# Multiresolution Deep Belief Networks

**Yichuan Tang**
Department of Computer Science
University of Toronto

**Abdel-rahman Mohamed**
Department of Computer Science
University of Toronto

## Abstract

Motivated by the observation that coarse and fine resolutions of an image reveal different structures in the underlying visual phenomenon, we present a model based on the Deep Belief Network (DBN) which learns features from the multiscale representation of images. A Laplacian Pyramid is first constructed for each image. DBNs are then trained separately at each level of the pyramid. Finally, a top level RBM combines these DBNs into a single network we call the Multiresolution Deep Belief Network (MrDBN). Experiments show that MrDBNs generalize better than standard DBNs on NORB classification and TIMIT phone recognition. In the domain of generative learning, we demonstrate the superiority of MrDBNs at modeling face images.

## 1 Introduction

Unsupervised learning seeks to discover latent representation which captures interesting and useful structures in high dimensional data. Deep learning architectures successively train unsupervised learners on the latent activations of previous learners, thereby sequentially transforming the raw input (e.g. pixels) into more interesting and useful features by capturing higher-order correlations of the input. Deep architectures such as the Deep Belief Network [1] (DBN), Deep Boltzmann Machine [2] (DBM) and Stacked Autoencoders [3] have been shown to be excellent at visual object recognition [4], speech phone recognition [5, 6], and image denoising [7].

In the domain of vision, it is common practice to use image pixels as the input to the bottom layer of deep models. While an image is a representation of an object at a specific resolution and scale, objects in the real world possess different structures at different scales [8]. For example, a forest to a far away observer is seen as individual trees to someone close by. Therefore, we hypothesize that it would be beneficial to explicitly provide multiple resolutions of an image as input to a deep model. This hypothesis is indeed confirmed by both generative and discriminative experiments in this paper.

Multiscale and multiresolution processing and analysis are applied widely in the fields of image processing and computer vision: object detection [9, 10], feature points detection [11], alignment and tracking [12], and image blending [13]. The usefulness of a multiresolution framework stems from its ability to select and perform computations at the optimal image scale.

In this paper, we combine multiresolution representation and unsupervised generative learning. We propose a modified Deep Belief Network which takes multiresolution images as its visible input. The Multiresolution DBN (MrDBN) has several advantages:

- Learning features from multiple resolutions and frequencies helps the resulting classifier generalize better.
- Learning coarse structure from low resolution images serves as a form of regularization that allows the learning of a better generative model.
- Utilizing high resolution information as needed during visual search can dramatically reduce computation costs.

In section 2 we review the Gaussian-Binary Restricted Boltzmann Machine, which is used as the first layer of DBNs. Learning of the visible nodes' residual variances and the preprocessing of the input are also discussed. Section 3 presents the details of MrDBN. Experimental results are in section 4, followed by conclusions.

Appearing in Proceedings of the $15^{th}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX. 

## 2 Gaussian-Binary Restricted Boltzmann Machine

Before describing the MrDBN, we briefly review the Gaussian-Binary Restricted Boltzmann Machine (GRBM). While the original Binary-Binary RBM is the standard building block of DBNs and DBMs, the GRBM extends the Binary-Binary RBM to handle continuous data [14].

GRBM is a bipartite Markov Random Field over the $N_v$ visible nodes $V \in \mathbb{R}^{N_v}$ and $N_h$ hidden nodes $H \in \{0,1\}^{N_h}$, defined by an energy function:

$$E(\mathbf{v},\mathbf{h};\theta) = \frac{1}{2}\mathbf{v}^\mathsf{T}\Lambda\mathbf{v} - \mathbf{v}^\mathsf{T}\Lambda\mathbf{b} - \mathbf{h}^\mathsf{T}\mathbf{c} - \mathbf{v}^\mathsf{T}\Lambda^{\frac{1}{2}}\mathbf{W}\mathbf{h} \quad (1)$$

$\theta = \{\mathbf{W},\mathbf{b},\mathbf{c},\Lambda\}$ are the model parameters. $\mathbf{W} \in \mathbb{R}^{N_v \times N_h}$, $\mathbf{b} \in \mathbb{R}^{N_v}$, $\mathbf{c} \in \mathbb{R}^{N_h}$, and $\Lambda$ is the precision matrix of $\mathbf{v}$, taken to be diagonal. We use $W_{ij}$ to refer to the element at the $i$th row and $j$th column and $\mathbf{W}_{(:,j)}$ to refer to the $j$th column of $\mathbf{W}$. By exponentiating and normalizing, we obtain a probability density function over the states of $V$ and $H$:

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

where $Z(\theta)$ is the normalization constant: $Z(\theta) = \int_{\mathbf{v}'} d\mathbf{v}' \sum_{\mathbf{h}'} \exp^{-E(\mathbf{v}',\mathbf{h}';\theta)}$ and $p^*(\cdot)$ denotes the unnormalized probability density. Conditioned on $H = \mathbf{h}$, the distribution over the visible nodes is:

$$p(\mathbf{v}|\mathbf{h}) \sim \mathcal{N}(\Lambda^{-\frac{1}{2}}\mathbf{W}\mathbf{h} + \mathbf{b}, \Lambda^{-1}) \quad (3)$$

The conditional distribution over the hidden nodes given the visibles is:

$$p(\mathbf{h}|\mathbf{v}) = sigmoid\ (\mathbf{v}^\mathsf{T}\Lambda^{\frac{1}{2}}\mathbf{W} + \mathbf{c}) \quad (4)$$

By analytically integrating out the binary hidden variables, we obtain the unnormalized log-probability of a visible vector:

$$\log p^*(\mathbf{v}) = -\frac{1}{2}\mathbf{v}^\mathsf{T}\Lambda\mathbf{v} + \mathbf{v}^\mathsf{T}\Lambda\mathbf{b} + \sum_j^{N_h} \log\left(1 + \exp\{\mathbf{v}^\mathsf{T}\Lambda^{\frac{1}{2}}\mathbf{W}_{(:,j)} + c_j\}\right) \quad (5)$$

Learning is accomplished by performing gradient ascent on the log-likelihood of the parameters given the i.i.d. training data. Since $Z(\theta)$ can not be computed exactly in less than exponential time, algorithms such as Contrastive Divergence [15] (CD) and Fast Persistent Contrastive Divergence [16] (FPCD) can be used for approximate maximum likelihood estimation (MLE).

GRBMs have been used to model images [17], speech [5], and human motions [18]. A deep architecture can be formed by stacking multiple layers of Binary-Binary RBMs on top of the GRBM, forming a DBN or a DBM.

### 2.1 Learning the residual variances

Most of the work in literature that use GRBMs treat $\Lambda_{ii}$ as constants. $\Lambda$ is usually set to be the identity matrix, while the data can be preprocessed by an arbitrary scaling [19]. This approach is not optimal and leads to worse density models. The residual variances represent the variance unexplained by the model, and should be much smaller than the data variances. For example, pixels of the cheek and forehead regions of faces have much lower residual variances than pixels of the eyes and mouth regions.

In this paper, we learn the precision parameters $\lambda_i \triangleq \Lambda_{ii}$ by gradient ascent on the log-likelihood objective. We can take the partial derivative of the unnormalized log-probability w.r.t. $\lambda_i$ to give:

$$\frac{\partial \log p^*(\mathbf{v})}{\partial \lambda_i} = -\lambda_i v_i^2 + 2\lambda_i v_i b_i + \sum_j^{N_h} p(h_j|\mathbf{v}) v_i W_{ij} \quad (6)$$

we update $\lambda_i$ using the difference of two expectations[1]:

$$\lambda_i \longleftarrow \lambda_i + \alpha\big(\mathbb{E}_{data}[\frac{\partial \log p^*(\mathbf{v})}{\partial \lambda_i}] - \mathbb{E}_{model}[\frac{\partial \log p^*(\mathbf{v})}{\partial \lambda_i}]\big) \quad (7)$$

We constrain $\lambda_i$ to be: $0 \le \lambda_i \le \kappa$. $\lambda_i$ is initialized with 1.0 and $\kappa$ is usually set to 1000. In our experiments, $\lambda_i$ never comes close to $\kappa$.

### 2.2 Contrast Normalization

Before learning the parameters, we first preprocess all data vectors $\mathbf{v}$ by subtracting the vectors' mean and setting the resulting vectors to have a norm of $C$:

$$\mathbf{v}' = C\frac{\mathbf{v} - \frac{1}{N_v}\sum_i v_i}{\parallel \mathbf{v} - \frac{1}{N_v}\sum_i v_i \parallel_2} \quad (8)$$

This step is crucial to learning a reasonable $\Lambda$. This is due to the energy function of the GRBM. Unlike directed models with separate parameters to explicitly specify the prior $p(\mathbf{h})$, the GRBM uses its weights $\mathbf{W}$ to define the prior $p(\mathbf{h})$ implicitly:

$$p(\mathbf{h}) \propto \exp\{\mathbf{c}^\mathsf{T}\mathbf{h} + \frac{1}{2}(\Lambda^{-\frac{1}{2}}\mathbf{W}\mathbf{h} + \mathbf{b})^\mathsf{T}\Lambda(\Lambda^{-\frac{1}{2}}\mathbf{W}\mathbf{h} + \mathbf{b})\} \quad (9)$$

[1] $\mathbb{E}_{data}[\cdot]$ is the expectation over the training distribution, $\mathbb{E}_{model}[\cdot]$ is the expectation over the model distribution.

where $\Lambda^{-\frac{1}{2}}\mathbf{W}\mathbf{h} + \mathbf{b}$ is the conditional mean of $p(\mathbf{v}|\mathbf{h})$ (see Eq. 3). Since the probability assigned to a visible vector $\mathbf{v}$ is $p(\mathbf{v}) = \sum_{\mathbf{h}'} p(\mathbf{v}|\mathbf{h}')p(\mathbf{h}')$, a GRBM is a mixture of exponentially many diagonal Gaussians, where each component of the mixture is specified with 1 of the $2^{N_h}$ possible hidden configurations. Let's refer to components of the mixture with conditional means which have relatively big norms the *big-norm components* and ones with small norms the *small-norm components*. Due to Eq. 9, *big-norm components* tend to have a much bigger mixing proportion compared to the *small-norm components*[2]. As the result of this property, when learning from data of different norms, a *big-norm component* will tend to take "ownership"[3] not only of data close to its conditional mean, but also of data close to the conditional mean of *small-norm components*. This leads to the inability to learn a proper $\Lambda$ since a big-norm component can not predict its "members" very precisely.

By using the normalization method of Eq. 8, we remove the inductive bias of the GRBM to prefer its big-norm components. In order to reconstruct the original images, we save the mean ($\frac{1}{N_v}\sum_i v_i$) and norm ($\| \mathbf{v} - \frac{1}{N_v}\sum_i v_i \|_2$) of each image $\mathbf{v}$ by concatenating those two scalars to $\mathbf{v}'$.

## 3 Multiresolution Deep Belief Network

After learning the first layer GRBM, a DBN is formed by learning a second RBM with the GRBM's hidden activations as input. This greedy layer-wise stacking can be repeated as many times as desired, forming a DBN with as many layers. See [1, 3] for a detailed account of the formulation process.

Recently, extensions to the simple GRBM has been developed which capture the covariance structure of the visible input. They include the mean and covariance RBM [20], Relu RBM [21], Spike and Slab RBM [22], and have been shown to be superior first layer models. In this work, we will use the simple GRBM as our first layer model, noting that MrDBN could use any of the above mentioned extensions.

MrDBN differs from the standard DBN by taking multiple resolutions of an input image as its visible inputs. Given a training set of images, a sample image $\mathbf{v}^0 \in \mathbb{R}^{M\times M}$ has image size of $M \times M$, where we assume $M = M_0 \times 2^K$ for some nonnegative integer $\{K : K \geq 2\}$ . Starting from the original full resolution image $\mathbf{v}^0$, we construct images $\mathbf{v}^1$ with size $\frac{M}{2}\times\frac{M}{2}$ and $\mathbf{v}^2$ with size $\frac{M}{4}\times\frac{M}{4}$. These lower resolution images form the upper levels of a Gaussian Pyramid and are generated by downsampling using bicubic interpolation after blurring $\mathbf{v}^0$ with a Gaussian kernel. We then upsample $\mathbf{v}^1$ and $\mathbf{v}^2$ by a factor of 2 using bicubic interpolation. The pixel intensity differences are computed:

$$\mathbf{f}^0 = \mathbf{v}^0 - \mathit{UPSAMPLE}(\mathbf{v}^1) \tag{10}$$

$$\mathbf{g}^0 = \mathbf{v}^1 - \mathit{UPSAMPLE}(\mathbf{v}^2) \tag{11}$$

$\mathbf{f}^0$ and $\mathbf{g}^0$ are the differences between adjacent layers of the Gaussian Pyramid which approximate the output response of filtering $\mathbf{v}^0$ with a Laplacian of Gaussian (LoG) filter. Also known as "Mexican hat" filters, LoG filters are widely used in models of biological vision and interest point detections. The process of Eq. 10 and 11 constructs a Laplacian Pyramid [23], which band-pass filters the original signal. $\mathbf{f}^0$, $\mathbf{g}^0$, and $\mathbf{v}^2$ are components of $\mathbf{v}^0$ with high, medium, and low frequency, respectively. These vectors form the input to MrDBN.

In Figure 1, a MrDBN that models $64 \times 64$ images is shown[4]. There are three separate "streams", one for each frequency component. Specifically, GRBMs are learned with $F^0$, $G^0$ and $V^2$ as their visible layers. RBMs are then learned in a greedy layer-wise fashion with the first hidden layer activations ($\mathbf{f}^1$, $\mathbf{g}^1$ and $\mathbf{h}^1$) as inputs. At the top layer, label information in the form of 1-of-K codes are included (if available) along with the combined topmost hidden activations of all three streams. While CD learning is used for the lower layers of MrDBN, the top layer uses FPCD to learn a better density model. We summarize the inference and generation steps of MrDBN in Algorithm 1.

## 4 Experiments

We use two image databases and a speech corpus to evaluate MrDBNs. The first database is the Toronto Face Database [24] (TFD). The TFD is a large set of face images collected from many existing databases. It contains over 100K size $100 \times 100$ gray-scale face images. It is created by merging 30 pre-existing databases and is one of the largest collection of face images. The database also contains identity and expression labels for a small number of the faces.

[2] Large values of $c_j$ only allow *small-norm components* to have a larger prior when the number of data modes is not much more than the number of hidden nodes (e.g. each hidden node represents a mode). Moreover, having large values of $c_j$ means that MCMC mixing (and therefore learning) would be virtually impossible.

[3] $\hat{\mathbf{h}}$ owns $\mathbf{v}$ if it is the hidden configuration with the largest posterior probability $p(\mathbf{h}|\mathbf{v})$.

[4] We use capital letters to denote a specific layer of nodes and use lowercase bold letters to denote a specific set of activations of the corresponding layer.

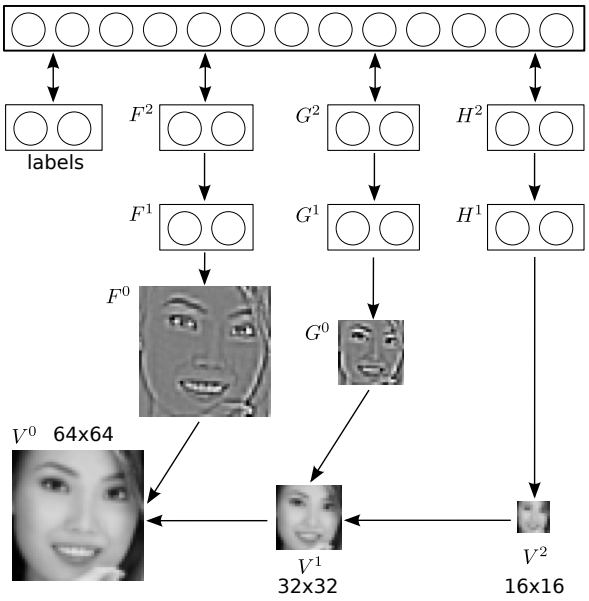



Figure 1: The architecture of MrDBN. Arrows pointing downwards represent directed connections, whereas dual arrows represent undirected ones. See text for details.

The second one is NORB [25]. NORB is a 3D dataset of toy objects. There are five object categories: *animals*, *human*, *airplane*, *cars*, and *trucks*. 10 different objects per category are captured with a stereo camera at different azimuths, elevations and lighting conditions. We will use the smaller uniform-normalized version. This dataset contains 24,300 pairs of stereo images for both training and test. For any object category, 5 of the 10 toy objects of that category are designated as part of the training set and the other 5 as part of the test set. This makes recognition challenging by having a test data distribution that is different from the training data distribution. We will first demonstrate the advantages of MrDBNs as generative models on both TFD and NORB. We then evaluate MrDBNs for classification on NORB.

The third is the TIMIT acoustic-phonetic continuous speech corpus. It contains over 500 American English speakers with lexically transcribed speech. We will show that MrDBN can also be applied to speech signals to improve speech phone recognition.

### 4.1 Generative Experiments

One of the advantages of generative models is the ability to better interpret noisy or ambiguous inputs. To compare MrDBNs to DBNs as generative models qualitatively, we draw samples from each model and visually compare them. Quantitatively, since MrDBN augments the original image with the upper layers of the Laplacian Pyramid, it is not straight forward to perform model comparison using the test log-probabilities. For an indirect measure, we compute the test reconstruction error after propagating the activities from the visible input all the way to the top and then back down again, using Algorithm 1. Reconstruction error is a crude approximation of Contrastive Divergence and is typically monotonically related to the average data log-probability, provided that reasonable looking samples are obtained when sampling from the model.

**Algorithm 1** Inference and Generation for MrDBN

**Inference**:
Given an input image $\mathbf{v}^0$ of dimensionality $M \times M$, where $M = M_0 \times 2^K$, $K$ an integer $\geq 2$.
1: Low-pass filter $\mathbf{v}^0$ with a Gaussian kernel and downsample by a factor of 2 and 4 using bicubic interpolation, producing $\mathbf{v}^1$ and $\mathbf{v}^2$, respectively.
2: Upsample $\mathbf{v}^1$ and $\mathbf{v}^2$ by a factor of 2 using bicubic interpolation.
3: Subtract the upsampled images from the original to give the difference images: $\mathbf{f}^0$ and $\mathbf{g}^0$.
4: As in a standard DBN, use the RBM weights to compute approximate posterior distribution of a higher layer given the layer below. E.g. $p(\mathbf{f}^1|\mathbf{f}^0) = sigmoid\ (\mathbf{f}^{0\mathsf{T}}\Lambda^{\frac{1}{2}}\mathbf{W} + \mathbf{c})$.
**Generation**:
5: Run n iterations of block Gibbs sampling on the top level RBM.
6: Project down directed connections by computing the conditional distribution $p(\mathbf{x}^l|\mathbf{x}^{l+1})$, where $\mathbf{x}$ represents any of the three streams' hidden layer nodes.
7: Sample $\mathbf{x}^l \sim p(\mathbf{x}^l|\mathbf{x}^{l+1})$
8: Repeat step 6-7 until after the sampling of the layer of $F^0$, $G^0$, and $V^2$.
9: Reconstruct $V^0$:
$V^0 = UPSAMPLE(UPSAMPLE(\mathbf{v}^2) + \mathbf{g}^0) + \mathbf{f}^0$

#### 4.1.1 Faces

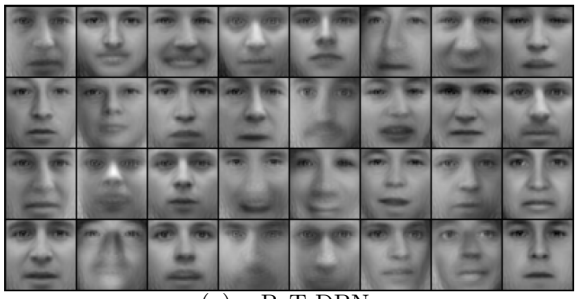
We randomly selected 60K images from the TFD as training data and 10K as test data. The $V^0$ layer of the MrDBN is created by downsampling[5] to $48 \times 48$ from the original resolution of $100 \times 100$. Lower resolutions of $24 \times 24$ and $12 \times 12$ are used in the MrDBN. Contrast normalization is done as described in section 2.2, setting $C$ to 10. The MrDBN has size 1500 for $F^1$, 1000 for $G^1$, 1000 for $H^1$; 500 for $F^2$, 500 for $G^2$ and 1000 for $H^2$; the top layer has 250 hidden nodes. For comparison, we trained a standard DBN with 3 hidden layers and approximately the same number of parameters as the MrDBN. For both models, all of the layers except

[5]Gaussian blurring and bicubic interpolation are used.

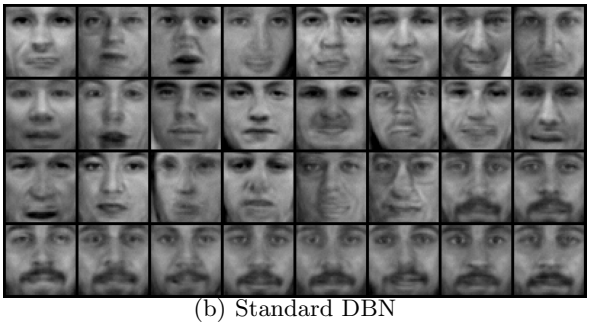
the top RBM are trained using CD. The top RBM is trained using FPCD with 100 fantasy particles and they are run for 50 Gibbs iterations before computing the negative phase statistics for one weight update. Sparsity is induced using the method described in [26] for the lower layers. 100 epochs are used to train the lower layers while 200 epochs are used for the top layer. No further fine-tuning is performed. The top layer are trained using a $L_2$ weight cost of 0.002. The weight decay for fast weights is set to 0.05. Learning rate is initially set to be 0.05, linearly decaying to 0.001. The fast learning rate is set to be the same as the initial learning rate.

To give a sense of the types of filters learned from each resolution, we plotted some randomly chosen filters in the left column of Figure 2. The right column is the learned residual variances $\Lambda$ for the corresponding resolution streams.

In Figures 3 and 4, we plot samples generated from the standard DBN, MrDBN and a more complicated DBN with mPoT first layer [7]. The Gibbs chain is started at $\mathbf{0}$ initially and samples are drawn every 100 steps. No burn-in samples were discarded.



(a) mPoT DBN



(b) Standard DBN

Figure 3: Samples drawn from various DBNs trained on the Toronto Faces Database. (a) Random samples of a DBN with mPoT as the first layer, reproduced from [7]. (b) Samples from a standard DBN with learned $\Lambda$. Starting at the top row, from left to right, each sample is generated after 100 steps of block Gibbs sampling.

Figure 4: Samples drawn from MrDBN. Starting at the top row, from left to right, each sample is generated after 100 steps of block Gibbs sampling. Note the level of details and the ability of the MCMC chain to mix rapidly.

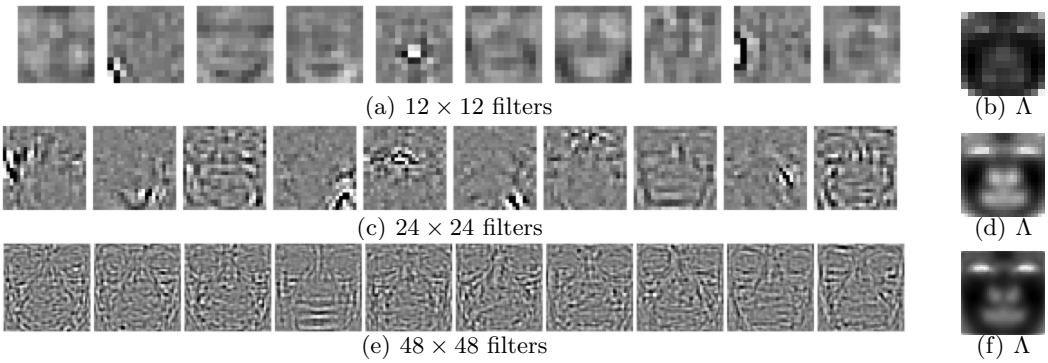
From visual examination, MrDBN generates sharper and more realistic faces than the standard DBN. In addition, it also mixes very fast, generating multiple types of expressions. It is worth noting the facial details MrDBN captures. For example, mustaches, teeth, and eyeglasses are clearly visible. Compared to the samples from the mPoT-DBN, samples from MrDBN allow us to distinguish the gender, age, and even the emotion of the synthesized faces.

We also looked at the test set reconstruction. The baseline DBN gave a mean squared error (MSE) of 17.9 per image while MrDBN gave 15.9 MSE per image[6].

### 4.1.2 NORB

Since border regions of the $96 \times 96$ NORB images are largely homogeneous, we cropped out the borders, leaving the middle $64 \times 64$ block. Contrast normalization sets $C$ to be 10. For MrDBN, we use 4000, 2000, and 3000 hidden nodes for layers $F^1$, $G^1$, and $H^1$, respectively. 2000, 1000 and 1500 hidden nodes are used for $F^2$, $G^2$, and $H^2$, respectively. 500 top layer hidden nodes are used. For comparison, we trained a standard DBN with 3 hidden layers and approximately the same number of parameters as the MrDBN. Training is same as in section 4.1.1 except we use a $L_2$ weight cost of

[6]For $48 \times 48$ images with intensity ranging from 0.0 to 1.0, MrDBN reconstructions are better by about 0.03 per pixel.



(a) $12 \times 12$ filters (b) $\Lambda$

(c) $24 \times 24$ filters (d) $\Lambda$

(e) $48 \times 48$ filters (f) $\Lambda$

Figure 2: Filters and residual variances of all three resolutions of MrDBN trained on faces.

0.001 for the top layer.

Figures 5 shows 100 samples drawn from each model. The states of the top layer for both DBNs are set to $\mathbf{0}$ initially. Following 1000 steps of block Gibbs sampling at the top, we project the activations down to the visible layers to generate a sample. These samples are shown starting at the top left, moving from left to right. As can be seen from Figure 5, the Gibbs chain from MrDBN mixes a bit faster and the samples are slightly cleaner than those generated from a standard DBN. Quantitatively, the MSE of the test reconstruction is 60.2 per image (DBN), and 52.0 per image (MrDBN).

Lower reconstruction errors are not conclusive evidence that MrDBN is better than DBN due to the possibility of spurious modes in some unknown parts of space. However, combined with samples drawn from both models in Figures 4 and 5(b), It is reasonable to believe that MrDBN is a better generative model.

### 4.2 Recognition Experiments

#### 4.2.1 NORB

We evaluate MrDBNs on NORB classification. To reduce computation time, we first downsampled normalized-uniform NORB images into size $32 \times 32$ stereo pairs from the original size of $96 \times 96$ using bicubic interpolation. We tried two different MrDBNs. The first one, MrDBN [32-16-8] uses three resolutions: $\{\ 32\times32,\ 16\times16,$ and $8\times8\ \}$. The second one, MrDBN [32-16], uses two resolutions: $\{\ 32 \times 32$ and $16 \times 16\ \}$.

The network size for MrDBN [32-16-8] is 1200 and 1200 for $F^1$ and $F^2$; 1000 and 1000 for $G^1$ and $G^2$; 700 and 700 for $H^1$ and $H^2$. The top layer has 4500 hidden nodes. The network size for MrDBN [32-16] is 1200 and 1200 for $G^1$ and $G^2$; 1500 and 2000 for $H^1$ and $H^2$. The top layer has 4500 hidden nodes. Note that the $F$ stream is not present in MrDBN [32-16], since it is the $64 \times 64$ resolution stream.

2300 random samples from the training setaistats 2012 are taken aside as validation while the other 22,000 samples are divided into 220 mini-batches of 100 samples each[7]. Contrast normalization is done and the norm of each vector is set to 10. We train the first 3 layers using CD while the top layer uses FPCD. The labels are one-of-K codes and are a part of the visibles of the top RBM. To classify, we first use step 4 of Algorithm 1 to approximate the posterior $p(\mathbf{f}^2, \mathbf{g}^2, \mathbf{h}^2|\mathbf{v})$. We then pick the label $\hat{\boldsymbol{\ell}}$ that gives the highest log-probability of the top RBM:

$$\hat{\boldsymbol{\ell}} = \arg\max_{\boldsymbol{\ell}} \quad \log p^*(\boldsymbol{\ell}, \mathbf{f}^2, \mathbf{g}^2, \mathbf{h}^2) \tag{12}$$

Since there are only 5 training objects for each category, overfitting is a problem. We set the top layer weight costs to be relatively high at 0.01. Learning rate is decayed from 0.05 to 0.001. The weights are initialized from a Normal distribution with standard deviation of 0.01. 50 training epochs are run and we use early stopping by looking at the validation error. We did not use additional discriminative fine-tuning on MrDBN as it leads to overfitting. Table 1 presents the test errors of different models. The test error of

| | |
|---|---|
| Deep Belief Net [4] | 8.3% |
| DBN (learning $\Lambda$) (this work) | 7.4% |
| Deep Boltzmann Machine [2] | 7.2% |
| MrDBN 32-16 (this work) | 5.8% |
| 3rd Order DBN [26] | 5.2% |
| Tiled Convolutional Nets [27] | 3.9% |

Table 1: NORB classification error rates.

[7] We randomly translated the training data during unsupervised learning of the first layer GRBMs. We found that using these additional unlabeled data helps slightly, but only when it was applied to the first layer.

[7] This is a deep convolutional network trained with additional labeled data generated from translations.

(a) Samples from a standard DBN

(b) Samples from MrDBN

Figure 5: Samples drawn from a DBN and a MrDBN trained on NORB. Samples are drawn from a single Markov chain sampling from each model. From left to right then top to bottom, each image is generated after 1,000 block Gibbs steps.

5.8% for MrDBN is significantly better than the error for the standard DBN. Comparing the error of a DBN learning $\Lambda$ to the previously reported error for a standard DBN, learning $\Lambda$ is found to be beneficial for classification.

To see the usefulness of the representation of each stream, we trained a separate top layer with labels for each of the streams. By doing these experiments, we are performing classification with features learned from a particular frequency component. These stream-specific top layer RBMs also use 4500 hidden nodes and are trained in the same way as MrDBNs. Table 2 shows the error rates.

| | 32x32 | 16x16 | 8x8 | Combined |
|---|---|---|---|---|
| MrDBN [32-16-8] | 11.2% | 10.7% | 9.2% | 7.1% |
| MrDBN [32-16] | 11.2% | 6.9% | N/A | 5.8% |
| DBN 32 | 7.4 % | N/A | N/A | N/A |

Table 2: Classification errors using each frequency stream. By combining features of multiple streams, we obtain reduction of error in the last column. Note that the two $16 \times 16$ stream is different because the $16 \times 16$ stream in MrDBN [32-16] is low-frequency and the $16 \times 16$ stream in MrDBN [32-16-8] is high-frequency.

What is surprising from Table 2 is that we can achieve a reasonable error rate of 9.2% with a resolution of only $8 \times 8$. The fact that high resolution is not a requirement of good recognition has been observed in human psychology [28]. In addition, we can clearly improve recognition by combining features from multiple streams. The combined model (which is a MrDBN) has a single RBM (with labels) layer on top. This is very different from a committee of classifiers, one from each of the streams.

Furthermore, we trained a standard DBN which took as its input the concatenated vector of the same image at the resolutions of $32 \times 32$, $16 \times 16$ and $8 \times 8$ *instead* of the Laplacian Pyramid decomposition. The test error achieved by this DBN is 7.2%, similar to that of a standard DBN trained on just $32 \times 32$ stereo images (which is 7.4%). This supports our hypothesis that generalization is improved by using the Laplacian Pyramid to decompose the input data into frequency components.

#### 4.2.2 TIMIT corpus

We tested MrDBN for phone recognition on the TIMIT acoustic-phonetic continuous read speech corpus[8]. TIMIT provides phone-level transcription of broadband recordings for American English speakers of different sexes and dialects. It is the standard benchmark dataset for phone recognition in the speech recognition research comunity. The corpus is divided

[8] http://www.ldc.upen.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S1.

into a 462-speaker training set, a 50-speaker validation set, and a 24-speaker core test set. We report our Phone Error Rates (PER) on the core test set while using the validation set to tune learning and model parameters.

The speech was analyzed using a 25-ms Hamming window with a 10-ms fixed frame rate. Our features are 40 Fourier-transform-based filter-bank coefficients distributed on a mel-scale (and energy) together with their first and second temporal derivatives. This results in a vector of dimensionality 123. The network inputs are contextual windows that are concatenations of 8 to 12 consecutive frames. The entire TIMIT training set contains over 1,000,000 training vectors and we form 8,700 minibatches of 128 training examples each. For example, we treat an input vector of 10 frames as a 2D image of 10 by 123, and we down-sample in the time domain only. Therefore, the corresponding low-resolution input would have dimensionality of 5 by 123. Following the training protocols of [29], We used 183 target class labels (i.e., 3 states for each one of the 61 phones). After estimating the hidden state probability using MrDBN, we perform Viterbi decoding using a bigram language model estimated from the training set. After decoding, the 61 phone classes were mapped to a set of 39 classes as in [30] for scoring.

Similar to the MrDBN used for NORB, we first learn a DBN with 3 hidden layers each containing 500 nodes on the low resolution stream (data is 5 by 123) and another DBN with 1000 hidden nodes layers on the high resolution stream (data is 10 by 123). We combine the two streams at the top layer and jointly fine-tune the model to minimize cross-entropy error of the 183 softmax labels using Stochastic Gradient Descent with Backprop. The learning rate, momentum and weight decay values were all selected by looking at the validation set errors.

See Figure 6 for the resulting error rates for both the standard DBN and the MrDBN. As can be seen in the figure, MrDBN reduces absolute test error rates by approximately 1.5%. We note that MrDBN obtained a core test set error rate of **20.3%**, which improves upon the current state-of-the-art recognition result (20.7% [9]) on TIMIT. See [29] for a detailed comparison of reported TIMIT results in literature[10].

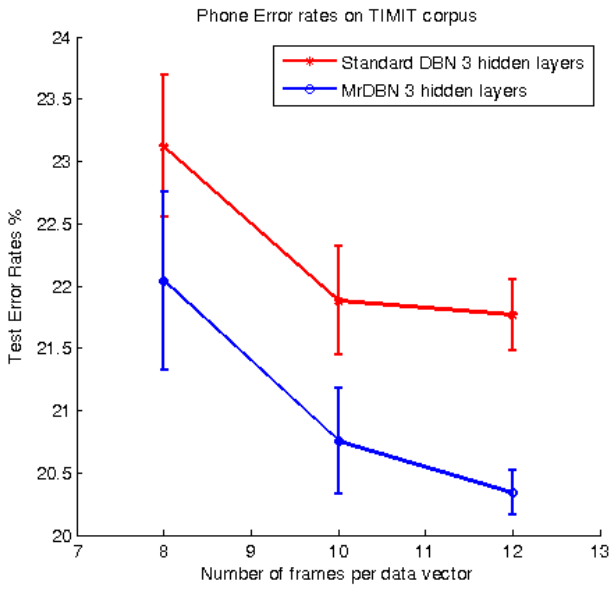



Figure 6: Test set error rates on the TIMIT Corpus. We plot the errors with respect to the number of input frames per data vector and their 95% confidence interval.

## 5 Conclusions

We have combined the widely used multiresolution framework with a popular deep generative model. By learning feature hierarchies on multiresolution data, MrDBN can generalize better than a standard DBN on the NORB classification task and the TIMIT phone recognition task. We have also shown that MrDBN is a superior generative model.

We hypothesize that training on lower resolution data can help regularize the network from overfitting on the details of training images. For generative learning, lower resolution and lower dimensional images also have a regularization effect by forcing the latent variables to model coarser structures of objects. In future work, we plan to apply MrDBN to problems with occlusions and noise, to speed up visual search, and adapt multiresolution learning to Deep Boltzmann Machines.

### Acknowledgments

We thank the anonymous reviewers for greatly improving the manuscript. This research was supported by the Natural Sciences and Engineering Research Council of Canada.

[9] Note that this result is from a much bigger network with 8 hidden layers of 2048 nodes each [29].

[10] We note that [31] used extra speaker identity labels (which are not part of the standard TIMIT evaluation) to achieve 19.6% PER by performing input transformations on input features to account for speaker differences.

## References


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

[2] R. Salakhutdinov and G. Hinton. Deep Boltzmann machines. In *AISTATS*, 2009.

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

[4] R. Salakhutdinov and H. Larochelle. Efficient learning of deep boltzmann machines. *AISTATS*, 2010.

[5] A. Mohamed, G. E. Dahl, and G. E. Hinton. Deep belief networks for phone recognition. In *NIPS Workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[6] G. E. Dahl, M. Ranzato, A. Mohamed, and G. E. Hinton. Phone recognition with the mean-covariance restricted Boltzmann machine. In *NIPS 23*. 2010.

[7] M. Ranzato, J. Susskind, V. Mnih, and G. E. Hinton. On deep generative models with applications to recognition. CVPR, 2011.

[8] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer, December 1993.

[9] W. Zhang, G. Zelinsky, and D. Samaras. Real-time accurate object detection using multiple resolutions. In *ICCV*, 2007.

[10] Y. Amit, D. Geman, and X. Fan. A coarse-to-fine strategy for multi-class shape detection. *IEEE Transactions PAMI*, 26(12), 2004.

[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[12] T. F. Cootes, C. J. Taylor, and A. Lantis. Active Shape Models: Evaluation of Multi-Resolution Method for Improving Image Search. In *BMVC*, 1994.

[13] C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid Methods in Image Processing, 1984.

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

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

[16] T. Tieleman and G. E. Hinton. Using fast weights to improve persistent contrastive divergence. In *ICML*, volume 382, page 130. ACM, 2009.

[17] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[18] G. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. In *NIPS*, 2006.

[19] H. Lee, C. Ekanadham, and A. Ng. Sparse deep belief net model for visual area V2. In *NIPS*, 2007.

[20] M. Ranzato and G. Hinton. Modeling pixel means and covariances using factorized third-order boltzmann machines. *CVPR*, 2010.

[21] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of ICML*, 2010.

[22] A. Courville, J. Bergstra, and Y. Bengio. A spike and slab restricted boltzmann machine. In *Proceedings of AISTATS*, 2011.

[23] P. Burt and T. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31:532–540, 1983.

[24] J.M. Susskind. The Toronto Face Database. Technical report, 2011. `http://aclab.ca/users/josh/TFD.html`.

[25] Y. LeCun, F. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

[26] V. Nair and G. E. Hinton. 3-D object recognition with deep belief nets. In *NIPS 22*, 2009.

[27] L. Quoc, J. Ngiam, Z. Chen, D. Chia, P. W. Koh, and A. Ng. Tiled convolutional neural networks. In *NIPS 23*. 2010.

[28] P. Sinha, B. Balas, Y. Ostrovsky, and R. Russell. Face recognition by humans: Nineteen results all computer vision researchers should know about. *Proceedings of the IEEE*, 94(11):1948–1962, 2006.

[29] A. Mohamed, G. Dahl, and G. Hinton. Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, 2011.

[30] K.-F. Lee and H.-W. Hon. *Speaker-Independent Phone Recognition Using Hidden Markov Models*, volume CMU-CS-88-121. CMU Computer Science Dept., Mar. 31, 1988.

[31] A. Mohamed, T. N. Sainath, G. E. Dahl, B. Ramabhadran, G. E. Hinton, and M. A. Picheny. Deep belief networks using discriminative features for phone recognition. pages 5060–5063. ICASSP, 2011.