JPEX: A Psychologically Plausible Joint Probability EXtractor

Sébastien Hélie (Helie.Sebastien@courrier.uqam.ca)
Department of Computer Science, C.P. 8888, Succ. Centre-Ville
Montréal, Qc H3C 3P8 CANADA

Robert Proulx (Proulx.Robert@uqam.ca)
Department of Psychology, C.P. 8888, Succ. Centre-Ville
Montréal, Qc H3C 3P8 CANADA

Bernard Lefebvre (Lefebvre.Bernard@uqam.ca)
Department of Computer Science, C.P. 8888, Succ. Centre-Ville
Montréal, Qc H3C 3P8 CANADA

Abstract

Extracting redundancies in the data is the main purpose of unsupervised learning and estimating the covariance using Hebbian learning is a widespread way to achieve this. However, Hebbian learning only leads to the extraction of between-unit covariance. Because most associative memories use distributed representations, it would be more useful to extract the covariance of states. Yet, this operation would still be insufficient to fully model more complex environments, which include higher-order relations. In the present paper, we propose a new architecture, JPEX, which extracts higher-order joint probabilities at the *state* level using the tensor product as a learning rule. This new learning rule is compared with simple Hebbian learning in an environment which includes second-order relations. Also, JPEX's ability to learn non-linear relationships is illustrated by training the model on the XOR categorization problem.

Introduction

The main goal of unsupervised learning is density estimation, which consists of estimating the joint probability of the input patterns (Hastie, Tibshirani, & Friedman, 2001). While this estimation can be achieved in most application domains, it is only useful to the extent that redundancy is present in the data (Hertz, Krogh, & Palmer, 1991): redundancy *is* knowledge (Barlow, 1989). Among the many statistics that explicitly quantify redundancy, the covariance is the most widely used (Hastie et al., 2001; Hertz et al., 1991). One reason which explains the popularity of this statistic is that covariance, which is the first centered joint statistical moment, is well understood and sufficient to completely define first order relations between the patterns.

The most well known unsupervised method to extract the covariance is Hebbian learning (Kohonen, 1972). Basically, Hebbian learning postulates that the connections between co-activated neurons are stronger (Proulx & Helie, 2005). This learning rule is mathematically defined by the summation of covariance matrices, which can be performed locally either online (Anderson et al., 1977) or offline (Hopfield, 1982). An associative memory trained with this learning procedure is tolerant to noise, can perform pattern completion, and its estimation of the covariance matrix is optimal according to the least-square criterion (Proulx & Hélie, 2005).

However, Hebbian learning has several limitations. First, each pattern is stored as a distinctive trace in the associative memory. Therefore, the introduction of noise during training increases the memory load, which is limited by the number of available units in the network. Second, the extracted covariance matrix describes the network's units, an information which is most useful when local representations are used. However, the robustness of associative memories is a result of the postulate that units are best described at the sub-symbolic level (Smolensky, 1988) and that the semantic (symbols) is represented at the level of *states*. Therefore, extracting the covariance of the network's states is more relevant. Finally, the information contained in covariance matrices might not be sufficient: higher-order relations between the states are also useful. For instance, state *A* might appear with state *B*, or with state *C*, but never with the conjunction of states *B* and *C*.

While this last point might seem farfetched, the need for models that can learn higher order relations in cognitive science is ubiquitous. For instance, it was argued for many years that non-linearly separable problems must be solvable by cognitive models in order to explain the human ability to understand simple rules such as: "In a restaurant, your appetizer may be a salad or a soup but not both" (the XOR problem: Minsky & Papert, 1969). Another human ability which has puzzled philosophers (Hume, 1888) and computer scientists (Pearl, 2000) for many years is causal abstraction. While this type of reasoning is taken for granted in everyday life, the inference of indirect causal factors (e.g., causal chains) requires sensitivity to higher order statistical relations. Accordingly, the most widespread approach to infer causal DAGs (Directional Acyclic Graphs) is constraint-based (e.g., Spirtes, Glymour & Scheines, 1993). This type of graph explicitly represents the independences in the joint frequency distribution of the data in order to simplify the transmission of uncertainty. In constraint-based approaches, the DAG is constructed by testing all order conditional independences. While higher order conditional independences can be tested using tetrad differences in covariance submatrices (Bollen & Ting,

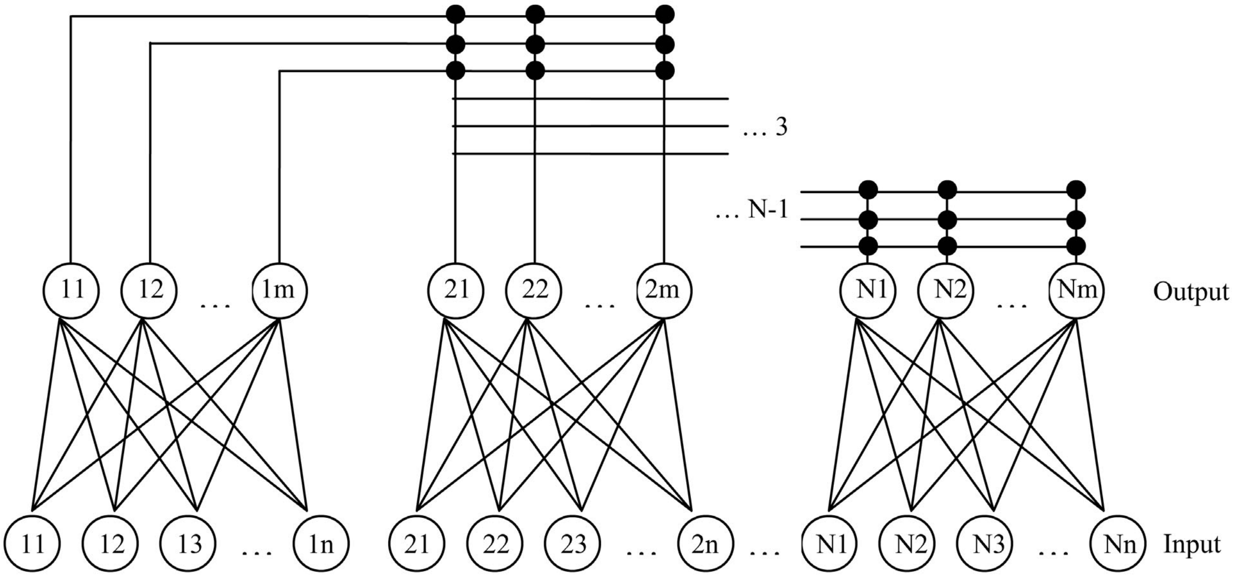

Figure 1: Architecture of JPEX.

1993), the psychological plausibility of such a process suffers from the use of qualitatively different tests of independence depending on the number of conditioned variables; however the direct extraction of higher order statistical relations (e.g., higher order joint probabilities) would result in tests of independence that are qualitatively identical at all levels, notwithstanding the number of conditioned variables. While nonlinear multilayer neural networks are able to extract these higher order statistics, no psychologically plausible linear way of learning these relationships is available[1].

In the present, we propose a psychologically plausible way of learning higher order statistical relations: the Joint Probability EXtractor (JPEX), which enables an online hybrid hard competitive (Rumelhart & Zipser, 1986) / Bidirectional Associative Memory (BAM: Kosko, 1988) neural network to learn in noisy environments and provide maximum likelihood estimations of the network(s)' states' joint probabilities. In the remaining sections, JPEX is described mathematically and compared with standard (Hebbian) covariance models.

JPEX

JPEX is an online joint probability extractor implemented using unsupervised neural networks augmented with a novelty detector of the vigilance type (Grossberg, 1976). Its main features are the following: First, the joint probabilities are extracted at the state level (instead of the usual unit level). Second, JPEX is an anytime algorithm: the model can be stopped at any moment and the maximum-likelihood estimations of the joint probabilities, given the inputs, are outputted. As for all artificial neural networks, JPEX can be entirely described by its architecture, transmission rules, and learning rules.

Architecture

The architecture of JPEX is shown in Figure 1. As seen, JPEX is composed of N receptive fields each composed of n units. There are no direct connections between the receptive fields. Each receptive field has feedforward connections to its own output layer, which is composed of m units (the number of output units varies during learning). Within each output layer, there are lateral inhibitive connections (not shown in Figure 1) which implement a hard competition process between the outputs (Rumelhart & Zipser, 1986). The output layers are connected in a serial manner to form a novel kind of BAM architecture (Kosko, 1988), in which each output unit of receptive field i is connected to every output units of receptive field $i + 1$.

Transmission Rules

Bottom-up Activation In JPEX, each receptive field receives activation that spreads to its output layer.

$$\mathbf{y}_{[i]} = \mathbf{W}_{[i]}\mathbf{x}_{[i]}$$

where $\mathbf{y}_{[i]}$ is the output vector of the i^{th} competitive network, $\mathbf{x}_{[i]}$ is the vector representing the state of the i^{th} receptive field, and $\mathbf{W}_{[i]}$ is the weight matrix of the i^{th} competitive network.

As in all hard competitive networks, the unit which is maximally activated is chosen as the winner. If the winner's activation is smaller than a predefined threshold, the input is not recognized as a member of a known category and a new output unit is recruited. The weight vector of the new output

[1] Linearity is an important caracteristic when it comes to understanding the behavior of the proposed model.

unit is defined as the input and this new output unit is the winner of the competition.

$$Max\left[\mathbf{y}_{[i]}\right] < \rho\left\|\mathbf{x}_{[i]}\right\|\left\|\mathbf{w}_{[i,k]}\right\|$$
$$\mathbf{w}_{[i,m+1]} = \mathbf{x}_{[i]}$$

where $0 \le \rho \le 1$is the vigilance parameter, $\mathbf{w}_{[i,k]}$ is the weight vector of the maximally activated output unit (k) of the i^{th} receptive field, ||•|| is the usual Euclidean norm, and $\mathbf{w}_{[\text{i, m+1}]}$ is the weight vector between the inputs and the new output unit.

After completion of this verification process, the activation of the winning node in each competitive network is set to unity while the remaining are shut down.

Top-down Activation In Figure 1, each filled circle is a bidirectional AND-gate that links three different receptive fields: $i - 1$, i, and $i + 1$. If the connected receptive fields are all receiving signals (e.g., there is a stimulus in each receptive field), the information stays at the associative level and the tensor learning rule is applied (see next section). Otherwise, if one of the receptive fields is not activated (e.g., no stimulus was presented), and the remaining two are, top-down signal is sent toward the inactive receptive field.

Prior to the mathematical formalization of the top-down activation rule, basic notions of tensor algebra must be introduced. First, tensor algebra is a generalization of matrix algebra. For instance, it is common to simplistically define matrices as two-dimensional arrays of numbers in introductory texts. Likewise, a tensor of rank N can be simplistically described as an N-dimensional table of numbers. Hence, a tensor of rank zero is a scalar, a tensor of rank one is a vector, and a tensor of rank two is a matrix. Second, the well-known inner product, used in matrix algebra, can be applied to tensors: the rank of a new tensor formed from the inner product of two other tensors is the sum of their individual rank minus two. While far from complete, this quick introduction to tensor algebra is sufficient to understand the inner working of JPEX (the interested reader is referred to: Kay, 1988).

Keeping these simple notions in mind, top-down activation through the associative tensor can now be formalized. When all but one receptive field are receiving activation, top-down activation is sent by iteratively using the standard inner product to reduce rank of the tensor until it reaches unity.

$$\mathbf{y}_k = (((\mathbf{V}\mathbf{y}_1)\mathbf{y}_2)\ldots)\mathbf{y}_{k-1}$$

where $\mathbf{y}_k$ is the output layer of the inactive receptive field and $\mathbf{V}$ is the associative tensor. The resulting $\mathbf{y}_k$ is the maximum likelihood estimation (MLE) of the output layer of the kth receptive field, and this vector can be sent downward to the input layer through the competitive weights:

$$\mathbf{x}_{[k]} = \mathbf{W}_{[k]}\mathbf{y}_{[k]}$$

where $\mathbf{x}_{[k]}$ is the input layer of the previously inactivated receptive field, of which the activation now represents the position of the centroid of the MLE category $\mathbf{y}_k$.

Learning Rules

Learning takes place at two levels in JPEX: the competitive level (**W**) and the associative level (**V**). All connections weights are initialized with zeros and the weights are updated after each iteration, whether there was a new output unit recruited or not. Learning at the competitive level is described by (Hertz et al., 1991):

$$\mathbf{w}_{[i,k,t+1]} = \mathbf{w}_{[i,k,t]} + \eta(\mathbf{x}_i - \mathbf{w}_{[i,k,t]})$$

where $0 \le \eta \le 1$ is a general learning parameter and $\mathbf{w}_{[i,\,k,\,t]}$ is the weight vector of the winning unit (k) of the i^{th} receptive field at time t. Because the networks are hard competitive learners, only the winning unit's weight-vector in each network is updated.

Most of JPEX's new properties result from learning at the associative level (BAM). At this level, the tensor product representation, proposed by Smolensky and Legendre (2006), is used to integrate the activation of the output units of every competitive network:

$$\mathbf{V}_{[t+1]} = \mathbf{V}_{[t]} + \bigotimes_{i=1}^{N}\mathbf{y}_i$$

where $\mathbf{V}_{[\text{t}]}$ is the weight tensor at time t, $\mathbf{y}_i$ is the output vector of the i^{th} competitive network and $\otimes$ is the usual tensor product defined as:

$$\bigotimes_{i=1}^{N}\mathbf{y}_\mathbf{i} = \begin{cases}\mathbf{y}_1\mathbf{y}_2^\mathbf{T}\mathbf{y}_3\mathbf{y}_4^\mathbf{T}\ldots\mathbf{y}_\mathbf{N}, \text{if } N \text{ is odd}\\ \mathbf{y}_1\mathbf{y}_2^\mathbf{T}\mathbf{y}_3\mathbf{y}_4^\mathbf{T}\ldots\mathbf{y}_\mathbf{N}^\mathbf{T}, \text{else}\end{cases}$$

The result of this learning rule is a tensor of rank N which either consists of a contingency table (N = 2), cuboid (N = 3) or hypercuboid (N > 3). Each coordinate represents a joint probability of order N – 1. Lower order joint probabilities are obtained by collapsing the (hyper)cuboid using summations. Because the resulting counts are following a Poisson distribution (viz. the probability of each non-negative integer is positive), which is a member of the exponential family, they are sufficient statistics to estimate the underlying probability distribution (Agresti, 1990), as shown in the following simulation.

Bottom-up Learning of Dependencies Between Handwritten Characters

Stimuli

The stimuli used are shown in Figure 2. Each stimulus was a handwritten digit coded as a 64-unit bipolar vector. During each iteration, three stimuli were randomly presented simultaneously in three distinct receptive fields. The dependencies between the categories in each receptive field were chosen in order to include both first and second order relationships (Table 1): the impossibility of having a “1” in the first and second receptive fields and a “7” in the third is a second order relationship while the impossibility of having a “7” in the second and a “1” in the third is a first order relationship (because the value of the first receptive field is not considered). The models’ ability to reconstruct this dependency table was used to assess the models’ performance.

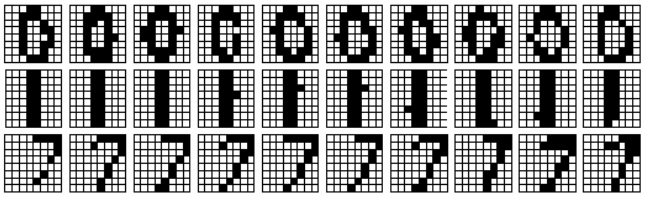

Figure 2: Stimuli used in the simulation.

Models

In this simulation, JPEX’s performance was compared to a regular (Hebbian) covariance estimator. This second model was identical to JPEX, except for the learning rule at the associative level, which was a standard Hebbian function (Kohonen, 1972):

$$\mathbf{V}^{Hebb}_{[t+1]} = \mathbf{V}^{Hebb}_{[t]} + \mathbf{y}\mathbf{y}^{T}$$

where $\mathbf{V}^{Hebb}_{[t]}$ is the estimated covariance matrix at time t and $\mathbf{y}$ is a vector resulting from the concatenation of the output vectors of every competitive networks. In both models, N = 3, n = 64, η = 0.2 and ρ = 0.55. This parameter setting is not optimal, but the assigned values are not responsible for the networks’ success / failure: the values given to ρ and η only affect the number of developed categories and the learning rate respectively. Each network was trained for 2 000 iterations and, because all the receptive fields were filled at each trial, no top-down activation was used in this simulation.

Results

All competitive networks in both models were perfectly able to recall the categories. The learned categories are shown in Figure 3. As seen, the competitive layer’s learning rule maximizes the overlap between the members of a given category and their corresponding weight vector (Rumelhart & Zipser, 1986). Thus, the categories are easily recognizable and, by the end of training, m = 3.

Table 1: Dependencies between receptive fields and categories

	RF2	0		
	RF3	0	1	7
RF1	0	0.00	0.02	0.04
	1	0.04	0.08	0.04
	7	0.05	0.10	0.05
	RF2	1		
	RF3	0	1	7
RF1	0	0.00	0.07	0.04
	1	0.06	0.03	0.00
	7	0.04	0.02	0.00
	RF2	7		
	RF3	0	1	7
RF1	0	0.11	0.00	0.07
	1	0.05	0.00	0.03
	7	0.04	0.00	0.02

Note. RF*n* is the n^{th} receptive field.

Of higher interest is the tensor developed at the associative level by JPEX (shown in Table 2). As seen, all the dependencies have been accurately estimated by JPEX. The joint distribution estimated by JPEX (Table 2) does not differ from the objective distribution (Table 1) of the stimuli ($G^2(8)$ = 3.46, p = .90). In particular, structural impossibilities were estimated with a frequency of zero and the largest estimation error was only 0.02.

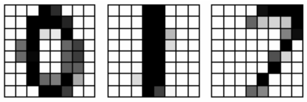

Figure 3: Competitive networks’ weight vectors after training.

In contrast, the covariance matrix developed using Hebbian learning left many independencies unnoticed. For instance, JPEX learned that it is impossible to have a “1” in the first and second receptive fields while having a “7” in the third. This is an example of second order relationship which is encoded in higher-order joint distributions but is invisible to first order statistics. On the other hand, the only information learned by the Hebbian model is that the probability of having a “1” in the first receptive field and a “7” in the third is 0.22. Likewise, the probability of having a “1” in the second receptive field and a “7” in the third is 0.15. The only impossible relation correctly inferred using Hebbian learning was that it is impossible to have a “7” in the second receptive field and a “1” in the third. This relation was learned because it is independent from the category present in the first receptive field, as shown by the column of zeros in Table 1.

Table 2: Joint frequency distribution estimated by JPEX

	RF2	0		
	RF3	0	1	7
RF1	0	0 (0.00)	45 (0.02)	66 (0.03)
	1	82 (0.04)	172 (0.09)	86 (0.04)
	7	116 (0.06)	216 (0.10)	103 (0.05)

	RF2	1		
	RF3	0	1	7
RF1	0	0 (0.00)	155 (0.08)	76 (0.04)
	1	104 (0.05)	60 (0.03)	0 (0.00)
	7	72 (0.04)	49 (0.02)	0 (0.00)

	RF2	7		
	RF3	0	1	7
RF1	0	212 (0.11)	0 (0.00)	98 (0.05)
	1	108 (0.05)	0 (0.00)	48 (0.02)
	7	91 (0.04)	0 (0.00)	41 (0.02)

Note. RF*n* is the *n*th receptive field. Numbers in parenthesis are proportions.

Interactive Hetero-associative Learning: The XOR problem

In the Introduction, it was argued that higher order relationships are needed to solve categorization problems that are not linearly separable. Hence, JPEX should be able to learn the classical XOR problem. This ability is shown in the present section.

Stimuli

The stimuli used were the handwritten “0”s and “1”s shown in Figure 2. The associations to be learned are shown in Figure 4. In the training phase, one line from Figure 4 was randomly chosen at each trial as the *state of the environment*. Each column was presented to a different receptive field ($N = 3$). Hence, if the second line of Figure 4 was chosen as the state of the environment, the first and third receptive fields received stimuli randomly chosen from the first line of Figure 2, while the second receptive field received a stimulus randomly chosen from the second line of Figure 2.

In the test phase, only the first two columns of Figure 4 were presented to the first two receptive fields of JPEX. The network had to reconstruct the correct stimulus in the third receptive field using top-down activation (the one in the third column of Figure 4).

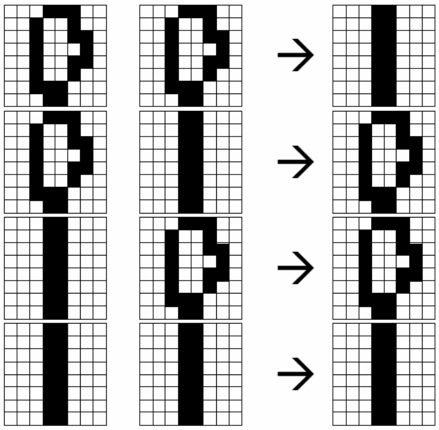

Figure 4: Associations learned in the XOR problem.

Model and Result

This simulation clearly illustrates the two types of transmission included in JPEX. In the training phase, all the receptive fields were filled with stimuli and only bottom-up learning occurred to form the associative tensor (as in the first simulation). In the test phase, one receptive field was always left empty (the third), and its most probable input was reconstructed using top-down processing: the first receptive field identified the first stimulus (bottom-up) and its output vector was transmitted through the associative tensor (top-down). The rank of the resulting associative tensor was $3 + 1 - 2 = 2$. The second receptive field identified the second stimulus (bottom-up) and its output layer was transmitted through the associative matrix resulting from the previous step (top-down). The resulting associative tensor was of rank $2 + 1 - 2 = 1$ and the third receptive field was not activated, which enabled downward propagation. As a result, the associative vector was sent through the third competitive network’s weight matrix ($\mathbf{W}_3$), and the ensuing activation of the input units forming the third receptive field reflected the position of the centroid of the most probable winning unit of this competitive network (given the associations learned in the training phase).

In the present simulation, the same values were given to the free parameters as in the preceding simulation. After only 100 training trials, all the associations were perfectly learned and the XOR problem was solved.

Conclusion

In the present paper, a new architecture was proposed to extract information about the states of several receptive fields. JPEX is a Joint Probability EXtractor which extracts higher-order conjunctive information about the environment by using the tensor product (Smolensky & Legendre, 2006) as a learning rule. This information is stored in a tensor which represents a contingency hypercuboid.

In the first simulation, limitations related to the strict extraction of lower-order statistics (e.g., the covariance) were brought forward by simulating a simple environment composed of three receptive fields. Second-order dependencies where not detected by Hebbian learning, while accurate density estimation of all conjunctive orders was achieved by JPEX.

It is interesting to note that tensor product learning is a generalization of Hebbian learning (Kohonen, 1972). Hence, JPEX inherits Hebbian learning's psychological plausibility and computational properties (pattern completion, robustness, optimality, locality, absence of an external teacher). In addition, the use of an associative tensor, instead of an associative matrix, adds structure to the memory, which allows the network to rapidly achieve non-linearly separable tasks, such as solving the XOR categorization problem. In Smolensky and Legendre's terms (2006), JPEX builds a structured representation of the conjunction of every training trial, which is sufficient to enable symbolic processing. Also, JPEX's top-down transmission rule is interpreted as "tensor contraction" according to their theory. Finally, even though higher-order relations are stored, lower-order statistics are still easily inferred by collapsing the hypercuboid using summations.

While JPEX's advantage over Hebbian models comes at the cost of higher complexity, $O(m^N)$ vs. $O[(m \times n)^2]$ respectively, a tensor representation can be contracted in several ways (Smolensky & Legendre, 2006). While the "exactness" of the representation is not preserved, graceful saturation has been found by these authors and complex phenomena have been modelled using tensor representations such has human memory and language processing (included in Smolensky & Legendre, 2006). However, further research is needed to assess the performance of a JPEX model containing N receptive fields whose joint frequencies are encoded using a tensor memory of rank k ($k < N$).

Also, the examples presented in this paper used dependencies between several spatially aligned receptive fields: JPEX can also be used to detect higher-order *temporal* dependencies. For instance, the N^{th} receptive field can reflect the actual state of the environment (time t), whereas the $(N\text{-}1)^{th}$ field can be fed with the state of the environment at time t-1, etc. Using enough receptive fields might enable JPEX to correctly learn chaotic series or solve the one-to-many problem in temporal learning. However, further research is needed to assess the limits and capabilities of higher-order joint probability estimation.

Acknowledgments

This research was part of the doctoral requirements of the first author and was supported by scholarships from *Le Fonds Québecois de la Recherche sur la Nature et les Technologies* and the *Natural Sciences and Engineering Research Council of Canada* granted to the first author.

References

Agresti, A. (1990). *Categorical Data Analysis*. New York: Wiley.

Anderson, J.A., Silverstein, J.W., Ritz, S.A. & Jones, R.S. (1977). Distinctive features, categorical perception, and probability learning: Some applications of a neural model. *Psychological Review*, *84*, 413-451.

Barlow, H.B. (1989). Unsupervised learning. *Neural Computation*, *1*, 295-311.

Bolen, K.A. & Ting, K.-F. (1993). Confirmatory tetrad analysis. *Sociological Methodology*, *23*, 147-175.

Grossberg, S. (1976). Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biological Cybernetics*, *23*, 121-134.

Hastie, T., Tibshirani, R. & Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

Hertz, J., Krogh, A. & Palmer, R.G. (1991). *Introduction to the Theory of Neural Computation*. Menlo Park, CA: Addison-Wesley Publishing Company.

Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*, 2554-2558.

Hume, D. (1888). *A Treatise of Human Nature*. Oxford: Clarendon Press.

Kay, D.C. (1988). *Schaum's Outline of Tensor Calculus*. New York: McGraw-Hill.

Kohonen, T. (1972). Correlation matrix memories. *IEEE Transactions on Computers C*, *21*, 353-359.

Kosko, B. (1988). Bidirectional associative memories. *IEEE Transactions on Systems, Man, and Cybernetics*, *18*, 49-60.

Minsky, M.L. & Papert, S. (1969). *Perceptron: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. New York: Cambridge University Press.

Proulx, R. & Hélie, S. (2005). Adaptive categorization and neural networks. In C. Lefebvre & H. Cohen (Eds.) *Handbook of Categorization in Cognitive Science* (pp. 793-815). Oxford: Elsevier.

Rumelhart, D.E. & Zipser, D. (1986). Feature discovery by competitive learning. In D.E. Rumelhart & J.L. McClelland (Eds.) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (pp. 151-193). Cambridge, MA: MIT Press.

Smolensky, P. (1988). On the proper treatment of connectionism. *Behavioral and Brain Sciences*, *11*, 1-74.

Smolensky, P. & Legendre, G. (2006). *The Harmonic Mind: From Neural Computation to Optimality-Theoric Grammar.* Cambridge, MA: MIT Press.

Spirtes, P., Glymour, C.N., & Scheines, R. (1993). *Causation, Prediction, and Search*. New York: Springer.