of error for which biological visual systems are robust. The first is the simple introduction of a border that does not overlap with the foreground of the digits. The second is the occlusion by a rectangular block random in size and location. The third is the corruption of the images by random noise.

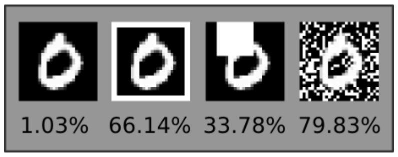


Figure 1. DBN fails to be robust to various noise. The noise include added borders, randomly placed occluding rectangles, and random pixels toggled to white.

The poor classification performance on the noisy test images is expected since a DBN models the joint probability of 28x28 = 784 pixels, all with black borders. Therefore, test images with white borders are not probable and the ensuing classification is not very accurate. Of course, when images with these variations were to added to the training set, we obtained better recognition results. However, due to the impractical nature of adding all possible noise that might exist in a real world environment, it is desirable to have a DBN which is more robust to out-of-sample test data before resorting to enlarging the training set.

3.1. Why Sparseness

In this paper we use V, H^1, H^2, H^3 to refer to each of the layers (see figure 3), and $V = \mathbf{v}$ to denote a specific activation of layer V. We will also use $q(\cdot)$ to refer to the approximate posterior computed by the recognition weights. Specifically, $q(\mathbf{h}^1|\mathbf{v}) = \sigma\big((\mathbf{W}^1_{rec})^\intercal \mathbf{v} + \mathbf{c}\big)$ and $\sigma(\cdot)$ is the logistic function.

We improve the robustness of the DBN by first reducing the effect that a noisy image $V = \tilde{\mathbf{v}}$ has on the hidden layer activation $q(\mathbf{h}^1|\tilde{\mathbf{v}})$. We accomplish this by specifying sparse connections between each hidden layer node and a spatially local group of visible layer nodes in the first RBM. We use sRBM to refer to this even more restricted type of RBM. For example, each $\mathbf{h}^1$ node is randomly assigned a 7x7 receptive field (RF), and it has no connections to visible nodes outside of its RF. With local connections, noise or occlusion in one subset of V nodes will only affect a subset of H^1 nodes. The main motivation here is to reduce the change between H^1 activation given the noisy image, $q(\mathbf{h}^1|\tilde{\mathbf{v}})$, and H^1 activation given the original image, $q(\mathbf{h}^1|\mathbf{v})$.

3.2. Sparse RBM Learning

The basic building block of a DBN is the RBM. A full account of RBM training and DBN formation is described in (Hinton et al., 2006; Bengio et al., 2007).

An RBM with visible layer nodes $V = \mathbf{v}$ and hidden layer nodes $H = \mathbf{h}$ is defined by an energy function

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\mathbf{b}^\intercal \mathbf{v} - \mathbf{c}^\intercal \mathbf{h} - \mathbf{v}^\intercal \mathbf{W} \mathbf{h} \qquad (1)$$

where $\theta = \{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$ are the model parameters. The probability distribution of the system $\{\mathbf{v}, \mathbf{h}\}$ can be written as:

$$p(\mathbf{v}, \mathbf{h}) = \frac{p^*(\mathbf{v}, \mathbf{h})}{Z(\theta)} = \frac{\exp^{-E(\mathbf{v},\mathbf{h})}}{Z(\theta)} \qquad (2)$$

where $Z(\theta)$ is the normalization constant: $Z(\theta) = \sum_{\mathbf{v},\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}$. Exact maximum likelihood learning is intractable due to the computation of an expectation w.r.t. the model's distribution. In practice, learning is often performed using n-step Contrastive Divergence (CD) (Hinton, 2002), where the weights are updated as:

$$\Delta W_{ij} \propto \mathbb{E}_{data}[v_i h_j] - \mathbb{E}_{recons}[v_i h_j] \tag{3}$$

$\mathbb{E}_{recons}[\cdot]$ represent the expectation w.r.t. the distribution after n steps of block Gibbs sampling starting at the data[1].

When learning a sRBM, the only modification needed is to zero out the weights connecting each hidden node to visible nodes that are outside of its RF:

$$\Delta W_{ij} \propto (\mathbb{E}_{data}[v_i h_j] - \mathbb{E}_{recons}[v_i h_j])\widetilde{W}_{ij} \tag{4}$$

where

$$\widetilde{W}_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is in } h_j\text{'s RF} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Additional computational efficiency can be obtained by using sparse matrix operations for learning and inference.

3.3. Sparse RBM Evaluation

As the RF approaches 28x28 (the dimension of the visible layer for the MNIST digits), the sRBM approaches the standard RBM. Using a 7x7 RF instead of the standard 28x28 reduces the number of weights for the first layer RBM by a factor of 16. Certainly a concern is whether or not this sRBM would still be a good generative model of the data. To find the average log probability of the test set, we estimated the normalization constant $Z(\theta)$ for each sRBM by using the Annealed Importance Sampling algorithm (Neal, 2001). Following (Salakhutdinov & Murray, 2008), we performed 100 annealing runs using around 15,000 intermediate distributions.

Table 1 shows the estimated average test log probability for various sparse RBMs. It also shows the error rate of the DBNs built from these sparse RBMs (section 3.4). The log probability is positively correlated with RF size and the number of hidden layer nodes. While not shown on this table, it is worth noting that the best 12x12 sRBM achieves a log probability that is only about 3 nats below an equivalently trained standard RBM. In addition, the worst sRBM considered here (7x7, 500 hidden nodes), is still about 11 nats better than a standard RBM trained using 3-step CD (Salakhutdinov & Murray, 2008). Figure 2 shows some of the filters learned by a 7x7 sRBM on MNIST.

[1]In our experiments, we use 25-step CD for sRBM training, with a learning rate of 0.1 for 50 epochs.

Table 1. Sparse RBM and sparse DBN evaluations

RF size	# hidden	log probability	sDBN error
7x7	500	-94.62	1.19%
	1000	-92.50	1.20%
	1500	-91.77	1.60%
10x10	500	-91.53	1.17%
	1000	-90.16	1.24%
	1500	-89.78	1.55%
12x12	500	-90.30	1.18%
	1000	-89.72	1.16%
	1500	-89.56	1.63%

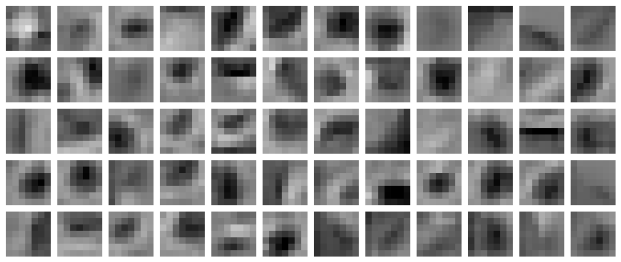


Figure 2. Filters from a sRBM with 7x7 RF learned on MNIST.

3.4. Sparse DBN

A DBN can be constructed with a hierarchical series of RBMs. We train our 2nd level RBM in the standard way and allow for full connectivity between layers H^1 and H^2. A greedy layer-wise training procedure is used where $q(\mathbf{h}^1|\mathbf{v})$ is treated as the visible layer data for the 2nd level RBM. A sDBN is then formed by stacking together the RBMs and fine tuning the entire network using the up-down algorithm (Hinton et al., 2006). Alternatively, we can convert the sDBN into a deterministic feedforward network and minimize the cross-entropy error (Bengio et al., 2007). An example of such a network is shown in figure 3, where the the *rec* weights, $\mathbf{W}^{1,2,3,4}_{rec}$ form a feedforward classifier. Layer Z, $\mathbf{W}^2_{denoise}$ and $\mathbf{W}^1_{gen}$ are part of the denoising process described in section 4.

Specifically, the sDBN in our experiments has the same size and depth as the DBN in (Hinton et al., 2006), but its first layer is sparse with 7x7 RFs. It is fine tuned using the up-down algorithm for 300 epochs before discriminatively optimized for 30 more epochs[2]. Figure 4 shows the recognition errors on the noisy test set of the sDBN using only feedforward weights. Significant improvements can be seen for all types of noise.

[2]We use Conjugate gradient method to minimize the cross-entropy error with training data divided into batches of 5K each.

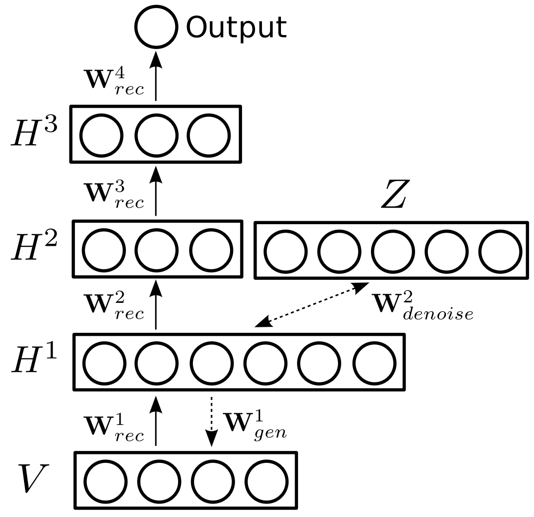


Figure 3. A deep network for feedforward recognition with denoising. Upward arrows are feedforward recognition weights, the downward dashed arrow is the generative weight, and the bidirectional dashed arrow is the weight of the denoising RBM. $\mathbf{W}^1_{gen}$ is part of the DBN and is used to calculate $p(\mathbf{v}|\mathbf{h}^1)$. If the network is not a DBN we can easily learn $\mathbf{W}^1_{gen}$ to predict the data $\mathbf{v}$ given $\mathbf{h}^1$.

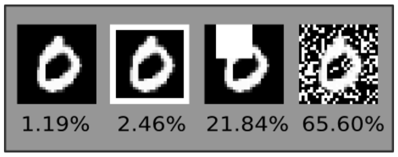


Figure 4. A sparse DBN is more robust to noise than the standard DBN, and only slightly worse on the clean images.

4. Probabilistic Denoising

When noise is present during recognition, the affected H^1 nodes increase the error rate. This is an out-of-sample problem, where an affected $q(\mathbf{h}^1|\tilde{\mathbf{v}})$ have low probability as defined by the training set. Classification boundaries in regions of state space with low probability cannot be trusted due to the lack of training data in those regions. Therefore, we seek to reduce the error rates by denoising $\mathbf{h}^1$ using a generative model of $q(\mathbf{h}^1|\mathbf{v})$[3].

We accomplish this by learning a separate *denoising* RBM that uses $q(\mathbf{h}^1|\mathbf{v})$ as its visible data[4]. $\mathbf{W}^2_{denoise}$ are the weights of this new RBM, which has Z (with 1000 nodes) as its hidden layer (figure 3). This RBM's energy function and the marginal of $\mathbf{h}^1$ are

$$E(\mathbf{h}^1, \mathbf{z}) = -\mathbf{d}^\intercal\mathbf{h}^1 - \mathbf{e}^\intercal\mathbf{z} - (\mathbf{h}^1)^\intercal\mathbf{W}^2_{denoise}\mathbf{z} \qquad (6)$$

$$p(\mathbf{h}^1) = \frac{p^*(\mathbf{h}^1)}{\sum_{\mathbf{h}^1,\mathbf{z}} \exp^{-E(\mathbf{h}^1,\mathbf{z})}} = \frac{\sum_{\mathbf{z}} \exp^{-E(\mathbf{h}^1,\mathbf{z})}}{\sum_{\mathbf{h}^1,\mathbf{z}} \exp^{-E(\mathbf{h}^1,\mathbf{z})}} \qquad (7)$$

Note that $\log p^*(\mathbf{h}^1)$ can be calculated analytically due to the bipartite nature of the RBM. We trained $\mathbf{W}^2_{denoise}$ for 600 epochs by using 100 persistent Markov chains to estimate the model's expectations (Tieleman, 2008). This method is known as Persistent CD and (compared to CD) can learn a better model for a fixed amount of computation.

The idea of denoising before classification can be understood schematically as depicted in figure 5, which shows a plot of noisy images above their unnormalized log probability $\log p^*(\mathbf{h}^1)$. Not surprisingly, highly noisy test images have much smaller $\log p^*(\mathbf{h}^1)$ and would be farther away from regions of high density. The dashed arrows indicate how we would like to *denoise* a noisy image by moving it (not necessarily one shot) to a region in state space of higher probability, putting it in a better region for classification.

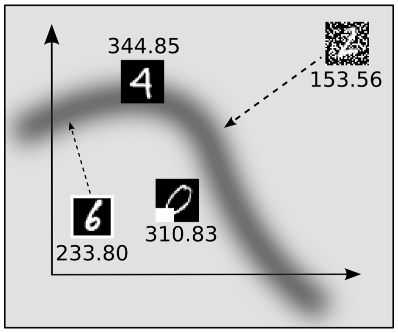


Figure 5. A hypothetical state space with the dark band being the region of high probability. See text for details.

4.1. Denoising via Unclamping

If we know which of the nodes in H^1 are affected, we can denoise by "filling in" their values by sampling from the distribution conditioned on all other H^1 nodes in the denoising RBM. For example, in figure 6, the two left most nodes of H^1 are unclamped while the rest are clamped.

Let us use $\psi_j \in \{0, 1\} = 1$ to denote the unclamping of node h^1_j and $\psi_j = 0$ the clamping of h^1_j. We run 50

[3]While denoising can also be done at the V layer, we prefer H^1 due to its more abstract representation of the input and smaller dimensionality.

[4]When the sDBN is fine tuned as a generative model by the up-down algorithm, we would ideally want to denoise using the $p(\mathbf{h}^1)$ defined by all the higher layers of the sDBN. However, we can only approximate the lower variational bound on $\log p^*(\mathbf{h}^1)$ by drawing samples from $q(\mathbf{h}^2|\mathbf{h}^1)$ (see (Salakhutdinov & Murray, 2008)). In contrast, a separate denoising RBM allows its model of $\log p^*(\mathbf{h}^1)$ to be calculated exactly.

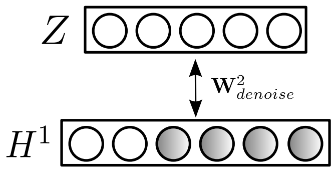


Figure 6. The shaded nodes are clamped. Denoising is performed by running block Gibbs sampling on the unclamped nodes.

iterations of block Gibbs sampling to sample H^1 nodes using

$$p(z_k|\mathbf{h}^1) = \sigma\big(\sum_j W^2_{jk}h^1_j + e_k\big) \tag{8a}$$

$$p(h_j|\mathbf{z}) = \sigma\big(\sum_k W^2_{jk}z_k + d_j\big) \tag{8b}$$

where we only use 8b to update the unclamped ($\psi_j = 1$) nodes. After denoising, we denote the H^1 activation as $\mathbf{g}$. Figure 7 shows denoised results $\hat{\mathbf{v}} = \sigma\big(\mathbf{W}^1_{gen}\mathbf{g}+\mathbf{b}\big)$ using the above method when the noisy hidden nodes or ψ_j are explicitly specified. It is clear that if the noisy nodes are correctly identified, correct classification will be much easier.

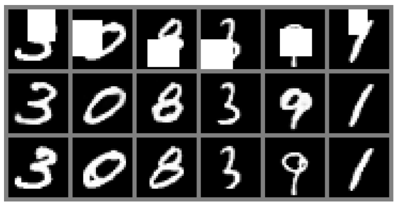


Figure 7. The first row are occluded images, the second row are the denoised results, and the third row are the original images.

4.2. Determining Which Nodes to Unclamp

During recognition, a DBN does not know which H^1 nodes to unclamp. We present here an iterative denoising algorithm which uses the gradient of $\log p(\mathbf{h}^1)$ of the RBM defined in eqs. 6 and 7 to determine which hidden layer nodes to unclamp. Denoting $\mathbf{h}^1_0 = q(\mathbf{h}^1|\tilde{\mathbf{v}})$ to be the initial H^1 activation at time step 0, we estimate $\boldsymbol{\psi}_0$ and compute $\mathbf{g}_0$. Setting $\mathbf{h}^1_1 = \mathbf{g}_0$, we repeat this process for several time steps.

The discrete gradient of the log probability with respect to ψ_j at time step t is given as:

$$\nabla_{\psi_j} \log p(\mathbf{h}^1_t) = \log p^*(\widetilde{\mathbf{h}}^1_{\backslash j,t}) - \log p^*(\mathbf{h}^1_t) \tag{9}$$

which is evaluated at $\boldsymbol{\psi} = \mathbf{0}$. We denote $\mathbf{h}^1_{\backslash j}$ to be the set of all nodes in H^1 except h^1_j. $\widetilde{\mathbf{h}}^1_{\backslash j,t}$ is $\mathbf{h}^1_t$ with the j-th node replaced by $p(h^1_j = 1|\mathbf{h}^1_{\backslash j})$, which is given by

$$p(h^1_j = 1|\mathbf{h}^1_{\backslash j}) = \frac{\exp(d_j)\prod_k^{N_z}(1+\exp(\phi_k + W^2_{jk}))}{\exp(d_j)\prod_k^{N_z}(1+\exp(\phi_k + W^2_{jk})) + \prod_k^{N_z}(1+\exp(\phi_k))} \tag{10}$$

and

$$\boldsymbol{\phi} = (\mathbf{W}^2_{\backslash j})^\top \mathbf{h}^1_{\backslash j} + \mathbf{e} \tag{11}$$

where $\mathbf{W}^2_{\backslash j}$ is $\mathbf{W}^2_{denoise}$ omitting the j-th row, $\mathbf{e}$ is the bias to layer Z, $\mathbf{d}$ is the bias to H^1, and N_z is the number of nodes in layer Z.

We can then estimate which of the H^1 nodes to unclamp by using a threshold

$$\psi_{j,t} = \begin{cases} 1 & \text{if } \nabla_{\psi_j} \log p(\mathbf{h}^1_t) > \eta(t) \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where $\eta(t)$ is a constant decreasing with time. $\psi_{j,t}$ is then used in the calculation of $\mathbf{g}_t$, as described in section 4.1. We update the hidden layer activations in the next time step to

$$\mathbf{h}^1_{t+1} \longleftarrow \mathbf{g}_t \tag{13}$$

The standard Bayesian approach to denoising is to specify a prior over $p(\mathbf{h})$ and then to find the MAP estimate of $p(\mathbf{h}|\widetilde{\mathbf{h}})$, where $\widetilde{\mathbf{h}}$ is the noisy H^1 activation. In contrast, we try to optimize $\log p(\mathbf{h})$ with respect to the parameters $\boldsymbol{\psi}$. In our algorithm, unclamping node $\widetilde{h}_j$ is similar to specifying the noise likelihood to be flat for node j: $p(\widetilde{h}_j|h_j) \propto$ *constant*; while clamping node h_j is similar to specifying the Dirac delta for the noise likelihood of node j: $p(\widetilde{h}_j|h_j) = \delta(\widetilde{h}_j - h_j)$.

4.3. Combining with Visible Layer Inputs

Having obtained a denoised $\mathbf{g}_t$, we can simply use $\mathbf{g}_t$ as our H^1 activation and compute $q(\mathbf{h}_2|\mathbf{g}_t)$, $q(\mathbf{h}_3|\mathbf{h}_2)$, etc. all the way up to the output for classification. However, it is much better if we also take into account the bottom-up inputs from V. This idea comes naturally for the Deep Boltzmann Machine (DBM) (Salakhutdinov & Hinton, 2009), where due to the fact that H^1 has undirected connections from both V and H^2, $p(\mathbf{h}^1|\mathbf{h}^2, \mathbf{v})$ involves both $\mathbf{v}$ and $\mathbf{h}^2$.

Since V layer nodes contain noise, we do not want to use the unreliable bottom-up influences directly. Instead, we would like to attenuate the noise part of V with an attention-like multiplicative feedback gating signal $\mathbf{u} = [0, 1]$. The attenuated bottom-up influence would be

$$q(\mathbf{h}^1|\mathbf{v};\mathbf{u}) = \sigma\big((\mathbf{W}^1_{rec})^\top(\mathbf{v}\odot\mathbf{u}) + \mathbf{c}\big) \tag{14}$$

where we denote $\odot$ to be element-wise multiplication. $\mathbf{v} \odot \mathbf{u}$ is the multiplicative interaction and partly inspired by visual neuroscience.

Recently, there are mounting neurophysiological evidence for considerable attentional modulation of early visual areas such as V1 (Posner & Gilbert, 1999; Buffalo et al., 2010). fMRI studies of human subjects performing recognition tasks with distractors have suggested that attentional modulation could be a delayed feedback to V1 from higher cortical areas (Martínez et al., 1999). Attention can also be stimulus dependent and has been shown to affect visual processing both spatially and feature-specifically (Treue & Martínez Trujillo, 1999). In one interesting study, (Lamme, 1995) showed that neurons in Macaque V1 responded better to texture in the foreground than to similar textures in the background 30-40 ms after onset of activation. The nonlocality nature and temporal latency of reponse differences strongly suggest feedback from higher visual areas.

By using $\mathbf{u}$ in our algorithm, we introduce a very simple method for dealing with noisy V layer nodes. To compute $\mathbf{u}$ we use[5]

$$\mathbf{u} = 1 - |\mathbf{v} - p(\mathbf{v}|\mathbf{g}; \mathbf{W}^1_{gen})| \tag{15}$$

where $\mathbf{W}^1_{gen}$ is the first layer's generative weights. To combine the modulated bottom-up input with the denoised activation $\mathbf{g}$, we compute a weighted average based on the amount of noise in the RF of a hidden node

$$\mathbf{g}_{combined} = q(\mathbf{h}^1|\mathbf{v};\mathbf{u}) \odot \frac{\mathbf{u}^\mathsf{T}\widetilde{\mathbf{W}}}{\gamma^2} + \mathbf{g} \odot (1 - \frac{\mathbf{u}^\mathsf{T}\widetilde{\mathbf{W}}}{\gamma^2}) \tag{16}$$

where γ is the size of the RF and $\widetilde{\mathbf{W}}$ is defined by eq. 5. To update our hidden layer activation in the next time step, we modify eq. 13 to be

$$\mathbf{h}^1_{t+1} \longleftarrow \mathbf{g}_{combined,t} \tag{17}$$

The entire training and inference process for the sDBN is summarized in Algorithm 1.

4.4. Denoising Results

In our experiments, we used 6 denoising iterations ($t = 1$ to $t = 6$) with a linearly decaying $\eta(t)$ from 1.0 to 0.0. Results were similar for other $\eta(t)$ and number of iterations. Figure 8 shows the intermediate denoising results. The combination of the top-down and bottom-up signals is vital to good results. Besides the aforementioned types of noise, we also experimented with pepper noise and occlusions by crossed lines.

[5] We can interpret u_i to be $p(v_i = noise|\mathbf{g})$.

Algorithm 1 Sparse DBN Training and Inference

```
  Learning:
1: Learn sRBM using eq. 4.
2: Greedy pretraining of higher layer RBMs and stack
   to form a sDBN.
3: Fine tune using the up-down algorithm.
4: Convert the sDBN into a discriminative classifier
   and minimize cross-entropy error.
5: Learn $\mathbf{W}^2_{denoise}$ using $q(\mathbf{h}^1|\mathbf{v})$ as input.
6: Learn $\mathbf{W}^1_{gen}$ by minimizing cross-entropy between
   the data and $p(\mathbf{v}|q(\mathbf{h}^1|\mathbf{v}))$.
  Recognition:
1: For noisy input $\tilde{\mathbf{v}}$, compute $\mathbf{h}^1_0 = q(\mathbf{h}^1|\tilde{\mathbf{v}})$.
  for $t = 1$ to n do
2:     Estimate $\boldsymbol{\psi}_t$ using eq. 12
3:     Gibbs sampling to obtain $\mathbf{g}_t$ using eq. 8
4:     Combine with bottom up input to obtain
       $\mathbf{g}_{combined,t}$ using Eq. 16
5:     $\mathbf{h}^1_{t+1} \longleftarrow \mathbf{g}_{combined,t}$
  end for
6: Compute $q(\mathbf{h}^2|\mathbf{h}^1_{n+1})$, then feedforward to output.
```

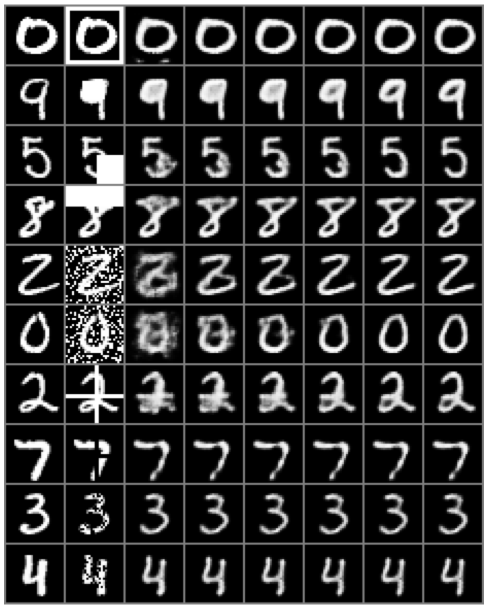
4.5. Recognition Results

For recognition, we performed 10 iterations of denoising with $\eta(t)$ decaying from 2.0 to 0.2 for each test image. After denoising, we proceeded with the feedforward recognition by computing $q(\mathbf{h}^2|\mathbf{g}_{10})$ and feedfoward to the output using the *rec* weights. In table 2, we summarize the error rates on MNIST for all the networks. The 7x7+denoised line has the error rates found after denoising. Denoising provides a large improvement over the accuracy of the sDBN for noisy images. However, the denoising sDBN is slightly worse than standard DBN on the clean images. This effect is hard to avoid since denoising seeks to increase probability of $\mathbf{h}^1$ defined over all 10 digits and may cross classification boundaries.

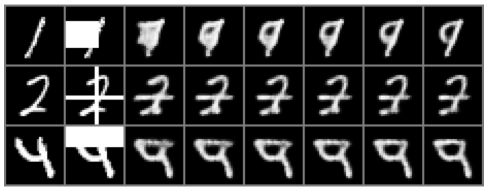
For comparison, we also trained a standard DBN and a 7x7 sDBN with noise added evenly to the 60K MNIST training set. They are fine tuned with 300 epochs of up-down algorithm followed by 30 epochs of discriminative optimization. The results show that sparse connections are better for recognition in this case as well. It is also revealing that in comparison to the denoising sDBN (trained only on clean images), the error rates is only lower on the block occluded test images.

5. Discussion

It should be noted that the specific approach taken here does not depend on our adoption of the DBN.



(a) Successful examples



(b) Failed examples

Figure 8. Denoised results on various types of noise. The first column from the left contains the original images, the second column contains images with noise added. Subsquent columns represent the denoised images from $t = 1$ to $t = 6$.

That is, if the network is not a DBN fine tuned by the up-down algorithm, $\mathbf{W}^1_{gen}$ can be learned by maximum likelihood estimation. Consequently, this denoising algorithm can be easily adapted to *any* deep feedforward classifier as long as the first layer has spatially localized receptive fields.

There are several avenues for extending the present model. For one, human visual recognition of partially visible objects is more accurate if the occluding object can be identified (Fukushima, 2001; Johnson & Olshausen, 2005). In our experiments, when the block occluded region is known, denoising is much better. Compare the results of the block occlusion from figure 7 with those of failed examples from figure 8. Accordingly, recognition error is reduced from 19% to 10% for the block occlusion noise test set. We hypothesize that the identification of the occluder is similar to specifying $\boldsymbol{\psi}$. Therefore, an important avenue for future work is in the improvement of estimating the occluding object or $\boldsymbol{\psi}$.

Table 2. Summary of recognition results

Network	clean	border	block	random
28x28 DBN	1.03%	66.14%	33.78%	79.83%
7x7 sDBN	1.19%	2.46%	21.84%	65.50%
7x7+denoised	**1.24%**	**1.29%**	**19.09%**	**3.83%**
28x28+noise	1.68%	1.95%	8.72%	8.01%
7x7+noise	1.61%	1.77%	8.39%	6.64%

Currently, denoising takes place on the hidden layer. It is also possible to denoise in the visible layer. Even though preliminary results of applying our denoising algorithm on the visible layer alone suggest that it is quite difficult, the combination of denoising on both the hidden and visible layers may give better results. Denoising at higher layers is also possible. However, due to the fact that the RFs of the first hidden layer are chosen randomly, H^1 is not topographically ordered. It is certainly possible to organize H^1 to be topographical and enforce sparse connections to H^2, thereby making denoising $\mathbf{h}^2$ effective.

6. Conclusions

In this paper, we have demonstrated that combining sparsification with explicit denoising results in a DBN that is much more robust to noise not in the training set than a standard DBN. We introduced an algorithm which is capable of denoising a test image by combining top-down influences with bottom-up inputs. Our denoising process does not model the noise process at all, but instead uses the log probability to estimate which nodes should be unclamped. It is able to handle a variety of noise and is inspired by findings in neurophysiology. Finally, the denoising itself can be adapted to a broad class of deep feedforward networks, making such an approach likely to be useful for other architectures not explored here.

Acknowledgements

We thank the anonymous reviewers for making this a much better manuscript. This research was supported by NSERC.

References

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy layer-wise training of deep networks. In *Adv. in Neural Information Processing Systems 19*, pp. 153–160, 2007.

Buffalo, E. A., Fries, P., Landman, R., Liang, H., and Desimone, R. A backward progression of attentional effects in the ventral stream. *Proceedings of the National Academy of Sciences*, 107(1):361–365, Jan. 2010.

Fukushima, K. Neocognitron: A neural model for a mechanism of visual pattern recognition. *IEEE Trans. SMC*, 13(5):826–834, 1983.

Fukushima, K. Recognition of partly occluded patterns: A neural network model. *Biological Cybernetics*, 84(4):251–259, 2001.

Haenny, P. E. and Schiller, P. H. State dependent activity in monkey visual cortex. I. single cell activity in V1 and V4 on visual tasks. *Experimental Brain Research*, 69 (2):225–244, 1988.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

Hinton, G. E. and Salakhutdinov, R. Reducing the dimensionality of data with neural networks. *Science*, 313:504–507, 2006.

Hinton, G. E., Osindero, S., and Teh, Y. W. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

Hubel, D. and Wiesel, T. Receptive fields of single neurons in the cats striate cortex. *Journal of Physiology*, 148:574–591, 1959.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. What is the best multi-stage architecture for object recognition? In *Proc. Intl. Conf. on Computer Vision (ICCV'09)*. IEEE, 2009.

Johnson, J. S. and Olshausen, B. A. The recognition of partially visible natural objects in the presence and absence of their occluders. *Vision Research*, 45 (25-26):3262–3276, Nov. 2005.

Lamme, V. A. The neurophysiology of figure-ground segregation in primary visual cortex. *The Journal of neuroscience: the official journal of the Society for Neuroscience*, 15:1605–1615, 1995.

LeCun, Y., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov. 1998.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Intl. Conf. on Machine Learning*, pp. 609–616, 2009.

Lee, T. S. and Mumford, D. Hierarchical bayesian inference in the visual cortex. *Journal of the Optical Society of America*, 20:1434–1448, 2003.

Martínez, A., Anllo-Vento, L., Sereno, M. I., Frank, L. R., Buxton, R. B., Dubowitz, D. J., Wong, E. C., Hinrichs, H., Heinze, H. J., and Hillyard, S. A. Involvement of striate and extrastriate visual cortical areas in spatial attention. *Natural Neuroscience*, 2 (4):364–369, Apr. 1999.

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

Norouzi, M., Ranjbar, M., and Mori, G. Stacks of convolutional restricted boltzmann machines for shift-invariant feature learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2735–2742, 2009.

Posner, M. I. and Gilbert, C. D. Attention and primary visual cortex. *Proc. of the National Academy of Sciences*, 96(6), March 1999.

Roth, S. and Black, M. J. Fields of experts: A framework for learning image priors. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 860–867, 2005.

Salakhutdinov, R. and Hinton, G. Deep Boltzmann machines. In *Proceedings of the Intl. Conf. on Artificial Intelligence and Statistics*, volume 5, pp. 448–455, 2009.

Salakhutdinov, R. and Murray, I. On the quantitative analysis of deep belief networks. In *Proceedings of the Intl. Conf. on Machine Learning*, volume 25, 2008.

Serre, T., Wolf, L., and Poggio, T. Object recognition with features inspired by visual cortex. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 994–1000, 2005.

Tieleman, T. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Intl. Conf. on Machine Learning*, volume 307, pp. 1064–1071, 2008.

Treue, S. and Martínez Trujillo, J. C. Feature-based attention influences motion processing gain in macaque visual cortex. *Nature*, 399(6736):575–579, Jun. 1999.

Welling, M., Hinton, G. E., and Osindero, S. Learning sparse topographic representations with products of student-t distributions. In *Adv. in Neural Information Processing Systems*, pp. 1359–1366, 2002.