Sparse deep belief net models for visual area V2

Chaitanya Ekanadham

Undergraduate Honors Thesis, Symbolic Systems Program, Stanford University

Advisor: Andrew Ng (Computer Science)
Second Reader: Honglak Lee (Computer Science)

June 14, 2007

Abstract[1]

Motivated in part by the hierarchical organization of the neocortex, a number of recently proposed algorithms have tried to learn hierarchical, or "deep," structure from unlabeled data. While several authors have formally or informally compared their algorithms to computations performed in visual area V1 (and the cochlea), little attempt has been made thus far to evaluate these algorithms in terms of their fidelity for mimicking computations at deeper levels in the cortical hierarchy. This thesis describes an unsupervised learning model that faithfully mimics certain properties of visual area V2. Specifically, we develop a sparse variant of the deep belief networks described by Hinton et al. (2006). We learn two layers of representation in the network, and demonstrate that the first layer, similar to prior work on sparse coding and ICA, results in localized, oriented, edge filters, similar to the gabor functions known to model simple cell receptive fields in area V1. Further, the second layer in our model encodes various combinations of the first layer responses in the data. Specifically, it picks up both collinear ("contour") features as well as corners and junctions. More interestingly, in a quantitative comparison, the encoding of these more complex "corner" features matches well with the results from Ito & Komatsu's study of neural responses to angular stimuli in area V2 of the macaque. This suggests that our sparse variant of deep belief networks holds promise for modeling more higher-order features that are encoded in visual cortex. Conversely, one may also interpret the results reported here as suggestive that visual area V2 is performing computations on its input similar to those performed in (sparse) deep belief networks. This plausible relationship generates some intriguing hypotheses about V2 computations.

[1] This thesis is an extended version of an earlier paper by Honglak Lee, Chaitanya Ekanadham, and Andrew Ng titled "Sparse deep belief net model for visual area V2."

1 Introduction

The last few years have seen significant interest in “deep” learning algorithms that learn layered, hierarchical representations of high-dimensional data. [1, 2, 3, 4]. Much of this work appears to have been motivated by the hierarchical organization of the cortex, and indeed authors frequently compare their algorithms’ output to the oriented simple cell receptive fields found in visual area V1 (e.g., [5, 6, 2]). Indeed, some of these models are often viewed as first attempts to elucidate what learning algorithm (if any) the cortex may be using to model natural image statistics. However, to our knowledge no serious attempt has been made to directly relate, such as through quantitative comparisons, the computations of these deep learning algorithms to areas deeper in the cortical hierarchy, such as visual areas V2 or V4. In this paper, we develop a sparse variant of Hinton’s deep belief network algorithm ([1]), and measure the degree to which it faithfully mimics biological measurements of area V2 on the macaque. Specifically, we take Ito & Komatsu’s ([7]) characterization of area V2 in terms of its responses to a large class of angled bar stimuli, and quantitatively measure the degree to which the deep belief network algorithm generates similar responses. Deep architectures attempt to learn hierarchical structure, and hold the promise of being able to first learn simple concepts, and then successfully build up more complex concepts by composing together the simpler ones. For example, Hinton et al. [1] proposed an algorithm based on learning individual layers of a hierarchical probabilistic graphical model from the bottom up. Bengio et al. [3] proposed a similar greedy algorithm based on auto-encoders. Ranzato et al. [2] developed an energy-based hierarchical algorithm, based on a sequence of sparsified auto-encoders/decoders.

In related work, several studies have compared models such as these, as well as nonhierarchical/non-deep learning algorithms, to the response properties of neurons in area V1.

A study by van Hateren and van der Schaaf [8] showed that the filters learned by independent components analysis (ICA) [9] on natural image data match very well with the classical receptive fields of V1 simple cells. (Filters learned by sparse coding [10] also give responses similar to V1 simple cells.) Our work takes inspiration from the work of van Hateren and van der Schaaf, and represents a study that is done in a similar spirit, only extending the comparisons to a deeper area in the cortical hierarchy, namely visual area V2.

2 Biological comparison

2.1 Features in early visual cortex: area V1

The selectivity neurons for oriented bar stimuli in cortical area V1 has been well documented [11, 12]. The receptive fields of simple cells in area V1 are localized, oriented, bandpass filters that resemble gabor filters. Several authors have proposed models that have been either formally or informally shown to replicate the gabor-like properties of V1 simple cells. Many of these algorithms, such as [10, 9, 8, 6], compute a (approximately or exactly) sparse representation of the natural stimuli data. These results are consistent with Barlow's "efficient coding hypothesis" which posits that the goal of early visual processing is to encode visual information as efficiently as possible ([13]). Some hierarchical extensions of these models [14, 6, 15] are able to learn features that are more complex than simple oriented bars. For example, hierarchical sparse models of natural images have accounted for complex cell receptive fields [16], topography [17, 6], collinear edges and contour coding [18]. Other models, such as [19], have also been shown to give V1 complex cell-like properties.

2.2 Features in visual cortex area V2

It remains unknown to what extent the previously described algorithms can learn higher order features that are known to be encoded further down the ventral visual pathway. In addition, the response properties of neurons in cortical areas receiving projections from area V1 (e.g., area V2) are not nearly as well documented. It is uncertain what stimuli cause V2 neurons to respond optimally ([20]). One V2 study by [21] reported that the receptive fields in this area were similar to those in the neighboring areas V1 and V4. The authors interpreted their findings as suggestive that area V2 may serve as a place where different channels of visual information are integrated. However, quantitative accounts of responses in area V2 are few in number. In the literature, we identified two sets of data that give us a good starting point for making measurements to determine whether our algorithms may be computing similar functions as in area V2.

In one of these studies, Ito and Komatsu ([7]) investigated how V2 neurons responded to angular stimuli. They summarized each neuron's response with a two-dimensional visualization of the stimuli set called an angle profile. By making several axial measurements within the profile, the authors were able to compute various statistics about each neuron's selectivity for angle width, angle orientation, and for each separate line component of the angle (see Figure 1 caption for details). Approximately 80% of the neurons responded to specific angle stimuli. They found neurons that were selective for only one line component of its peak angle as well as neurons selective for both line components. These neurons yielded angle profiles resembling those of Cell 1 and Cell 2 in Figure 1 (right), respectively. In addition, several neurons exhibited a high amount of selectivity for its peak angle. No neurons were found that had more elongation in a diagonal axis than in the horizontal or vertical axes, indicating that neurons in V2 were not selective for angle width or orientation. Therefore, an important conclusion made in [7] was that a V2 neuron's response to an angle stimulus is highly dependent on its responses to each individual line component of the angle. While the dependence was often observed to be simply additive, as was the case with neurons yielding profiles like those of Cells 1 and 2 in Figure 1 (right), this was not always the case. 29 neurons had very small peak response areas and yielded profiles like that of Cell 3 in Figure 1 (right), thus indicating a highly specific tuning to an angle stimulus. While the former responses suggest a simple linear computation of V1 neural responses, the latter responses suggest a nonlinear computation [20]. The angle profile analysis tools employed in [7] are very useful in characterizing the response properties, and we use these methods to evaluate our own model.

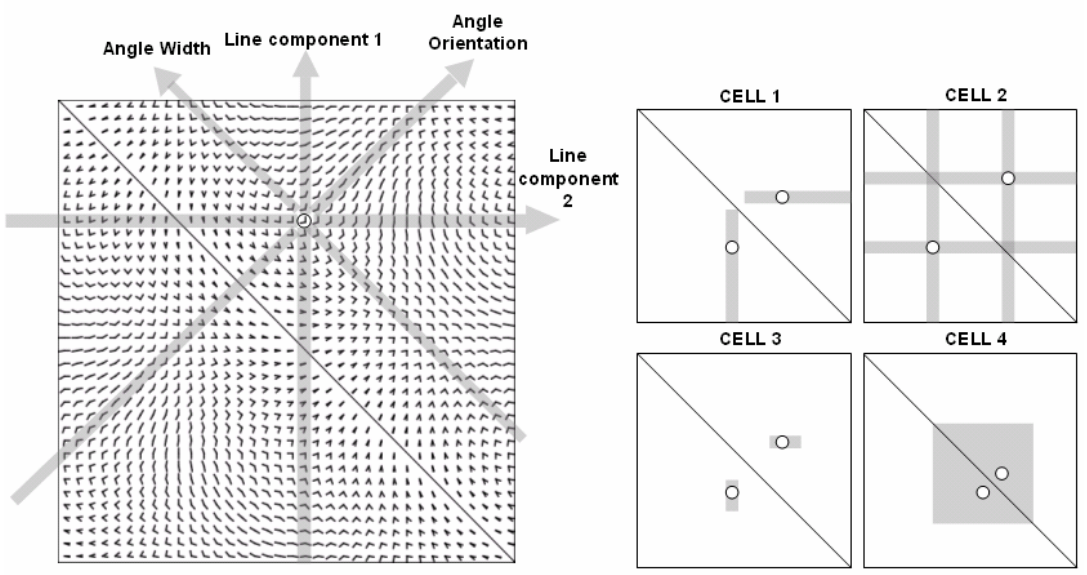

Figure 1: Left: Visualization of angle profiles (adapted from [7]). Note that the upper and lower triangles contain the same stimuli. Darkened squares correspond to stimuli that elicited a large response. The peak responses are circled. Above each directed axis is the property of the angle that remains constant along that axis. The arrangement of the profile is so that that one line component remains constant as one moves along any vertical or horizontal axis. The angle width remains constant as one moves along the diagonal indicated The angle orientation remains constant as one moves along the diagonal indicated. After identifying the optimal stimuli for a neuron in the profile, the number of stimuli along these various axes eliciting responses larger than 80% of the peak response measure the neuron's tolerance to perturbations to the line components, peak angle width, and orientation, respectively. Right: Examples of 4 prototypical angle profiles for V2 neurons (adapted from [7]). As before, stimuli eliciting large responses are highlighted. Cell 1 has one axis of elongation, indicating selectivity for one orientation. Cell 2 has two axes of elongation, and responds strongly so long as either of two edge orientations is present. Cell 3 has a selective response to a stimulus, so there is no elongation of any axis. Cell 4 has no clear axis of elongation.

Another study by Hegdé and Van Essen ([22]) studied the responses of a population of V2 neurons to complex contour and grating stimuli. They found several V2 neurons responding maximally for angles, and the distribution of peak angles for these neurons is consistent with that found by [7]. In addition, several V2 neurons responded maximally for complex shapes such as intersections, tri-stars, five-point stars, circles, and arcs.

Computational models of more complex features than bars have produced mostly features that encode the co-activation of collinear low-level features. That is, no learning algorithms to our knowledge have produced corner or junction features of the type thought to be encoded in area V2 of the cortex (as suggested by the previously mentioned neurophysiological studies).

3 Algorithm

Hinton et al. [1] proposed an algorithm for learning deep belief networks based on treating each layer as a restricted Boltzmann machine (RBM), and greedily training the network one layer at a time from the bottom up ([23, 1]). When trained on natural images, this algorithm does not, however, yield localized, gabor-like basis functions (see Figure 3 in section 4.2). Based on results from other methods (e.g., sparse coding [10], ICA [9], heavy-tailed models [6], and energy based models [2]), sparseness seems to play a fairly fundamental role in learning gabor-like filters. Therefore, we modify Hinton's learning algorithm to enable deep belief nets to learn sparse representations.

3.1 Sparse restricted Boltzmann machines

We begin by describing the restricted Boltzmann machine (RBM), and present a modified version of it. An RBM has a set of hidden units $\boldsymbol{h}$, a set of visible units $\boldsymbol{v}$, and symmetric connections between these two layers represented by a weight matrix W. Suppose that we want to model k-dimensional real-valued data using an undirected graphical model with n binary hidden units. The negative log probability of any state in the RBM is given by the following energy function:

$$-\log P(\mathbf{v},\mathbf{h}) = E(\mathbf{v},\mathbf{h}) = \frac{1}{2\sigma^2}\sum_i \mathbf{v}_i^2 - \frac{\lambda}{\sigma^2}\left(\sum_i c_i\mathbf{v}_i + \sum_j b_j\mathbf{h}_j + \sum_{i,j}\mathbf{v}_i w_{ij}\mathbf{h}_j\right) \tag{1}$$

Here, *σ* and λ are parameters, and *b* and *c* are bias weights for the hidden and visible units, respectively. For the results reported in this paper, λ was fixed at 1 (see Section 7.1 for other variations). Informally, the maximum likelihood parameter estimation problem corresponds to learning *W*, *b,* and *c* so as to minimize the energy of states drawn from the data distribution, and raise the energy of states corresponding to states that are improbable given the data.

Under this energy model, we can easily compute the conditional probability distributions. Holding either *h* or *v* fixed, we can sample from the other as follows:

$$\begin{aligned} P(\mathbf{v}_i|\mathbf{h}) &= \mathcal{N}(\lambda(c_i + \textstyle\sum_j w_{ij}\mathbf{h}_j), \sigma^2) && (2)\\ P(\mathbf{h}_j|\mathbf{v}) &= g(\tfrac{\lambda}{\sigma^2}(b_j + \textstyle\sum_i w_{ij}\mathbf{v}_i)) && (3) \end{aligned}$$

where *N(·)* is the Gaussian density, and *g(·)* is the logistic function. For learning the parameters of the model, the objective is to maximize the log-likelihood of the data. We also want hidden unit activations to be sparse. Thus, we impose a constraint that the expected activations of the hidden units are kept to a (low) fixed level *p* (see Section 7.2 for details). Thus, given a training set {v(1), . . . , v(m)} comprising *m* examples, we pose the following optimization problem:

$$\begin{aligned} \text{minimize}_{\{w_{ij},c_i,b_j\}} \quad & -\textstyle\sum_{i=1}^m \log \sum_{\mathbf{h}} P(\mathbf{v}^{(i)},\mathbf{h}^{(i)}) && (4)\\ \text{subject to} \quad & \tfrac{1}{m}\textstyle\sum_{i=1}^m \mathbb{E}[h_j^{(i)}|\mathbf{v}^{(i)}] = p, \quad \forall j = 1,...,n, && (5) \end{aligned}$$

where *E[·]* is the conditional expectation given the data, and *p* is the constant controlling the sparseness of the hidden units h_j . In principle, this problem can be solved using the projected gradient method (i.e. updating the parameters by taking a gradient descent step followed by a projection into the constraint set satisfying Equation 5 (see [24]). However, computing the gradient of the likelihood term is computationally expensive. Fortunately, the contrastive divergence learning algorithm has been shown to be effective in approximately optimizing the likelihood ([25]). Therefore, on each iteration we apply the contrastive divergence update rule, followed by a projection to the constraint space. The details of our procedure are summarized in Algorithm 1 (see also Section 7 for further technical details).

3.2 Learning deep networks using sparse RBMs

Once a layer of the network is trained, the parameters *W*, *b*, *c*, are frozen and the hidden unit values given the data are inferred using Equation 3. These inferred values serve as the "data" used to train the next higher layer in the network. Hinton et al. [1] showed that by repeatedly applying this procedure, one can learn a multilayered deep belief network. Furthermore, this iterative "greedy" algorithm actually optimizes a variational bound on the data likelihood as long as each layer has at least as many units as the layer below (although in practice this is not necessary to arrive at a desirable solution; see [1] for a detailed discussion). In our experiments, we learn a three-layer network (comprising one observed layer, and two hidden layers) using the abovementioned sparse RBM learning algorithm.

Algorithm 1 Sparse RBM learning algorithm

```
1. Update the parameters using contrastive divergence learning rule. More specifically,
       $w_{ij} := w_{ij} + \alpha(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{recon}})$
       $c_i := c_i + \alpha(\langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{recon}})$
       $b_i := b_i + \alpha(\langle b_i \rangle_{\text{data}} - \langle b_i \rangle_{\text{recon}})$,
   where $\alpha$ is a learning rate, and $\langle \cdot \rangle_{\text{recon}}$ is an expection over the reconstruction data, estimated
   using one iteration of Gibbs sampling (as in Equations 2,3).
2. Project to the constraint space: update the hidden bias $b_j$'s using gradient descent until the
   sparseness constraint (5) is satisfied.
3. Repeat Steps 1 and 2 until convergence.
```

4 Visualization

4.1 Learning "strokes" from handwritten digits

We applied this algorithm to the MNIST handwritten digit data (see section 7.3 for details about this data). We learned a sparse RBM with 200 visible units and 200 hidden units. The learned bases are shown in Figure 2. Each basis corresponds to one column of the learned weight matrix W.[2] Many bases found by the algorithm roughly correspond to different "strokes" of which handwritten digits are comprised. This is consistent with results obtained by applying different algorithms to learn sparse representations of this data set (e.g., [2, 5]).

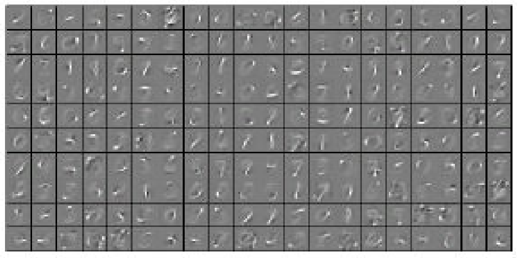

Figure 2: Bases learned from the MNIST data using a sparse RBM.

4.2 Learning from natural images

We also applied the algorithm to a training set a set of 14 x 14 natural images patches, taken from a dataset compiled by van Hateren (see section 7.3 for details about this data). We learned a sparse RBM model with 196 visible units and 200 hidden units. The learned bases are shown in Figure 3 (left); they are oriented, gabor-like bases and resemble the receptive fields of V1 simple cells. In contrast, learning without the sparseness constraint resulted in bases that were neither oriented nor localized, as shown in Figure 3 (right).

4.3 Learning a two-layer model of natural images using sparse RBMs

We learned a two-layer network by stacking one sparse RBM on top of another (see Section 3.2). After learning, the second layer weights were quite sparse—most weights were

[2] In the sparse coding literature ([10]), the columns of the weight matrix W correspond to "basis images." The idea is that the data samples can be thought of as linear combinations of these basis images with sparse coefficients. The coefficient values correspond to the hidden unit activations (or activation probabilities) and model neural activations.

very small, and only a few were either highly positive or highly negative. Positive elements represent excitatory connections between model V1 and model V2 units, whereas negative elements represent inhibitory connections. By visualizing the second layer bases as shown in Figure 4, we observed bases that encoded collinear first layer bases as well as corners and other edge junctions. This shows that by extending the sparse RBM to two layers and using greedy learning, the model is able to learn bases that encode contours, angles, and junctions of edges. We used p=0.02 for the first layer and tried values ranging from 0.02 to 0.1 for the second layer. For p values 0.05 and more for the second layer, the results were qualitatively similar to Figure 4.

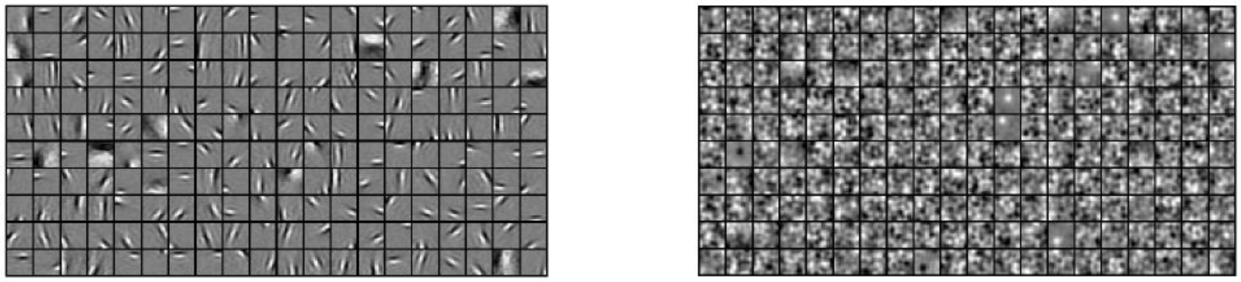
Figure 3: Left: 200 Bases learned from the van Hateren natural image dataset using the sparse RBM algorithm. Results were similar for values of p ranging from 0.01 to about 0.05. The parameter σ was chosen to be 0.5 and decayed (see Section 7.1). Right : Bases learned using the regular (non-sparse) RBM.

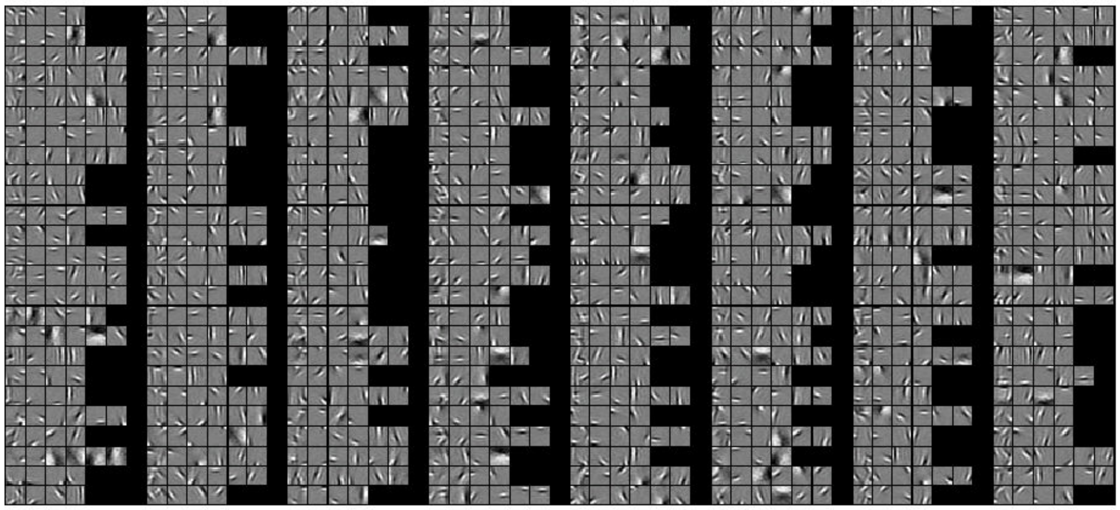
Figure 4: Visualization of 200 second layer bases (model V2 receptive fields) learned from natural images (p = 0.1). Each small group of 4-6 (arranged in a row) images corresponds to one model V2 unit; the leftmost patch in the group is a visualization of the model V2 basis, and is obtained by taking a weighted linear combination of the first layer "V1" bases to which it has strong positive connections. The next few patches in the group show the first layer bases that have the strongest weight connection to the model V2 unit.

4.4 Saliency sampling

We also tested an interesting hypothesis that if the natural image patch data is sampled not according to a uniform distribution (over all positions patches in the images) but rather a distribution that is biased toward more visually salient features in the images, then the resulting second-layer sparse RBM representation would be different. In particular, we expected there to be a higher number of "interesting" corners and junctions encoded in the second layer learned from saliency-sampled data, rather than simple oriented bar or collinear features. We used the saliency maps described by [27] to sample the van Hateren natural image data based on saliency. However, we could observe no quantifiable difference between the bases (both first and second-layer) learned from uniformly sampled and saliency sampled data.

5 Evaluation experiments

We now quantitatively compare the algorithm's learned responses to biological measurements. It is important to note that the sparse RBM model was not even minimally tuned to match any biological phenomena, and is described entirely in section 3. The only free parameters were p (which is chosen based on rough estimates of the average firing rate of neurons in V1 and V2 when exposed to natural scenes [26]), and σ. The results reported used $\sigma = 0.4$, but are very insensitive to this choice.

5.1 Method: Ito-Komatsu paper protocol

We now describe the procedure to compare our model with experimental data in [7]. We generated a stimulus set consisting of the same set of angles (pairs of edges) as [7]. To identify the spatial "center" of each model neuron's receptive field, we translate all stimuli densely over the 14x14 input patch, and identify the position at which the maximum response is elicited. All measures are then taken with all angle stimuli centered at this position. Using these stimuli, we compute the hidden unit probabilities from our model V1 and V2 neurons. In other words, for each stimulus we compute the first hidden layer activation probabilities, then feed this probability as data to the second hidden layer and compute the activation probabilities again in the same manner. Following a protocol similar to [7], we also eliminate from consideration the model neurons that do not respond strongly to corners and edges. Some representative results are shown in Figure 5. Notice that the four angle profiles resemble those of the prototypical profiles in Figure 1 (right). The results are consistent with the results from [7] in that there are model V2 units that are sensitive to one or both of the line components of the peak angle, as well as units responding only when the conjunction of the line components is present (see the top-left, bottom-left, and bottom-right angle profiles in Figure 5, respectively). Thereby, the bases seem to encode the edge junctions or crossings.

We also compute similar summary statistics as [7] (described in Figure 1 (left)), that more quantitatively measure the distribution of V2 or model V2 responses to the different angle stimuli. Figure 6 plots the responses of our model, together with V2 data taken from [7]. Along many dimensions, the results from our model match the neurophysiological data quite well.

5.2 Complex shaped stimuli

Our second experiment represents a comparison to Hegde and van Essen [22]. We generated a stimulus set comprising complex shaped stimuli [22]: angles, single bars, tri-stars, and arcs/circles, and measured the response of the second layer of our sparse RBM model to these stimuli. We observe that many V2 bases are activated mainly by one of these different stimulus classes. For example, some model V2 neurons activate maximally to single bars; some maximally activate to (acute or obtuse) angles, and others to tri-stars (see Figure 7). Furthermore, the number of V2 bases that are maximally activated by acute angles is significantly larger than

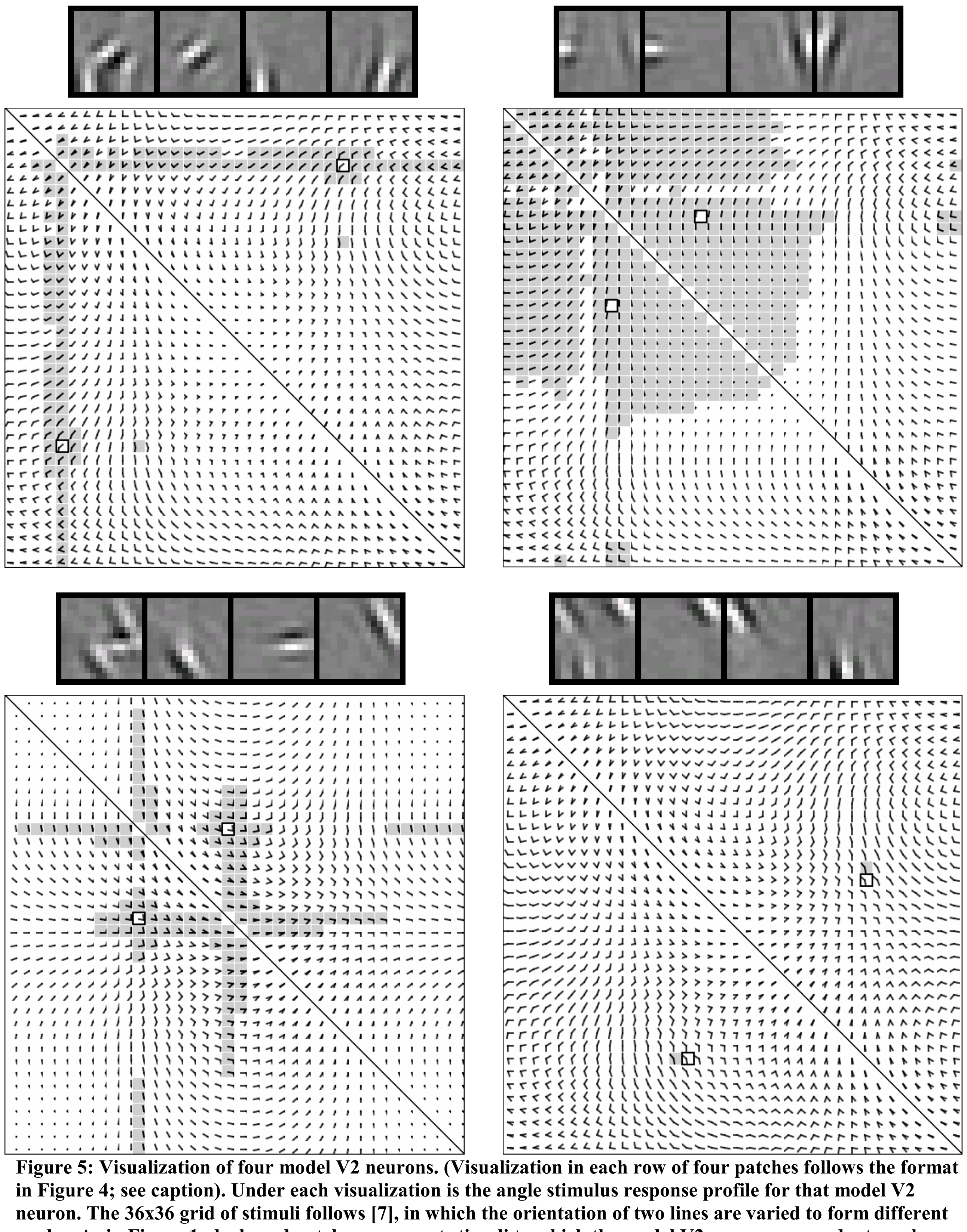

Figure 5: Visualization of four model V2 neurons. (Visualization in each row of four patches follows the format in Figure 4; see caption). Under each visualization is the angle stimulus response profile for that model V2 neuron. The 36x36 grid of stimuli follows [7], in which the orientation of two lines are varied to form different angles. As in Figure 1, darkened patches represent stimuli to which the model V2 neuron responds strongly; also, a small black square indicates the overall peak response.

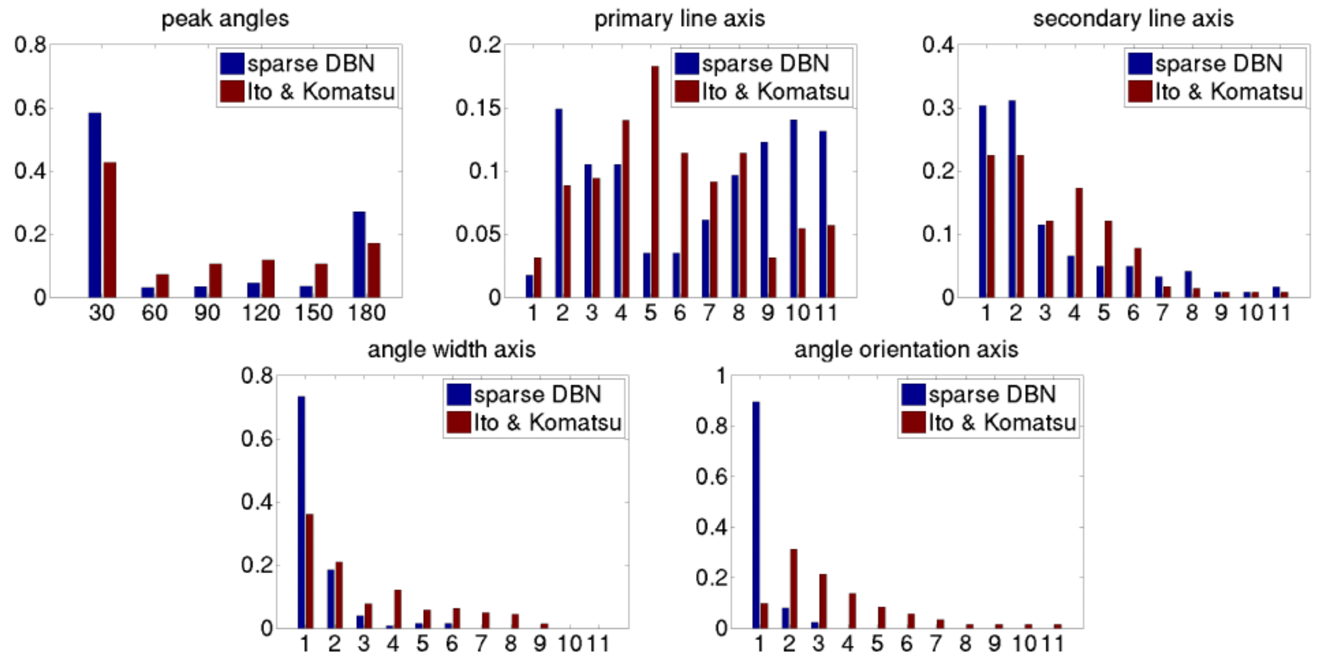

Figure 6: Images show distributions over stimulus response statistics from our algorithm (blue) and in data taken from [7]. The five figures show respectively (i) the distribution over peak angle response (ranging from 0 to 180 degrees), (ii) distribution over tolerance to primary line component (Figure 1 left, in dominant vertical or horizontal direction), (iii) distribution over tolerance to secondary line component (Figure 1 left, in non-dominant direction), (iv) tolerance to angle width (Figure 1 left), (v) tolerance to angle orientation (Figure 1 left). See Figure 1 caption, and [7], for details.

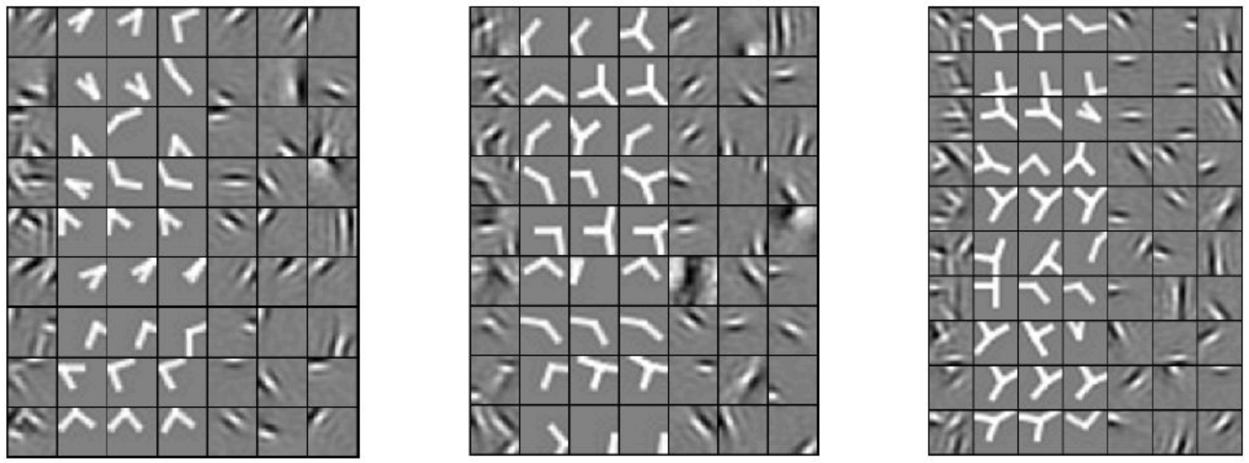

Figure 7: Visualization of representative model V2 neurons that maximally activates for various complex stimulus. For each V2 bases visualization, the first column shows linear combination of top-three V1 components that comprises the V2 bases; columns 2-4 represents the top-three optimal stimulus; the remaining three columns shows the top-3 V1 bases themselves. The V2 bases shown in the figures are maximally activating for acute angles (left), obtuse angles (middle), and tri-stars and junctions (right).

the number of obtuse angles, and the number of V2 bases that respond maximally to the tri-stars were much fewer than both preceding cases. This is also consistent with the results described in [22].While there were a few model V2 units that responded maximally for arcs and five-point stars, a more detailed analysis of the features they were encoding showed that these units were not *selectively* activating for these complex shapes. Rather, since these shapes tend to have higher overall pixel values, some units would elicit a response higher than for any other stimulus.

5 Discussion

Two important questions to be asked are 1) Are these angular and other complex features unique to the sparse RBM learning algorithm, and 2) how should these results affect our view of computations done in area V2? It is quite obvious that our learning algorithm is probably not the only algorithm that can learn angle bases from natural images. However, if it were the case that other traditional algorithms such as sparse coding or ICA definitively do *not* learn such representations at the second or higher layer (this has not been verified in detail, although our preliminary results do suggest this), there would be some interesting implications about what we know about area V2 computations. For instance, it could suggest that V2 units not only encode just correlated V1 outputs, but also encode *co-activations* of V1 outputs. This allows relatively infrequent co-activations of V1 units to be encoded by V2 units. The sparse RBM algorithm may be achieving this via the sigmoid function applied to the matrix-vector product which roughly "thresholds" the product whether or not a hidden unit will fire. The fact that sparse RBM hidden unit values are constrained to be in [0,1] is also a computational element not present in other algorithms, of which the precise effects need to be explored in further detail.

6 Conclusions

We presented a sparse variant of the deep belief networks model. When trained on natural images, this model learns local oriented, edge filters in the first layer. More interestingly, the second layer captures a variety of both collinear ("contour") features as well as corners and junctions, that in a quantitative comparison to measurements of area V2 or the macaque taken by Ito & Komatsu, appeared to give responses that were similar along several dimensions. Given that the algorithm was derived by sparsifying deep belief nets and was not otherwise tuned to match biological data, we were in retrospect surprised by the quality of the match. This by no means indicates that the cortex is a sparse RBM, but perhaps is more suggestive of contours, corners and junctions being fundamental to the statistics of natural images. Nonetheless, we believe that these results also suggest that sparse deep algorithms, such as our sparse variant of deep belief nets, hold promise for modeling higher-order features such as might be computed in the ventral visual pathway in the cortex.

7 Appendix

7.1 Sigma decay

In the RBM model described in Equations 1, 2, and 3, the parameter σ controls both the noise of the reconstruction (Equation 2) and the "steepness" of the sigmoid function g in Equation 3. To achieve good reconstructions of the data when solving the optimization problem described in Equations 4 and 5, we decayed σ by a factor of 0.99 on every training epoch as long as it was above some small value (in the results reported we used 0.05). In this way, the reconstruction error (i.e. the squared difference between the image and the RBM's reconstruction of the image using one full iteration of Gibbs Sampling) was reduced, and the hidden unit activation probabilities became more skewed towards 0 and 1 (the model becomes almost deterministic for very small σ). The decaying of σ was not necessary to learn gabor-like bases at the first layer, nor was it necessary to learn interesting features at the second layer. However, if sigma is not decayed, the reconstruction errors are naturally higher than if it is decayed. There is a tradeoff between achieving good reconstructions and losing information by making the hidden unit probabilities more deterministic. It would be nice if these two properties were controlled by two independent parameters. This was our motivation in introducing the parameter λ in our energy model. However, changing the value of λ produced qualitatively similar results.

7.2 Hidden bias adaptation

To satisfy the constraint described in Equation 5, we first calculate for each hidden unit the average probability of firing over the data. Subtracting the parameter *p* (see Section 3.1) from this value yields the update rule:

$$\mathbf{h}_j := \mathbf{h}_j - \eta\left(\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[\mathbf{h}_j^{(i)}|\mathbf{v}^{(i)}] - p\right) \qquad (6)$$

where η is the learning rate, *m* is the number of samples, and $\mathbf{v}^{(i)}$ denotes the *i*'th image sample. $E(\boldsymbol{h}_j^{(i)}|\boldsymbol{v}^{(i)})$ is calculated using Equation 3. In practice, we performed only one update step on each epoch, so the average hidden unit activations were not necessarily *p* in the leading few training epochs. We found that in practice, performing a single gradient step on each epoch seemed to work better than performing multiple gradient steps on each epoch so that the activations were exactly *p*. This may be due to sensitivity of the constrained optimization space to the initialization of the parameters.

7.3 Data sets and preprocessing

1. **MNIST**: Downloaded from `http://yann.lecun.com/exdb/mnist/`. The data was normalized to [0,1] and PCA whitening was applied to reduce the dimension to 200 principal components for computational efficiency. Very similar results were obtained without whitening.

2. **Natural images**: The images were obtained from `http://hlab.phys.rug.nl/imlib/index.html`. The images were also gamma corrected (gamma=0.5), and whitened using 1/f whitening (see [10] for details of the 1/f whitening procedure). 100,000 14x14 images patches were randomly sampled from an ensemble of 2000 images.

3. **Angle stimuli:** The stimulus set created by generating a binary-mask image, that is then scaled to normalize contrast. To determine this scaling constant, we used single bar images by translating and rotating to all possible positions, and fixed the constant such that the top 0.5% (over all translations and rotations) of the stimuli activate the model V1 cells. This normalization step corrects for the RBM having been trained on a data distribution (natural images) that had very different contrast ranges than our test stimulus set.

4. **Random stimuli**: In detail, we generated a set of random low-frequency stimulus, by generating small random KxK (K=2,3,4) images with each pixel drawn from a standard normal distribution, and rescaled the image using bicubic interpolation to 14x14 patches. These stimuli are scaled such that about 2% of the V2 bases fires maximally to these random stimuli. We then exclude the V2 bases that are maximally activated to these random stimuli from further analysis.

5. **Complex stimuli**: The same protocol was followed as with the angle stimuli.

7.4 MATLAB pseudo-code for the sparse RBM training algorithm

```
% numsamples = # of image patch samples
% numdims = # of visible units
% numhid = # of hidden units
α = 0.01
σ = 0.5
batchSize = 200
W := randn(numdims, numhid)
vbias := randn(numdims,1)
hbias := randn(numhid,1)

[W, hbias, vbias] = train_rbm(data, W, hbias, vbias, σ, α)
    while (not converged)
        for each training batch $X_{numdims \times batchSize}$ (randomly sample batchSize patches from data w/o replacement)
            poshidprobs := hidden unit probabilities given X (use Equation 3)
            poshidstates := sample using poshidprobs
            negdata := reconstruction of visible values given poshidstates (use Equation 2)
            neghidprobs := hidden unit probabilities given negdata (use Equation 3)
            $W := W + \alpha(X*poshidprobs^T - negdata*neghidprobs^T)/batchSize$
            $vbias := vbias + \alpha(\text{rowsum}(X) - \text{rowsum}(negdata))/batchSize$
            error := SquaredDiff(X,negdata)
        end for
        update hbias (use Equation 6)
        if (σ>0.05)
            σ := σ*0.99
        end if
    end while
end train_rbm
```

Acknowledgements

This work was done in collaboration with Honglak Lee and Andrew Ng. I would like to thank both of them for their helpful guidance and patience throughout the term of my research.

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] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun. Efficient learning of sparse representations with an energy-based model. In NIPS, 2006.

[3] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In NIPS, 2006.

[4] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In ICML, 2007.

[5] G. E. Hinton, S. Osindero, and K. Bao. Learning causally linked MRFs. In AISTATS, 2005.

[6] S. Osindero, M.Welling, and G. E. Hinton. Topographic product models applied to natural scene statistics. Neural Computation, 18:381–344, 2006.

[7] M. Ito and H. Komatsu. Representation of angles embedded within contour stimuli in area v2 of macaque monkeys. The Journal of Neuroscience, 24(13):3313–3324, 2004.

[8] J. H. van Hateren and A. van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. Proc.R.Soc.Lond. B, 265:359–366, 1998.

[9] Anthony J. Bell and Terrence J. Sejnowski. The ‘independent components’ of natural scenes are edge filters. Vision Research, 37(23):3327–3338, 1997.

[10] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature, 381:607–609, 1996.

[11] D. Hubel and T. Wiesel. Receptive fields and functional architecture of monkey striate cortex. Journal of Physiology, 195:215–243, 1968.

[12] R. L. DeValois, E. W. Yund, and N. Hepler. The orientation and direction selectivity of cells in macaque visual cortex. Vision Res., 22:531–544, 1982a.

[13] H. B. Barlow. The coding of sensory messages. Current Problems in Animal Behavior, 1961.

[14] P. O. Hoyer and A. Hyvarinen. A multi-layer sparse coding network learns contour coding from natural images. Vision Research, 42(12):1593–1605, 2002.

[15] Y. Karklin and M. S. Lewicki. A hierarchical bayesian model for learning non-linear statistical regularities in non-stationary natural signals. Neural Computation, 17(2):397–423, 2005.

[16] A. Hyvarinen and P.O. Hoyer. Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces. Neural Computation, 12(7):1705–1720, 2000.

[17] Aapo Hyv¨arinen, Patrik O. Hoyer, and Mika O. Inki. Topographic independent component analysis. Neural Computation, 13(7):1527–1558, 2001.

[18] A. Hyvarinen, M. Gutmann, and P. O. Hoyer. Statistical model of natural stimuli predicts edge-like pooling of spatial frequency channels in v2. BMC Neuroscience, 6:12, 2005.

[19] Laurenz Wiskott and Terrence Sejnowski. Slow feature analysis: Unsupervised learning of invariances. Neural Computation, 14(4):715–770, 2002.

[20] G. Boynton and J. Hegde. Visual cortex: The continuing puzzle of area v2. Current Biology, 14(13):R523–R524, 2004.

[21] J. B. Levitt, D. C. Kiper, and J. A. Movshon. Receptive fields and functional architecture of macaque v2. Journal of Neurophysiology, 71(6):2517–2542, 1994.

[22] J. Hegdé and D.C. Van Essen. Selectivity for complex shapes in primate visual area v2. Journal of Neuroscience, 20:RC61–66, 2000.

[23] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.

[24] Y. Censor and S. A. Zenios. Parallel Optimization: Theory, Algorithms and Applications. Oxford University Press, 1997.

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

[26] B. A. Olshausen and D. J. Field. How close are we to understanding V1? Neural Computation, 17(8):1665–1699, 2005.

[27] D. Walther and C. Koch. Modeling attention to salient proto-objects. Neural Networks, 19, 1395-1407, 2006.