1999-05

Favoring Generalists over Specialists: How Attentional Biasing Improves Perceptual Category Learning

Williamson, James

Boston University Center for Adaptive Systems and Department of Cognitive and Neural Systems

Favoring generalists over specialists: How attentional biasing improves Perceptual category learning

James Williamson

May, 1999

Technical Report CAS/CNS-99-015

Boston University Center for Adaptive Systems and
Department of Cognitive and Neural Systems
677 Beacon Street
Boston, MA 02215

Favoring Generalists over Specialists: How Attentional Biasing Improves Perceptual Category Learning

James R. Williamson
Department of Cognitive and Neural Systems
Boston University
Boston, MA 02215
jrw@cns.bu.edu

Abstract

A model of cortical learning is proposed, which incorporates supervised feedback using two forms of attention: (i) feature-specific attention which allows the network to learn associations between specific feature conjunctions (or categories) and outputs, and (ii) nonspecific attentional "vigilance" which biases this learning when the associations appear to be incorrect. Attentional vigilance improves learning if it favors, via learned modulatory weights, *generalist* categories over *specialist* categories. A biologically plausible neural network is proposed which implements these computational principles and which outperforms several alternative classifiers on classification benchmarks.

1 Introduction

Standard neural network learning models such as radial basis function networks (RBFs) use gradient descent-based learning algorithms in which error signals computed in the output layer are used directly to adjust weights governing the basis functions in the hidden layer. These networks are biologically implausible because they require a massive number of feedback connections to adjust individual weights in the hidden layer using complicated computations. From the standpoint of biological plausibility, a more promising approach is that of mixture modeling, in which signals from the output layer are treated merely as additional inputs activating nodes in the hidden layer. The network can thereby learn, using local, correlational learning rules, a model of the joint input/output density. Output predictions are generated by obtaining the marginal output distribution conditioned on the input.

A problem with this approach is its strictly unsupervised nature, in which the aim is to learn the most likely model of the input/output density rather than the model most likely to avoid errors in generating input→output predictions. Biasing the learning rates when errors are generated during training has been shown to improve the performance of an adaptive resonance theory (ART) network which learns via an on-line mixture modeling approach [1]. The network's hidden layer receptive

fields (or basis functions) are biased by raising a "vigilance" threshold, the effect of which varies inversely with the width of each receptive field. This temporarily makes the receptive fields narrower and thereby alters the activity patterns, and hence learning rates, in the hidden layer.

In this paper, the functional requirements for attentional biasing are clarified by demonstrating that receptive fields do not need to be made narrower in an absolute sense, and that a vigilance threshold is not required. Rather, attention merely needs to favor, with a modulatory bias, less selective *generalist* nodes over more selective *specialist* nodes. A modulatory bias is more subtle and flexible than a thresholding bias, avoiding the danger of a network becoming completely silenced when the threshold is raised too high. Recent neurophysiological experiments also suggest that attention plays a modulatory role [2].

The attentionally biased learning approach outlined above is quite general, and conceivably can be implemented in a variety of ways. We propose one such implementation called the Attentionally Biased Learning (ABL) network (Figure 1). The ABL network self-organizes internal categories, whose smooth receptive fields are defined by locally weighted connections from smooth activity distributions in lower layers. Thus, the ABL network uses simple, biologically plausible basis functions, as opposed to the explicit gaussian distributions typically used by RBF and mixture modeling networks [3],[1]. Section 2 describes the ABL network in detail. Section 3 illustrates ABL's performance on several classification benchmarks.

2 Attentionally biased learning network

2.1 Bottom-up activation

Input to the network consists of topographic, one-dimensional feature maps, $\boldsymbol{f}_i$, each containing L nodes (see Figure 1).[1] Because the maps are topographic (and hence have locally correlated activities) the activity envelope over each map tends to have a smooth, unimodal shape, such as a gaussian distribution. Varying an input value along the relevant perceptual dimension corresponds to translating the activity distribution across the feature map. A set of orientational columns in primary visual cortex is an example of a feature map *in vivo*. Presenting an oriented bar to the visual system produces a lump of activity across the orientational columns. As the bar is rotated, this pattern translates.

Unidimensional (1-D) basis nodes are activated by the match (or inner product) between the activity distribution in a feature map, $\boldsymbol{f}_i$, and the distribution of the node's weights, $\boldsymbol{w}_{ji}$:

$$x_{ji} = \frac{\sum_{h=1}^{L} f_{ih} w_{jih}}{1 + \rho\, T_{ji}}. \tag{1}$$

This computation yields a smooth receptive field over a perceptual dimension, as in Figure 2. Aliasing is avoided by requiring activity distributions in the feature map to be sufficiently wide with respect to the number of nodes in the map. The denominator in equation (1) shows that raising vigilance above its default value of $\rho = 0$ produces divisive inhibition. Raising vigilance has a different effect on each node, due to differing inhibitory weights T_{ji}, as is illustrated in Figure 2.

Category nodes represent feature conjunctions across M perceptual dimensions. They are initially activated only by a conjunction of bottom-up input from their M

[1] Extensions to two-dimensional feature maps are straightforward.

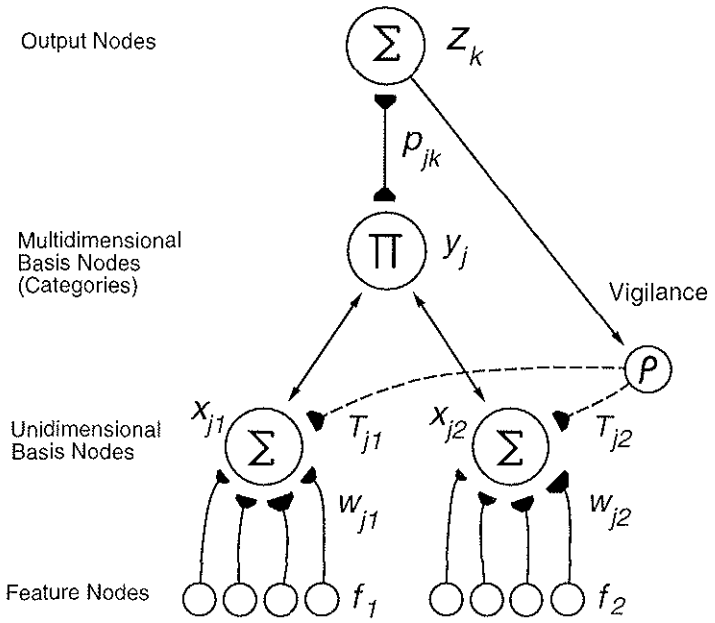

Figure 1: ABL network, shown with two feature dimensions, one category node, and one output node. Dashed lines indicate inhibitory connections. Symbols indicate mathematical operation computed at each node.

1-D basis nodes:

$$y_j = \prod_{i=1}^{M} x_{ji}. \tag{2}$$

The network's output nodes are then activated by the category nodes, via weighted connections p_{jk} which represent the probability of each output given the category.

$$z_k = \sum_{j=1}^{N} y_j p_{jk}. \tag{3}$$

The class prediction, K, is the index of the maximally activated output node:

$$K = \arg\max_k(z_k). \tag{4}$$

2.2 Supervision and attention

Let K^* denote the index of the "correct" supervised output class. An output criterion (OC) determines whether the network's output prediction is similar enough to the supervised output to allow learning. If the OC is not met, "attention" is invoked: the vigilance level, ρ, is incrementally raised from an initial value of $\rho = 0$. This causes the predictions to change due to differential modulations, via equation (1), of activities in the 1-D basis nodes. Vigilance is raised until either the OC is satisfied or until the maximal vigilance level is reached:

$$\text{If } z_K/z_{K*} < OC \text{ then raise } \rho \text{ until either } z_K/z_{K*} \geq OC \text{ or } \rho = \rho_{max}. \tag{5}$$

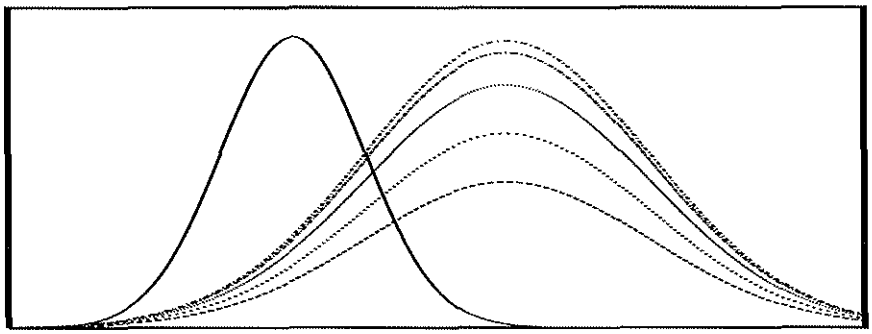

Figure 2: Differential effect of raising vigilance on two receptive fields. Node 1 is a *specialist*, with a narrow receptive field (left), whereas Node 2 is a *generalist* with a wide but low receptive field (right, bottom curve). Due to this difference, Node 1 is usually more active when it learns than Node 2 is when it learns. T_1 is thus larger than T_2 due to equation (12). Therefore, raising vigilance has a differential effect on the two nodes, attenuating Node 1's receptive field much more quickly than Node 2's receptive field. For plotting purposes, the receptive field heights are rescaled for each vigilance level so that the differential modulation is revealed in Node 2's receptive field only. Thus, the curves on the right (from bottom to top) show that, in a relative sense, Node 2's receptive field gets bigger with successively larger vigilance levels of $\rho = 0, 1, 4, 16, 64$. This shows that raising vigilance favors the generalist over the specialist.

Once equation (5) is satisfied, top-down feedback incorporates information as to the correct output. The resulting category activations represent the posterior probability of each category given both the input *and* the correct output. Supervised feedback to the output nodes turns the correct node on and the incorrect nodes off:

$$z_k = 1 \text{ if } k = K^*;\ z_k = 0 \text{ otherwise.} \tag{6}$$

Top-down output→category feedback factors in the conditional probabilities of the correct output (via the term $\sum_k z_k p_{jk}$). In addition, normalization causes each input sample to have the same net impact during learning, since learning rates depend on y_j.

$$y_j = \frac{\prod_{i=1}^{M} x_{ji} \cdot \sum_k z_k p_{jk}}{\sum_{j'=1}^{N} \prod_{i=1}^{M} x_{j'i} \cdot \sum_k z_k p_{j'k}}. \tag{7}$$

The output→category feedback is a feature-specific form of attention which serves a different role than the non-specific vigilance form of attention. Feature-specific attention favors categories that have learned associations with external expectations. Vigilance-based attention, on the other hand, performs a memory search by increasingly favoring categories that have wider receptive fields as vigilance is raised.

2.3 Learning

Since category activities represent posterior probabilities conditioned on the current input/output, correlational learning rules allow the network to learn a mixture model of the input/output density. The learning procedure is, in effect, an on-line approximation of a statistical batch learning approach for optimizing mixture models, the expectation-maximization (EM) algorithm [1].

On-line learning obtains better statistical sampling if the learning rate begins high and then reduces with experience. Experience is represented by n_j, which begins at zero and converges toward 1 via:

$$\Delta n_j = \alpha y_j (1 - n_j). \tag{8}$$

Each node's learning rate is represented by α_j, which begins with a relatively large value but converges toward α as $n_j \rightarrow 1$.

$$\alpha_j = \frac{\alpha}{\alpha\beta + n_j}, \tag{9}$$

The feature weights w_{jih} track their inputs. Over time, each weight vector essentially makes copies of the spatially varying activity distributions that input to it, and thereby ends up encoding a wider distribution than exists in any single activity distribution.

$$\Delta w_{jih} = \alpha_j y_j (f_{ih} - w_{jih}). \tag{10}$$

The output weights p_{jk} track the output activities, and thereby learn the conditional probability that each output is correct given the category activation.

$$\Delta p_{jk} = \alpha_j y_j (z_k - p_{jk}). \tag{11}$$

The inhibitory weights T_{ji} track the activations of their 1-D basis nodes:

$$\Delta T_{ji} = \gamma y_j (x_{ji} - T_{ji}). \tag{12}$$

Weight pruning is also performed to speed up processing. If w_{jih} or p_{jk} fall below the pruning threshold, Γ, they are set to zero permanently.

2.4 Category instantiation

Various heuristics are possible for determining when to instantiate new categories. In our simulations the following procedure is used: training always begins with zero categories, and a new category is instantiated every time vigilance reaches its maximal level, $\rho = \rho_{max}$. As a result, the number of categories that is created depends on the difficulty of the classification task. New categories are initialized in a "tabula rasa" state, with uniformly distributed w_{jih} and p_{jk} weights, and with $n_j = T_j = 0$. Immediately following its instantiation, only the new category is allowed to learn the current input sample.

3 Simulations

3.1 Methods

The same set of parameters is used on all the benchmarks: $OC = 0.8$, $\rho_{max} = 100$, $\alpha = 10^{-7}$, $\beta = 4$, $\gamma = 0.01$, $\Gamma = 0.005$. Average results are obtained from 25 independently trained networks on the first two benchmarks, and from 5 networks on the third, larger benchmark. Each network is trained for 30 epochs (or iterations through the data), with randomized ordering in each epoch. After training, the networks are tested on separate test data, the results of which are shown in Tables 1–4. Due to the large number of runs, these results are highly reliable.

Preprocessing is required to format the input data appropriately for the ABL network. Scalar input values are converted into activity distributions in feature maps.

Table 1: Classification benchmark comparisons. Percents correct are shown, with standard deviation (if available). See text for details.

Spoken Vowel Classification

ABL	KNN	GAM	EM	FAM	MLP
59.1 ± 2.3	56.3	56.1	54.6	51.1	50.6

Waveform Classification (300 training samples)

	ABL	GAM	CART	KNN
Clean:	84.2 ± 0.6	82.4	72	78
Noisy:	79.9 ± 0.6	77.5	72	38

Waveform Classification (2,500 training samples)

	ABL	GAM	DA-RBF	TR-RBF	MD-RBF	G-RBF
Clean:	86.7 ± 0.2	85.1	84	83	81	84
Noisy:	86.6 ± 0.2	85.6	87	83.2	82	80

Texture Classification

	ABL	GAM
6 textures	95.0 ± 0.3	93.7
12 textures	95.7 ± 0.2	94.5
18 textures	95.3 ± 0.1	94.8

In these spatial codes, the input value is represented by the location in the feature map of the activity distribution.[2] The magnitude of the i^{th} input value, I_i, is converted into a Gaussianly distributed pattern of activity in the i^{th} feature map:

$$f_{ih} = \frac{\exp\left[-\frac{(I_i - h/L)^2}{2(\lambda\sigma_i)^2}\right]}{\sum_{h'=1}^{L} \exp\left[-\frac{(I_i - h'/L)^2}{2(\lambda\sigma_i)^2}\right]}. \tag{13}$$

The input vectors are first normalized (across the entire data set) to a range of [0:1] in each dimension. After this normalization, each σ_i value—the standard deviation in dimension i of the training set—is computed. These σ_i values cause the width of each lump of activity produced by equation (13) to be proportional to the standard deviation of the data in that dimension. The parameter λ controls common width scaling. The first two benchmarks contain relatively sparse data, so wider activity distributions work best: $\lambda = 0.75, L = 7$. The third benchmark is less sparse, so narrower activity distributions work best: $\lambda = 0.375, L = 10$.[3]

3.2 Results

The ABL network is tested on three classification benchmarks, spoken vowel classification [4], waveform classification [5], and natural texture classification [7]. The spoken vowel benchmark consists of 528 training samples and 462 test samples, with 10-dimensional data obtained from real processed speech, and 11 English vowel output classes. The waveform benchmark consists of 21 features obtained via a

[2]Spatial codes provide a richer representation than scalar values by also allowing variance to be explicitly represented by the width of the activity distribution.

[3]Ideally, the appropriate widths of the activity distributions would be automatically determined by self-organizing feature maps.

convex combination of basis functions with random perturbations, and three output classes. The noisy variation contains an additional 19 pure noise dimensions. This benchmark has been evaluated with either 300 or 2,500 training samples. The natural texture benchmark contains 17 feature dimensions consisting of orientational contrast at four orientations and four spatial scales, plus a single brightness feature. For each texture class, there are 768 training samples, each derived from a local region of a texture image.

Table 1 illustrates ABL's results on all three benchmarks, alongside those of several other classifiers. Results on the spoken vowel benchmark are obtained from [1],[4]. The classifiers are: k-nearest-neighbors (KNN); Gaussian ARTMAP (GAM), a predecessor of ABL which uses explicitly defined Gaussian receptive fields and a vigilance threshold; expectation-maximization (EM) which learns an unbiased mixture model of the input/output density using a batch learning procedure; Fuzzy ARTMAP (FAM), an older ART network which uses non-smooth receptive fields, a vigilance threshold, and winner-take-all learning; and multilayer perceptrons (MLP). ABL outperforms all of these alternatives. Results on the waveform benchmark are obtained from [5],[6]. The classifiers are: CART, a decision tree; three standard RBF variations, and a sophisticated dynamic annealing RBF network (DA-RBF). ABL outperforms all of these except for DA-RBF on the large, noisy variation. Results on the natural texture benchmark are obtained from [7]. ABL outperforms GAM on all three texture sets. GAM in turn has outperformed several alternative classifiers on a slightly different texture database [7].

Conclusions. A clear computational function of nonspecific vigilant attention is proposed, in which attentional modulation favors generalist categories over specialist categories. A neural network implementation demonstrates the effectiveness of this type of vigilant attention. The network performs well on several classification benchmarks despite the network's biologically motivated constraints of simple computations, on-line learning, and limited forms of supervised feedback.

Acknowledgments

Supported in part by the Defense Advanced Research Projects Agency and the Office of Naval Research (ONR N00014-95-1-0409).

References

[1] Williamson, J.R. (1997). A constructive, incremental-learning network for mixture modeling and classification. *Neural Computation*, **9**, 1517-1543.

[2] McAdams, C.J. and Maunsell, J.H. (1999). Effects of attention on orientation-tuning functions of single neurons in macaque cortical area V4. *Journal of Neuroscience*, **19**, 431-441.

[3] Poggio, T. and Girosi, F. (1989). A theory of networks for approximation and learning. *A.I. Memo No. 1140*, M.I.T., 1989.

[4] Deterding, D.H. (1989). *Speaker normalisation for automatic speech recognition.* Ph.D. thesis, University of Cambridge.

[5] Breiman,L., Friedman,J.H., Olshen,R.A., and Stone,C.J. (1984). Classification and Regression Trees. Wadsworth International Group: Belmont, California.

[6] Miller, D., Rao A.V., Rose K., and Gersho, A. (1996). A global optimization technique for statistical classifier design. *IEEE Transactions on Signal Processing*, **44**, 3108-3122.

[7] Grossberg, S. and Williamson, J.R. (1999). A self-organizing neural system for learning to recognize textured scenes. *Vision Research*, **39**, 1385-1406.