INFORMATION STORAGE CAPACITY OF CONNECTIONIST SYSTEMS: THE LINEAR ASSOCIATOR

Technical Report AIP - 25

Dean C. Mumme & Walter Schneider

Learning Research and Devlopment Center
University of Pittsburgh
Pittsburgh, PA 15260

29 September 1987

This research was supported by the Computer Sciences Division, Office of Naval Research and DARPA under Contract Number N00014-86-K-0678. Reproduction in whole or in part is permitted for purposes of the United States Government. Approved for public release; distribution unlimited.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; Distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AIP - 25	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Carnegie-Mellon University 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Computer Sciences Division Office of Naval Research (Code 1133)
6c. ADDRESS (City, State, and ZIP Code) Department of Psychology Pittsburgh, Pennsylvania 15213	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street Arlington, Virginia 22217-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Same as Monitoring Organization 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-86-K-0678
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS p400005ub201/7-4-86

PROGRAM ELEMENT NO	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO
N/A	N/A	N/A	N/A

11. TITLE (Include Security Classification)

Information Storage Capacity of Connectionist Systems: The Linear Associator

12. PERSONAL AUTHOR(S) D.C. Mumme and W. Schneider

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM 86Sept15 TO 91Sept14	87 September 29	19

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Artifical Intelligence, connectionism, Linear Assocator

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The information-storage capacity of hetero-associative memory systems is addressed. The associator can be treated as an M-ary symmetric channel when M associations are stored. The maximum number of associations storable is bounded asymptotically by N/2 where N is the number of connection weights. Storage efficiency is bounded by M/N so that it never exceeds 1/2. Information capacity degrades as inter-vector correlations increase and also when classification tasks are performed. The correlation effect is most pronounced in high-dimensional systems storing a large number of associations.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Alan L. Meyrowitz	22b. TELEPHONE (Include Area Code) (202) 696-4302 22c. OFFICE SYMBOL N00014

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

Information Storage Capacity of Connectionist Systems: The Linear Associator.[1,2]

Dean C. Mumme
and
Walter Schneider

Learning Research and Development Center
University of Pittsburgh

Abstract

Information theory is applied to determine the number of items storable in a linear associator. An ensemble of association matrices is treated as an M-ary symmetric information channel where M is the number associations stored via the outer-product rule. The entropy of the ensemble under the outer-product learning rule is derived and used to bound the number of ***usefully-storable*** items for the ensemble. In particular, if the ensemble has input dimensionality n_I and output dimensionality n_O, and M associations between vectors of $\pm 1's$ are stored, then the entropy of each weight is $1/2 \log_2 \frac{\pi e M}{2}$. Assuming independent weights gives the upper bound $1/2 \cdot n_I n_O \cdot \log_2 \frac{\pi e M}{2}$ for the entropy of the ensemble. The ***task*** of the ensemble as an M-ary symmetric channel is correct identification of which output-prototype corresponds to the prototype presented at the input. The corresponding task entropy or ***task load*** for M stored items is $M \cdot \log_2 M$ which leads to the upper bound

$$\frac{M}{n_I n_O} \leq \frac{1}{2} + \frac{1}{\log_2 M}$$

for the ratio of the number of associations storable to the number of weights in the system. Asymptotically, large matrices can store at most half as many associations as there are weights in the system. ***Storage efficiency*** is defined as the number of bits stored in the ensemble divided by the number of bits needed to specify the ensemble itself. The efficiency can be shown to be less than $\frac{M}{n_I n_O}$.

Performance degradation due to storage of correlated vectors is addressed. A performance merit parameter, d', is derived as a function of matrix size, number of items stored, and correlation between stored prototypes. This parameter is shown to decrease with the square-root of M if the vectors are uncorrelated, otherwise it decreases with M. This indicates a marked capacity decline in the correlated case and reveals quantitatively the sensitivity of large systems to prototype correlation. In order for correlation effects to be negligible, the probability p that a 1 occurs should be very nearly 1/2 as M gets large. A sufficient condition is that $|p - 1/2| < \frac{2}{\sqrt{3}} \cdot M^{-1/4}$. A sufficient condition for correlation effects

---

[1] Paper is based on a thesis by the first author for the doctoral degee in Computer Science at the University of Illinois at Urbana-Champaign.

[2] This research is funded by a grant from the Office of Naval Research.

to be prevalent is $|p - 1/2| > 2 \cdot \sqrt{3} \cdot M^{-1/4}$. This reflects the sensitivity of large systems vector correlation.

More generally, performance limits are derived by evaluating the task-entropy and using information theoretical relations between input, memory, and output random-variables. This has implications for memory-classification of input vectors. This task can be viewed as a retrieval on information-degraded inputs (e.g. retrieval on partial or noisy input vectors). Performance is limited by the amount of information the input vector provides about the correct input prototype. The amount of information provided by the input decreases as more classification "fan-in" is allowed. The amount of information the input provides and its relation to memory storage and classification can be derived analytically in certain interesting cases.

Numerical evaluation of derived relations and simulations are included to verify the theory. The intent of this investigation is to provide a basis for the eventual development of an information-theory of memory.

Information Storage and Classification in Connectionist Systems: The Linear Associator

[1]

Introduction

The systems under consideration are an outgrowth of work done on self-organizing automata and perceptrons [26, 30] and later work in parallel associative memories, e.g. [15, 31] Minsky and Papert in [26] had carried out rather extensive mathematical analysis on perceptrons revealing inherent limitations in the classes of problems they could solve. These systems were "learning" automata expected to classify input "stimuli" based on their past experience on "training" inputs. Minsky and Papert showed that multiple-stages of perceptrons were required for many problems of interest yet no training algorithm was known at the time for multi-level systems. They concluded in their book that the systems held little promise and subsequent investigation of perceptrons evaporated.

Eventually however, with more powerful computers to carry out simulations, and the development of several multi-level learning algorithms [32, 16, 27, 5] descendant offshoots of the perceptron have regained interest. Currently a variety of these automata exist and are known by names such as "Neural-nets", "Parallel Distributed Processors" (PDP networks), "Associative Memories". They are collectively called "Connectionist Architectures" and have been studied as self-organizing memories of perception [21] content-addressable memories, heirarchical knowledge bases, and classification systems [3, 2] models of human "neural-computation" [13, 3] of human task performance and attentional learning [37, 35] speech performance and natural language understanding [36, 33, 11]

These and other efforts have led to guarded optimism for the future of Connectionist architectures as knowledge engines or as models of human intelligence. Capabilities and limitations of both task learning and performance have been demonstrated. However, with the exception of a few mathematical investigations [21, 13, 14, 5, 12] these structures are understood primarily in a qualitative sense.

In this paper, we utilize concepts from information theory to study a simple matrix model of distributed memory. Its information-storage capacity and efficiency are evaluated allowing definition of a matrix's storage load factor. Memory performance in problems such as pattern completion can then be

[1]This resaerch supported by a grant from the Office of Naval Research

viewed as a function of matrix loading. Degradation of storage capacity with inter-stimulus correlation and noise at the input are also addressed.

This work is motivated by a simulation-model of human attentional learning developed by the authors [35]. Though these results are specifically intended for fuller understanding of the model, the apply to a much broader class of "Connectionist" systems.

"Neural-based" systems

Matrix models of parallel distributed memories were derived as a simplistic model of brain ce computation. In the model, the output of each cell is a real number, y representing the deviation of th cell's firing frequency from some reference frequency. As such, y can be negative as well as positive. Tl inputs $\{x_1, x_2, \ldots, x_n\}$ to the cell are similarly real valued and each input, x_i has an associated coupli strength w_i to the cell which determines the effectiveness of that input on the cell output. The c determines its output by taking the weighted average of the inputs,

$$y = \frac{1}{n}\sum_{i=1}^{n} w_i x_i$$

The matrix memory is constructed from a collection of these cells, each sampling the same set of inpu If n_I is the number of inputs to the memory and n_O is the number of cells in the memory, the vect $\mathbf{x} \equiv (x_1, x_2, \ldots, x_{nI})$ of inputs when presented to the input of the system produces an output vect $\mathbf{y} \equiv (y_1, y_2, \ldots, y_{nO})$ given by the relation:

$$\mathbf{y} = \frac{1}{n_I} A\mathbf{x}$$

where A is the matrix of coupling weights w_{ji} connecting the ith input to the jth cell [15, 21]

To store information in this system, two sets of vectors called the input prototypes $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_M\}$ a the output prototypes $\{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_M\}$ are used. For each input prototype $\mathbf{f}_m$, the weights of the system a adjusted so that the $\mathbf{g}_m$ vector results at the system output when $\mathbf{f}_m$ is presented at the input. T system is then said to associate $\mathbf{f}_m$ with $\mathbf{g}_m$. For each m=1,2,....,M, the matrix that is used to associa $\mathbf{f}_m$ with $\mathbf{g}_m$ (called the mth association) is the *outer-product* $\mathbf{g}_m \mathbf{f}_m^T$ [15, p. 18]. To store the associations, these M matrices are added to obtain:

$$A = \sum_{m=1}^{M} \mathbf{g}_m \mathbf{f}_m^{\mathrm{T}} \tag{1}$$

The information for each association is distributed over the whole of A and therefore is overlaid with the information for the other associations. The resulting interference between associations increases with M, and ultimately limits the number of associations storable in the system.

In the case that $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_M$ are mutually orthogonal, no interference exists. When $\mathbf{f}_k$ is input to the system, we have[2]

$$A\mathbf{f}_m = \frac{1}{n_I}\sum_{m=1}^{M} \mathbf{g}_m \mathbf{f}_m^T \mathbf{f}_k$$
$$= \frac{1}{n_I}\mathbf{g}_k \mathbf{f}_k^T \mathbf{f}_k$$
$$= \frac{1}{n_I}||\mathbf{f}_k||^2 \mathbf{g}_k, \qquad k = 1,2, \ldots, M.$$

The matrix produces a multiple of $\mathbf{g}_k$ when $\mathbf{f}_k$ is present at the input. If the $\mathbf{f}_k$ are chosen so that $||\mathbf{f}_k||^2 = n_I$ then $\mathbf{g}_k$ is reproduced exactly [3, p. 804]

The synopsis is concerned primarily with the case that $M > n_I$ so that the input vectors are linearly dependent and interference effects must be accounted for. In this case the output vector is only an approximation of the proper prototype output. Our concern is the number M of associations that can be stored in a matrix of a given size before the output becomes unrecognizable.

Characterizing Storage Capacity

To estimate the storage capacity of the matrix, we examine a system that has stored M associations $(\mathbf{f}_m, \mathbf{g}_m)$, m= 1,2,...,M for some M. The input-prototype vectors are n_I-dimensional and the output-prototypes are n_O-dimensional. Initially, the values allowed for the components are $\pm$ 1. All input-prototypes will then have $||\mathbf{f}_m||^2 = n_I$ and all output-prototypes $||\mathbf{g}_m||^2 = n_O$. Later we can generalize but this case is interesting in itself as these values represent saturation extremes of the inputs/outputs. A value of 1 represents a cell firing at its maximum rate and a value of -1 represents the minimum rate. Storing prototypes of this limited form corresponds to the cells each producing a "polarized" response to an input vector that itself is the result of a previous stage of saturated cells. The vectors are assumed to have an unbiased distribution of ± 1's as explained later.

To motivate the method of storage measurement, we make an analogy with digital memory. The address to the memory can be viewed as an input vector and the retrieved data as the output vector. A particular address vector and the data vector stored at the address location can be regarded as a vector-association pair. The number of bits represented by the data vector is the information the system provides upon performing the input-to-output association. For digital memory, the number of bits represented is the same as the number of bit-locations in the data vector and so is identical with the dimensionality of the data vector. **Storage** is defined as the amount of information per association multiplied by the number of associtions stored in memory. **Storage capacity** is the maximum storage the system can provide. In this case, the storage capacity is limited by the number of storage locations of

[2] The norm $||\,||$ refers to the euclidean norm.

the memory. Though the dimensionality of both the input and output vectors is specified in advance, the data items are not. That is, the number of items that can be stored is not determined by what they are.

For the matrix memory, the storage is likewise given by the information per association multiplied by the number of associations stored. The dimensionality of the input and output prototypes are specified in advance, but the prototypes themselves are not. For this reason, the storage of the memory is not defined for a particular matrix but rather for a *class* of matrices all of the same size. The class of outer-product matrix-associators of a given size is the set of all matrices that can be generated from vectors of ± 1's via equation (1). An association is not considered to be stored in a particular matrix of the class unless unless it is explicitly included in the sum, (1) that determines the matrix.

Unlike digital memory, the information per association can be characterized in two ways. The first is to present for arbitrary $k \in \{1,2,...,M\}$ the k^{th} input prototype to the system, and regard the matrix-output as a probabilistic rendition of the k^{th} output prototype. On the average, (over all matrices of the class) given M, the matrix-output will provide a certain amount of information about the prototype output and this is taken as the information provided by the association.

The second method is to consider the matrix as an information channel. The k^{th} input is presented to the system and an output is generated. The latter is compared with each prototype-output vector via a similarity measure and the best prototype match is chosen. This is called an **output decision**. If the i^{th} output prototype is chosen, an error is identified with $i \neq k$. The probability of error averaged over the matrix-class is taken as the error probability for the associator as an M-ary symmetric channel. The average mutual information between the output and input is thus defined. This average is considered as the information per association.

In either case, the storage is the product of M and the information represented by a single association. Initially, the storage of the matrix increases proportionally with M. The error probability increases with M as well so that the information per association gradually decreases. For some value M^* of M, the information per association begins to diminish more rapidly than M increases. At this point, storing more associations decreases total information storage of the system. The system has reached its storage capacity.

For the second case, we define for each matrix-size, N, the **matrix channel** *of size N on M associations*. It consists of the *ensemble* of all possible matrices with $n_I n_O = N$ that can be constructed from a set of M prototype-pairs $(\mathbf{f}_m, \mathbf{g}_m)$. Once a set of associations is chosen for storage, a particular matrix is selected from the ensemble via equation (1). This matrix is deterministic and therefore is not a channel in the usual sense. The storage for a particular matrix constructed from M associations is defined as the storage of the matrix channel from which it was selected.

The matrix-channel does not require that the system *reconstruct* the appropriate output response as does the first storage characterization. The matrix channel merely selects the best match from among the M prototype-outputs. Therefore one would expect the number of associations storable in the matrix-channel to be larger than in the first type of associator. The storage capacity of the matrix-channel identifies the maximum number of useable[3] associations that can be stored. Use of the channel for input-classification will require storage at some fraction of this maximal figure. Our objective is to quantify the maximal figure as a function of channel-size and use it to determine memory requirements for particular classification tasks. For this purpose, matrix-channel will considered in what follows.

Bounds on Storage Capacity

Assumptions and Notation

This analysis assumes important relative magnitudes among the parameters. We assume $n_i \geq 100$, $i=1,2$. The number of associations, M satisfies $n_i << M << 2^{n_i}$ $i=1,2$.[4] The upper bound in this case is assumed to exceed M by many orders of magnitude. This assures that sampling without replacement is virtually identical to sampling with replacement and simplifies the analysis. An optimal value M^* of M will be shown to exist that is less than the **net-size**, $n_I n_O$. Therefore, as long as the net-size is insignificant compared with 2^{n_i}, $i = 1,2$ the assumption on M is justified.

The vector-prototypes are chosen by independently assigning values $\pm$ 1 to the components. The probability that either value is taken is 1/2. Random vectors will be referred to with bold capitols (e.g. $\mathbf{X}$) whereas specific vector-outcomes are denoted in bold lower-case. For m=1,2,...,M, the input-prototypes are $\mathbf{F}_m$ and the output-prototypes are $\mathbf{G}_m$ when considered as random vectors. The components of the input vectors will be indexed by "i" (e.g. $\mathbf{F}_{ki}$) and the output vectors will be indexed by "j". The range of i is $1,2,\ldots,n_I$ and that of j is $1,2,\ldots,n_O$.

If $X_1, X_2, \ldots, X_n$ are independent identically distributed (*i.i.d.*) random variables (r.v.'s) on $\{-1,1\}$ with $p \equiv P(X_i=1)$ and S is their sum, then S is binomial with parameters $\pm$ 1, n, p. We denote this by $S \sim Bin(\pm 1, n, p)$. Similarly, if X is a normal r.v., with mean μ, and variance σ^2, we put $X \sim N(\mu, \sigma)$.

The matrix-associator will be referred to as "A". Whether a random matrix or a particular outcome is being discussed should be clear from context. To be consistent with the "i, j" indexing of input and output vectors, "i" will refer to the <u>column</u> and "j" to the <u>row</u> of a matrix entry, e.g. A_{ji}. We define the k^{th} **matrix-output** as

$$\mathbf{g}'_k \equiv A\mathbf{f}_k \tag{2}$$

[3] "Usable" for the purposes of input-classification

[4] For positive parameters, "$y >> x$" indicates that y is *minimally* $10 \cdot x$ and is typically much larger.

(The constant $1/n_I$ is dropped) and write the corresponding random vector as $\mathbf{G'}_k$. The dot-product of the k^{th} matrix-output and the l^{th} prototype-output is D_{kl} with outcome d_{kl}.

Parameters that take values in $\{-1,1\}$ are referred to as "bits" with -1 acting as the logical "0". Logical operations on these parameters are defined in this context as are terms "parity", "compliment" (logical), etc.

Derivation of Storage Limits

Given the M input-output prototype-pairs $(\mathbf{f}_m, \mathbf{g}_m)$, the matrix defined by equation (1) is seen as the sum of M outer-product matrices. The m^{th} outer-product or **association-plane** is completely determined by the n_I+n_O bits of $\mathbf{f}_m$ and $\mathbf{g}_m$. Its ji^{th} component, m_{ji} is the product $f_{mi}g_{mj}$, which takes values in $\{-1,1\}$. The m^{th} association-plane is not changed if both $\mathbf{f}_m$ and $\mathbf{g}_m$ are multiplied by -1. This indicates that the m^{th} plane represents at most n_I+n_O-1 bits of information. The n_I+n_O-1 entries that make up a particular row and column, are easily seen to be independent, so that n_I+n_O-1 is also the lower bound. In fact, the entries of the row and column are enough to determine every other entry in the plane. To illustrate, examine the k^{th} row and l^{th} column and the entry $m_{ji}=f_{mi}g_{mj}$. The three entries (bits) m_{ki}, m_{kl} and m_{jl} determine m_{ji} so that the parity of these four numbers is even.[5] Therefore each association-plane represents *exactly* n_I+n_O-1 bits.

When the association-planes are summed information is lost. Storage is bounded above by the information contained in the weights (entries) of the associator. An assessment of the matrix entropy provides a bound on the number of association pairs storable. To begin, it can be shown that the entropy or self-information of a r.v. $X \sim Bin(\pm 1, n, 1/2)$ is virtually identical to that of a normal r.v. with variance n. The A_{ji} are $Bin(\pm 1, M, 1/2)$ so each has entropy $H(A_{ji}) = \frac{1}{2}\log_2 2\pi e M$. An upper bound on the matrix entropy can be obtained by assuming independence of the individual weights. One multiplys the weight-entropy by the number of weights in the system to get $H(A) = \frac{1}{2}n_I n_O \cdot \log_2 2\pi e M$

For M stored associations, there are M! ways to map the M (distinct) inputs to the M (distinct) outputs. To produce an output vector for each input prototype that results in a correct output-decision, the matrix entropy must exceed $\log_2 M! \approx M\cdot[\log_2 M - \log_2 e]$.[6] For $M<M^*$ we must have

$$\frac{1}{2}n_I n_O \log_2 2\pi e M \;>\; M[\log_2 M - \log_2 e]$$

which leads to

---

[5]Therefore exactly half of the 16 concievable configurations of these four bits are possible.

[6]For $M > 2.2\cdot 10^4$, $\log_2 M \gg \log_2 e$ so that the $\log_2 e$ term can be ignored. Even for M as small as 3000 however, the approximation $\log_2 M! \approx \log_2 M$ is reasonably accurate.

$$\frac{M}{n_I n_O} < \frac{1}{2} \cdot \frac{\log_2 M + \log_2 2\pi e}{\log_2 M - \log_2 e}.$$

Generally we can ignore the term $\log_2 e$ and since $\log_2 2\pi e \approx 4$, an approximate bound is

$$\frac{M}{n_I n_O} < \frac{1}{2} + \frac{2}{\log_2 M}$$

For the systems considered, the right side of the inequality will not exceed 1 for M near M^*. As the net-size approaches infinity, M^* is seen to lie beneath one-half the net-size. An important observation here is that though one row and one column are enough to specify the bits in each assocition-plane, the other bits act to preserve information stored in the plane when the planes are summed together. Without the additional bits, the entropy of the row and column alone becomes $\frac{1}{2}(n_I + n_O)\log_2 2\pi e M$. This is much smaller than the entropy calculated above and will serve as a lower bound. The assumption of independent weights is false for individual association-planes but should be accurate for M near M^* since the inter-correlations between bits in a given plane should be "washed out" by "counter-correlations" in the other (independent) planes in the sum.

Measuring Similarity

The output decisions of a matrix-associator depend on the similarity measure used at the output. A given system will perform differently under different similarity measures. Therefore, the performance of a system must be defined with respect to a particular similarity measure. The general definition of similarity measure follows from the Hamming distance function. Defining $\{-1,1\}^n$ to be $\{\mathbf{x} \in \mathbf{R}^n \mid x_i \in \{-1,1\}, i=1,2,\ldots,n\}$, the Hamming Distance is the function $HD:\{-1,1\}^n \times \{-1,1\}^n \rightarrow \mathbf{R}$ given by $HD(\mathbf{x},\mathbf{y}) = \frac{1}{2}\sum_{i=1}^{n} |x_i - y_i|$. The Hamming Distance is the number of components at which $\mathbf{x}$ and $\mathbf{y}$ disagree. Its negative is a similarity measure on $\{-1,1\}$. If V is an n-dimensional vector-space, then a similarity measure is a function $S:V \times V \rightarrow \mathbf{R}$ such that for $\mathbf{x},\mathbf{y} \in V$,

1. $S(\mathbf{x},\mathbf{y}) = S(\mathbf{y},\mathbf{x})$
2. For $\mathbf{x}, \mathbf{y} \in \{\mathbf{x} \in V \mid \|\mathbf{x}\|=1\}$, $S(\mathbf{x},\mathbf{y})$ is maximized by $\mathbf{x}=\mathbf{y}$.
3. For $\mathbf{x},\mathbf{y},\mathbf{w},\mathbf{z} \in \{-1,1\}^n$, $HD(\mathbf{x},\mathbf{y}) < HD(\mathbf{w},\mathbf{z})$ *implies* $S(\mathbf{x},\mathbf{y}) \leq S(\mathbf{w},\mathbf{z})$

Under this type of similarity, $\mathbf{x}$ and $\mathbf{y}$ are to said to be more similar than $\mathbf{w}$, $\mathbf{z}$ whenever $S(\mathbf{x},\mathbf{y}) < S(\mathbf{w},\mathbf{z})$. The function is *maximal* for similar vectors. Condition 3 requires the similarity measure to be consistent with the negative Hamming distance similarity, $-HD(\mathbf{x}, \mathbf{y})$ on $\{-1,1\}^n$.

We allow the word "minimumized" to be replaced by "maximized" in 2 with the reversal of the inequality in 3. This results in a function that is *minimal* for similar vectors. The negative of a similarity function is therefore also a similarity function.

Examples of similarity measures include those based on *Minkowski Metrics*. For instance, either of the forms $S(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|^p$ or $S(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} |x_i + y_i|^p$, $p > 0$ or their negatives can be used. An inner-product can also be used, e.g. the *dot-product*, $S(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} x_i y_i$. The dot-product has several advantages the first of which is the relative ease of analysis it provides. The dot-product detection distributions are readily identified. Additionally, the dot-product similarity criterion should be a good benchmark for the expected performance of systems that are connected to the output of the matrix-associator. This is because an associator often determines its output by comparing the matrix-input with its stored input-prototypes via the dot-product similarity measure. The resultant output is constructed as a weighted sum of the output-prototypes according to how similar their respective input-prototypes are to the matrix-input. If an associator of this form is connected to the output of a first-stage matrix-associator, it will function best if the first stage always produces a vector that is close to the "correct" input-prototype of the second stage with respect to dot-product similarity.

Detection

The dot-product will be the subject of the analysis, so that $S(\mathbf{x}, \mathbf{y})$ will represent this function. Detection will consist of placing $\mathbf{f}_k$ at the input of the matrix, determining the output $\mathbf{g'}_k$ and calculating $S(\mathbf{g'}_k, \mathbf{g}_m)$ for $m=1,2,\ldots,M$. The value of m for which this quantity is largest will be chosen as the best match. Since the vectors were originally chosen randomly, the dot-products produced are random variables. The distribution of $S(\mathbf{G'}_k, \mathbf{G}_m)$ varies according to whether $m = k$. The condition $m = k$ is the **match condition** and defines the **match distribution** for the system. The condition $m \neq k$ is the **no-match condition** defining the **no-match distribution**. Determination of the distributions will allow evaluation of the probability P_e of an incorrect output-decision.

The dot product is $D_{kl} \equiv \mathbf{G'}_k \cdot \mathbf{G}_l$, $k = 1,2,\ldots,M$ where $\mathbf{G'}_k = A\mathbf{F}_k$, and A is given by (1). More explicitly,

$$\begin{aligned} \mathbf{G'}_k &= A\mathbf{F}_k \\ &= \sum_{m=1}^{M} \mathbf{G}_m \mathbf{F}_m^{\mathrm{T}} \mathbf{F}_k \\ &= \sum_{m=1}^{M} (\mathbf{F}_m \cdot \mathbf{F}_k)\mathbf{G}_m \end{aligned} \tag{3}$$

From this D_{kl} is seen to be,

$$\begin{aligned} D_{kl} &= \mathbf{G'}_k \mathbf{G}_l \\ &= \sum_{m=1}^{M} (\mathbf{F}_m \cdot \mathbf{F}_k)(\mathbf{G}_m \cdot \mathbf{G}_l) \end{aligned} \tag{4}$$

Since $\mathbf{F}_m \cdot \mathbf{F}_k = \sum_{i=1}^{nI} F_{mi} F_{ki}$ and similarly for $\mathbf{G}_m \cdot \mathbf{G}_l$, the sum for D_{kl} expands to

$$D_{kl} = \sum_{m=1}^{M} \sum_{i=1}^{n_I} \sum_{j=1}^{n_O} \mathbf{F}_{mi}\mathbf{F}_{ki}\mathbf{G}_{mj}\mathbf{G}_{lj} \tag{5}$$

The components of the prototype-vectors are chosen independently over $\{-1,1\}$ with each of these two values occurring with probability 1/2. This implies that the terms in (5) are "bits" by our definition so that $D_{kl} \sim Bin(\pm 1, Mn_In_O, 1/2)$ when $k \neq l$. For $k = l$,

$$D_{kk} = (\mathbf{F}_k \cdot \mathbf{F}_k)(\mathbf{G}_k\mathbf{G}_k) \quad + \sum_{m=1, m \neq k}^{M} \sum_{i=1}^{n_I} \sum_{j=1}^{n_O} \mathbf{F}_{mi}\mathbf{F}_{ki}\mathbf{G}_{mj}\mathbf{G}_{kj} \tag{6}$$

and $D_{kk} \sim Bin(\pm 1, (M-1) \cdot n_In_O, 1/2)$. For the assumed range of M, $M - 1 \approx M$ so that D_{kk} and D_{kl} have the same variance, $\sigma_D = Mn_In_O$. The two distributions are identical except for the difference in the means. The mean of the sums in (6) and (5) are zero. The first term in (6) however, is the constant n_In_O. The match distribution then, has mean $\mu_1 = n_In_O$ and the no-match has mean $\mu_2 = 0$.

The separation, d' of the two distributions is defined as the absolute difference of the means divided by the geometric mean of the standard-deviations. Since the same standard-deviation is common to both distributions, d' is the difference between the means measured in standard-deviation-length units:

$$\begin{aligned} d' &\equiv \frac{|\mu_1 - \mu_2|}{\sigma_D} \\ &= \frac{n_In_O}{\sqrt{Mn_In_O}} \\ &= \sqrt{n_In_O/M} \end{aligned} \tag{7}$$

The larger the relative separation between between the distributions, the smaller the probability that an outcome from one distribution will be found near typical outcomes from the other distribution. As we will see, a large d' will afford a low error-rate. From (7), d' increases with increasing net-size and decreases with M as would be expected.[7]

Evaluation of Error Probability

In order to determine the information storage for a system whose net-size is n_In_O with M stored associations, P_e must be determined as a function of M. An error on the k^{th} input, $P_{e,k}$ occurs if there is an $l \in \{1,2,...,M\}$, $l \neq k$ such that $D_{kl} \geq D_{kk}$. The average over k of $P_{e,k}$ is P_e.

Let R_k denote the range of possible values of D_{kk}. One minus the probability that an error occurs is the probability that $D_{kl} \geq D_{kk}$, i.e.

[7] A matrix with a large number of stored associations should poorly discriminate between match v.s. no-match output-prototype vectors

$$1 - P_{e,k}$$
$$= \sum_{a \in R_k} P(D_{k1} < a, D_{k2} < a, \ldots, D_{kk-1} < a, D_{kk} < a, D_{kk+1} < a, \ldots, D_{km} < a)$$

Since $D_{kl} = \mathbf{G'}_k \cdot \mathbf{G}_l$ and $D_{kl'} = \mathbf{G'}_k \cdot \mathbf{G}_{l'}$ the two r.v.'s both contain information from $\mathbf{G'}_k$ and are not strictly independent. However, the dot-products are independent given $\mathbf{G'}_k$ and they each provide very little information about its components. We assume then that they are very nearly independent. This allows approximation of $P_{e,k}$ by

$$P_{e,k} = 1 - \sum_{a \in R_k} P(D_{kk} = a) \prod_{l=1, l \neq k}^{M} P(D_{kl} < a).$$

The D_{kl}, $k \neq l$ are identically distributed as no-matches, so letting D_k be a r.v. with the no-match distribution gives[8]

$$P_{e,k} = 1 - \sum_{a \in R_k} P(D_k \leq a)^{M-1} P(D_{kk} = a) \tag{8}$$

If we define F' as the distribution $Bin(\pm 1, Mn_I n_O, 1/2)$ with mean of zero, and f' as the corresponding density function, then (8) can be written,

$$P_{e,k} = 1 - \sum_{a \in R_k} F'(a) \cdot f'(a - \mu_1) \tag{9}$$

where the argument to f' must be displaced by the mean of D_{kk}. The distribution, F' can be "normalized" by dividing all dot-product r.v.'s by $\sigma_D = \sqrt{Mn_I n_O}$ to obtain the distribution $F \sim Bin(\pm 1/\sqrt{Mn_I n_O}, 1, 1/2)$ with mean of zero. The error $P_{e,k}$ becomes

$$P_{e,k} = 1 - \sum_{a \in R_k} F(a)^{M-1} \cdot f(a - d') \tag{10}$$

where f is the density of the normalized distribution F.

The expression above is not dependent on k, so the average probability P_e that an input will produce an error at the output is given by equation (10); the matrix-channel has been shown to be M-ary symmetric. If $\mathbf{X}$ represents the input vector r.v. and $\mathbf{Y}$ the subsequent output vector r.v., then the information per association is given by

$$I(\mathbf{X};\mathbf{Y}) = \log_2 \mathrm{M} - P_e \cdot \log_2 (\mathrm{M}-1) - H_b(P_e) \tag{11}$$

where $H_b(x) \equiv -x \log_2 x - (1-x) \log_2 (1-x)$, $0 \leq x \leq 1$ is the binary entropy function.

For a given matrix-class, we evaluate the storage $M \cdot I(\mathbf{X};\mathbf{Y})$ for increasing M until the maximum storage is found. The maximum is called the storage capacity of the net. The value M^* of M that produces the maximum is called the **storage addressability** of the system under this storage

---

[8]Since $P(D_k = a) \approx 0$, no distinction between $P(D_k < a)$ and $P(D_k \leq a)$ is made.

characterization. Uniqueness of M^* depends upon the nature of $I(\mathbf{X};\mathbf{Y})$ as a function of M. This function is plotted in figure for a net-size of $10^{5.5}$. The **through-put addressability** is the value M^o of M at which the maximum is achieved. The function is believed to be unimodal, increasing to a maximum before M reaches the net-size and then decreasing rapidly thereafter. The storage should reach a maximum before M reaches the bound given by equation and remaining low for larger M as long as $M \ll 2^{n}i,\ i = 1,2$ is satisfied. The numerical analysis carried out to date bears this out. However, a normal approximation to the distribution F in equation (10) was used and is highly inaccurate for large M. Presently, a more accurate approximation is being devised [29] Numerical methods based on the new approximation and actual simulations of associator matrices will be used to determine storage of the systems and the validity of the analysis.

Data-Dependence of Capacity

In the forgoing development, we assumed the vector-prototypes were chosen randomly. Random vectors' tendency toward pairwise orthogonality keeps interference among associations low. Subsequent sections examine *suboptimal* prototype storage and retreival. The object will be to characterize deleterious effects of storing low-entropic associations.

Storage Efficiency

Storage efficiency of a matrix is the matrix-storage divided by the information required to represent a matrix associator on M associations. Examination of equation (1) reveals that each entry in an associator matrix is the sum of M bits. The range of values of each entry is the integers between -M and M. The extremes are realized whenever the bits for that entry all agree in value. Further, the entry will be be even if and only if M is even. It follows that the number of values an entry can assume is M+1. This means that $n_I n_O$ weights will require $n_I n_O \log_2 (M+1) \approx n_I n_O \log_2 M$ bits for storage. Letting $\Sigma = M{\cdot}I(\mathbf{X};\mathbf{Y})$ be the storage of the net, then we define the efficiency η by

$$\eta = \frac{\Sigma}{n_I n_O \log_2 M}$$

Since $I(\mathbf{X};\mathbf{Y}) \leq \log_2 M$ by equation (11), it follows that $\Sigma \leq M\log_2 M$ and we have

$$\eta \leq \frac{M\log_2 M}{n_I n_O \log_2 M} = \frac{M}{n_I n_O}$$

From equation , the bound becomes

$$\eta \leq \frac{1}{2} + \frac{2}{\log_2 M}$$

If one could take advantage of the fact that each weight has entropy $1/2{\cdot}\log_2 2\pi e M$, the information required to impliment the matrix becomes $1/2{\cdot}n_I n_O \log_2 2\pi e M$ as stated earlier. One could therefore define the efficiency by

$$\eta = \frac{\Sigma}{1/2{\cdot}n_I n_O \log_2 2\pi e M}$$

Equation stipulates the efficiency defined this way is less than unity.

The second of these efficiency definitions might be most appropriate if $1/2n_In_O\log_2 \cdot 2\pi eM$ were the maximum achievable entropy of the weights. However, a method for achieving the matrix entropy $n_In_O\log_2(M+1)$ is being formulated through judicious choice of the associations to be learned. If successful, the maximum storage possible for a matrix would be shown to be $n_In_O\log_2 M$. The first definition of efficiency would then indicate the relation of random storage to optimal storage.

Sensitivity of Storage to Vector Correlation

Previously the vector-components of the prototype-vectors were independently selected from $\{-1,1\}$ with probability 1/2 that either value was taken. If a bias is made in choosing the vectors so that the probability that the value 1 occurs is p for each vector component, then the storage capacity is adversely affected. In this sense, the unbiased selection was optimal. Two questions are important for the consideration of biased vector selection:

1. What does bias cost in terms of reduced memory capacity?

2. How nearly unbiased must the selection process be in order for the matrix to perform nearly optimally?

The first question addresses the severity of memory degradation with bias. The second relates to the practicality of achieving near optimal storage.

The analysis reveals capacity degradation as a consequence of reduced d' due to bias-induced vector-correlation. The **bias**, Δ is defined as $\Delta = |p - 1/2|$ where the **bias-probability** p, is the probability that any vector component is assigned the value 1. The input may be selected with a different bias than the output so we let p_F^* be the bias-probability for the input prototypes and p_G^* be the bias-probability for the output.

To see how bias affects vector correlation, let **U** and **V** be n-dimensional vectors on $\{-1,1\}$ with bias-probability p^*. When the components are chosen independently, the probability that a component of **U** will agree with its counterpart in **V** is

$$
\begin{aligned}
P(U_i = V_i) &= P(U_i = 1,\ V_i = 1) + P(U_i = -1,\ V_i = -1) \qquad i = 1,2,\ldots,n. \\
&= P(U_i = 1)P(V_i = 1) + P(U_i = -1)P(V_i = -1) \\
&= p^{*2} + (1 - p^*)^2 \\
&= 2\cdot(p^* - 1/2)^2 + 1/2 \\
&= 1/2 + 2\cdot\Delta^2
\end{aligned}
\tag{12}
$$

So $P(\mathbf{U}_i = \mathbf{V}_i) \geq 1/2$ with equality when p^* is 1/2 $\quad(\Delta = 0)$.

We define $p_F \equiv p_F^{*2} + (1 - p_F^*)^2$ to be the probability that $F_{mi} = F_{m'i}$ for arbitrary

$m, m' \in \{1,2, \ldots ,M\}$ and $i \in \{1,2, \ldots ,n_I\}$. We say that the input prototypes are p_F-correlated. Similarly, the parameter p_G represents the correlation between pairs of output prototypes. The d' parameter can be evaluated for the system by determining the mean and variance of both the match and no-match distributions. The derivation of these is tedious and non-informative and so will be left to an appendix. Results pertinent to the discussion will be related here. For the match distribution, the mean μ_1 and variance σ_1^2 are

$$\mu_1 = n_I n_O \cdot [1 + (M-1)(2p_F - 1)(2p_G - 1)]$$

$$\sigma_1^2 = n_I n_O M[1 + M(2p_F - 1)(2p_G - 1)][1 - (2p_F - 1)(2p_G - 1)] \tag{13}$$

The no-match parameters are[9]

$$\mu_2 = n_I n_O[(2p_F - 1) + (2p_G - 1) + M(2p_F - 1)(2p_G - 1)]$$

$$\sigma_2 = n_I n_O M \cdot [1 + M(2p_F - 1)(2p_G - 1)(1 - (2p_F - 1)(2p_G - 1))] \tag{14}$$

If p_F and p_G are set to 1/2 in the above equations, the mean and variance assume the values for the unbiased distributions considered earlier. On the other hand, if each bias is large enough (but not too close to 1) for the relation

$$M(2p_F - 1)(2p_G - 1)(1 - (2p_F - 1)(2p_G - 1)) \gg 1 \tag{15}$$

to hold, then both the match and no-match variances can be approximated by $n_I n_O M^2(2p_F - 1)(2p_G - 1)(1 - (2p_F - 1)(2p_G - 1))$. The absolute difference between the means is $4n_I n_O(p_F - 1)(p_G - 1)$ so that from the definition of d' in (7), we have[10]

$$d' = \frac{4\sqrt{n_I n_O}}{M} \cdot \frac{(1 - p_F)(1 - p_G)}{\sqrt{(2p_F - 1)(2p_G - 1)(1 - (2p_F - 1)(2p_G - 1))}} \tag{16}$$

Whereas d' varied inversely as $\sqrt{M}$ in the unbiased case, it varies inversely as M when a bias is present. Therefore, a bias is thought to severely limit the capacity of the associator. On the other hand, a bias must be present on *both* the input and output vectors for the effect to be present. Correlated vectors are not as nearly orthogonal as are uncorrelated vectors. Interference effects will not be present if the associator either maps correlated vectors to nearly orthogonal vectors or vice-versa. In particular, if correlated input vectors are associated to uncorrelated output vectors, no resulting capacity degradation is present. An associator could be used as a "front-end" to other associator units in order to translate correlated input vectors to uncorrelated outputs for further processing.

[9] Notice the subtle difference between the match and no-match variances. This is not an error!

[10] In this discussion, the correlations are considered as by-products of the bias so that the vector prototypes can be considered as mutually independent. However, calculation of the match/no-match mean and variances and that of d' was carried out without the assumption of independence between respective components of the prototype vectors.

In order for correlation effects not to be significant, the bias should be small enough so that the reverse of the conditions (15) should hold. One could ignore p_F and p_G in (14) and (13) if they satisfied[11]

$$M(2p_F - 1)(2p_G - 1) \leq 1/9$$

Say for example, the bias of the input and the output prototypes were the same. We set both p_F and p_G equal to $1/2 \pm \Delta^2$ in accordance with (12). From condition , it follows that $M(2p_F - 1)(2p_G - 1) \gg 1$. The bias, Δ would have to satisfy $[2(1/2 \pm 2\Delta^2) - 1]^2 \leq 1/9M$ so that Δ cannot exceed $\frac{1}{2\sqrt{3}} M^{-1/4}$. Large associators with many stored associations will require small values of Δ to perform nearly optimally. It is the large systems that will suffer substantial capacity deterioration if care is not taken to insure that the vector prototypes are chosen with nearly even distribution of -1's and 1's.

When Δ is large enough to limit performance, it is desirable to substitute d' from equation (16) into (10) and (11) to estimate the reduced capacity. A large bias however will compromise the independence of the dot-products D_{kl}, $k,l \in \{1,2, \ldots ,M\}$ that was assumed for the derivation of (10). At best, (10) might be accurate for the smallest values of Δ in the non-optimal range. If we assume p_F equals p_G, then the smallest non-optimal value for M associations is determined from (15) and so must satisfy

$$M(2p_F - 1)(2p_G - 1) \gg 1$$

We take "9" to be much greater than 1 and get

$$\Delta \approx \frac{\sqrt{3}}{2} M^{-1/4}$$

An upper bound on the capacity may be found by estimating the entropy of the matrix weights which will be distributed as $Bin(\pm 1,n,p)$ where p is determined from p_F and p_G. Again, only the smallest values of non-optimal Δ can be considered by this method since the weights will lose their independence as the bias becomes large.

---

[11] The fraction "1/10" is $\ll 1$ but 9 is a perfect square so "1/9" is used.

References

1. Aigner, Martin. *A Series of Comprehensive Studies in Mathematics.* Volume 234: *Conbinatorial Theory.* Springer-Verlag, New York, New York, 1979.

2. Anderson, James A., Silverstein, Jack W., Ritz, Stephen A., and Jones, Randall S. "Distinctive Features, Categorical Perception, and Probability Learning: Some Applications of a Neural Model". *Psycological Review 84*, 5 (1977), 413-451.

3. Anderson, James A. "Cognitive and Psychological Computation with Neural Models". *IEEE Transactions on Systems, Man, and Cybernetics SMC-13*, 5 (September/October 1983).

4. Ash, Robert B.. *Inter-Science Tracts in Pure and Applied Mathematics.* Volume 19: *Information Theory.* John Wiley and Sons, New York, New York, 1965.

5. Barto, A. G. "Learning by Statistical Cooperation of Self-Interested Computing Elements". *Human Neurobiology 4* (1985), 229-256.

6. Conte, Samuel D., and de Boor, Carl. *International Series in Pure and Applied Mathematics.* Volume : *Elementary Numerical Analysis, An Algorithmic Approach, 3rd Ed.* McGraw-Hill Books, 1980.

7. Duda, Richard and Hart, Peter. *Pattern Classification and Scene Analysis.* John Wiley and Sons, New York, New York, 1973.

8. Dudewicz, Edward J., and Ralley, Thomas G.. *The Handbook of Random Number Generation and Testing with TESTRAND Computer Code.* American Sciences Press, P.O. Box 21161, Columbus, Ohio 43221, 1981.

9. Feller, William. *An Introduction to Probability Theory and its Applications 3rd. Ed.* John Wiley and Sons, New York, New York, 1968.

10. Gallager, Robert G.. *Information Theory and Reliable Communication.* John Wiley and Sons, New York, New York, 1968.

11. Golden, Richard M. *Modelling Causal Schemata in Human Memory: A Connectionist Approach.* Ph.D. Th., Dept of Psychology, Brown University, Providence Rhode Island, 1986.

12. Golden, Richard M. "The "Brain-State-in-a-Box" Neural Model Is a Gradient Descent Algorithm". *Journal of Mathmatical Psychology 30*, 1 (March 1986), 73-80.

13. Grossberg, Steven. *Boston Studies in the Philosophy of Science.* Volume 70: *Studies of Mind and Brain, Neural Principles of Learning, Perception, Development, Cognition and Motor Control.* D. Reidel Publishers, Boston, Mass., 1982.

14. Grossberg, Stephen. Competive Learning: From Interactive Activation to Adaptive Resonance. Bibliographic information on this article is not complete.

15. Hinton, Geoffrey E., and Anderson, James A.. *Parallel Models of Associative Memory.* Lawrence Erlbaum Associates, 365 Broadway, Hillsdale, New Jersey 07642, 1981.

16. Hinton, Geoffrey E. "Boltzmann Machines: Constraint Satisfaction Networks that Learn". *Cognitive Science 9* (1985), 147-169.

17. Hopfield, J. J. "Neurons with Graded Response have Collective Computational Properties like those of Two-State Neurons". *Proceedings of National Acadamy of Science 81* (May 1984).

18. Hopfield, J.J. and Tank, D.W. 'Neural' Computation of Decisions in Optimization Problems. Research Document. Information pertaining to the source of this document is incomplete.

19. Kanerva, Pentti. *Self-Propagating Search: A Unified Theory of Memory.* Ph.D. Th., Stanford University, 1983.

20. Knuth, Donald E.. *Addison-Wesley Series in Computer Science and Information Processing.* Volume 2: *The Art of Computer Programming, 2nd Ed.* Addison-Wesley, Reading Mass., 1968.

21. Kohonen, Tuevo. *Springer Series in Information Sciences.* Volume 8: *Self-Organization and Associative Memory.* Springer-Verlag, New York, New York, 1984.

22. Krylov, V.I.. *Approximate Calculation of Integrals.* Translated from the Russian by A. H. Stroud. MacMillan (Crowell-Collier Publishers, New York, New York, 1962. CopyWright A. H. Stroud 1962.

23. Lancaster Peter. *Theory of Matrices.* Reproduced Photographically by Author from the 1969 Edition of Academic Press, 1977.

24. Lindgren, Bernard W.. *Statistical Theory, 3rd Ed.* MacMillan Publishing, New York, New York, 1976.

25. McEliece, Robert J.. *Encyclopedia of Mathematics and its Applications.* Volume 3: *The Theory of Information and Coding.* Addison-Wesley, Reading, Mass., 1977.

26. Minsky, Marvin and Papert, Seymour. *Perceptrons, An Introduction to Computational Geometry.* M.I.T. Press, Cambridge, Mass., 1969.

27. Parker, David B. Learning Logic. TR-47, Center for Computational Research in Economics and Management Science, M.I.T., April, 1985.

28. Pearlmutter, Barak A. and Hinton Geoffrey E. G-Maximization: An Unsupervised Learning Procedure for Discovering Regularities. Neural Networks for Computing, American Institute of Physics, 1986. Bibliographic information not complete.

29. Pelzer, David B. and Pratt, John W. "A Normal Approximtion for Binomial, F, Beta, and Other Common Related Tail Probabilities". *American Statistical Association Journal 1*, 68 (December 1968), 1416-1483.

30. Rosenblatt, Frank. *Principles of Neurodynamics.* Spartan Books, New York, New York, 1962.

31. Rumelhart, David E., McClelland, James L., and the PDP Research Group. *Parallel Distributed Processing, Explorations in the Microstructure of Cognition.* M.I.T. Press, Cambridge, Mass., 1986.

32. Rumelhart, David E., Hinton, Geoffrey E., and Williams, Ronald J. Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing, Explorations in the Microstructure of Cognition*, M.I.T. Press, 1986, pp. 318-362.

33. Rumelhart, David E. and McClelland, James L. On Learning the Past Tenses of English Verbs. In *Parallel Distributed Processing, Explorations in the Microstructure of Cognition*, M.I.T. Press, 1986, pp. 216-271.

34. Samuelson, Paul A. "Constructing an Unbiased Random Sequence". *American Statistical Association Journal 1*, 68 (December 1968), 1526-1527.

35. Schneider, Walter and Mumme, Dean C. Attention, Automatic Processing and the Compiling of Knowledge: A Two-Level Architecture for Cognition. To appear in Psychology Review.

36. Sejnowski, Terrance J. and Rosenberg, Charles R. NETtalk: A Parallel Network that Learns to Read Aloud. JHU/EECS-86/01, The Johns Hopkins University Electrical Engineering and Computer Science, 1986.

37. Shiffrin, Richard M. and Schneider, Walter. "Controlled and Automatic Human Information Processing: II. Perceptual Learning, Automatic Attending, and a General Theory". *Psychological Review 84*, 2 (1977), 127-189.

38. Stroud, A. H. and Don Secrest. *Gaussian Quadrature Formulas*. Prentice Hall, Englewood Cliffs, New Jersey, 1966.