

Neural Network Associative Memories and Input Space With Non-Orthonormal Basis

JOEL SUAREZ

Centro de Investigación en Tecnologías de Información y Sistemas

Universidad Autónoma del Estado de Hidalgo

One of the main problems with using Neural Network Associative Memories appears when the signals to be learned determine a non-orthogonal basis. In dealing with this type of basis, an error is introduced in the association and, because of this, a perfect matching is not obtained. Based in theoretical considerations only, this paper addresses the problem of getting rid cross-talk terms in the final recalling of an autoassociative neural network when input patterns belong to a subspace embedded into the whole input pattern space. One of the methods obtained also considers optimization in matrix weight size, at the expense of redefining the weight matrix, the activation function and the output space. Particularly interesting results the fact that we get rid of the cross-talk term by just considering one-single layer heteroassociative-competitive neural network defined on the subspace generated by the set of linearly independent input patterns. The results are very important in practical questions such as image and pattern recognition, as well.

Categories and Subject Descriptors: I.5.1 [**Pattern Recognition**]: Models—*Neural nets*; I.4.0 [**Image Processing and Computer Vision**]: General—*Image displays*; I.4.5 [**Image Processing and Computer Vision**]: Image Representation—*Morphological*

General Terms: Algorithms, Design, Performance, Reliability

Additional Key Words and Phrases: Association, Basis, Neural network, Perfect matching

The author wishes to thanks to the *Centro de Investigación en Tecnologías de Información y Sistemas* for its kind support. **Paper to be submitted for publication to JERIC.**

Author address: CITIS/Universidad Autónoma del Estado de Hidalgo, Carretera Pachuca/Tulancingo, Km. 4.5, Cd. Universitaria, C.P. 42184, Pachuca de Soto, Hidalgo, México. e-mail jsuarez@uaeh.reduaeh.mx

Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.

© 2002 ACM 0004-5411/2002/0100-0001 \$5.00

1. INTRODUCTION

One of the main problems with using Neural Network Associative Memories appears when the signals to be learned determine a non-orthogonal basis. In dealing with this type of basis, an error is introduced in the association and, because of this, a perfect matching is not obtained [Haykin 1994; Zurada 1992]. The problems arising from a non-orthogonal basis of input patterns has been addressed by several researchers when binary and bipolar signals are considered [Haryono and Phien 1994], but we are interested here in the continuous-valued input pattern case because of its importance in lecturing and the development of pattern recognition projects.

Haykin [1994] proposes a preprocessor to perform a *Gram-Schmidt orthonormalization* on the input basis prior to the association. This transformation maintains a one-to-one correspondence between the input basis and the orthonormal basis obtained through the preprocessor. However, it is not difficult to prove that such a procedure alone does not remove the presence of noise if the input vector is not an element of the non-orthogonal basis.

This paper focuses on the problem of constructing an hybrid neural network on input space with non-orthogonal basis, as a combination of competitive and associative memory neural network, with the purpose of obtaining a perfect matching; in other words, an association without noise for input signals necessarily belonging to the subspace generated by the input basis. The more general case, where the input signal does not necessarily belong to the subspace generated by the non-orthogonal basis, deserves space in a future work.

The rest of the paper is divided in four more sections, where the second section defines some mathematical notations and formally defines the basic notions of the problem. The third section poses a trivial solution to the problem of perfect recall by introducing a non-optimal hybrid neural network architecture, while the fourth and fifth sections focus on the problem of minimizing the dimensionality of the input space and on conclusions, respectively.

2. MATHEMATICAL NOTATION AND SOME BASIC DEFINITIONS

To start with, we assume that the input signal, α , belongs to the n -dimensional Euclidean space, \mathbf{R}^n , and that the set $V = \{\alpha_0, \dots, \alpha_{m-1}\}$ is a non-orthogonal basis of \mathbf{R}^m , where $m \leq n$; in other words, V generates a subspace, $\mathcal{L}(V)$, of \mathbf{R}^n . We will assume also that the finite set $W = \{\beta_0, \dots, \beta_{m-1}\}$ contains patterns that are orthogonal to elements in V in a one-to-one correspondence.

The problem can be formally posed as follows: Given an element $\alpha \in \mathbf{R}^n$, we are interested in constructing an associative memory neural network that perfectly matches α with some element in V . Notice that α does not necessarily belong to $\mathcal{L}(V)$; in fact, we can make a division of the problem depending on the subspace of \mathbf{R}^n which the input pattern α belongs to, and on the orthogonality of the elements of V , Table I shows this.

Case	$\mathcal{L}(V)$	Subspace which α belongs to	V
1	\mathbf{R}^m ($m \leq n$)	$\mathcal{L}(V)$	Orthogonal
2	\mathbf{R}^m ($m < n$)	\mathbf{R}^n	Orthogonal
3	\mathbf{R}^m ($m \leq n$)	$\mathcal{L}(V)$	Non-orthogonal
4	\mathbf{R}^m ($m < n$)	\mathbf{R}^n	Non-orthogonal

Table I. Possible combinations of subspace and input patterns

In Table I, although $\alpha \in \mathbf{R}^n$, to say that α belongs to the subspace generated by V means that α can be written as a linear combination of the elements in V , which are also elements of \mathbf{R}^n . In any case, the weight matrix, Ω , of the associative neural network can be defined by just considering the elements of the basis V , and it is given by Expression (1), where α^T means the transpose of α

$$\Omega = \sum_{i=0}^{m-1} \alpha_i^T \alpha_i \quad (1)$$

Ω is an order n matrix, and the recall process is activated by multiplying this matrix with the input pattern α^T [Haykin 1994; Zurada 1992], as Expression (2) shows

$$\Omega\alpha^T = \sum_{i=0}^{m-1} \alpha_i^T \alpha_i \alpha^T \quad (2)$$

As an example, consider case 1 in Table 1, where $m \leq n$ and R^m is generated by V . Therefore, α can be written as a linear combination of the elements in V as Expression (3) shows

$$\alpha = \sum_{i=0}^{n-1} \lambda_i \alpha_i \quad (3)$$

The coefficients λ_i in Expression (3) will be called *components of α in basis V* [Hoffman 1971]. After substituting Expression (3) in Expression (2) and making some calculations, we get Expression (4) for some $k \in \{0, 1, 2, \dots, m-1\}$

$$\Omega\alpha^T = \lambda_k (\alpha_k \alpha_k^T) \alpha_k^T + \sum_{i \neq k} \sum_{j \neq k}^{m-1} \lambda_j \alpha_i^T (\alpha_i \alpha_j^T) \quad (4)$$

What Expression (4) says is that we are expecting that the associative neural network recalls α_k^T as soon as α^T is presented. However, because of the factors in α_k and the second term in the sum (from left to right), called *the cross-talk term*, the matching is not perfect. Expression (4) is a general one ($m \leq n$ is the only restriction), but even under the assumption of orthonormality of basis V (case 1 in Table 1), the cross-talk term exists, as given in Expression (5), unless, of course, $\alpha \in V$

$$\Omega\alpha^T = \lambda_k \alpha_k^T + \sum_{i \neq k}^{n-1} \lambda_i \alpha_i^T \quad (5)$$

If $\alpha \in V$, then $\alpha = \alpha_k$, $\lambda_k = 1$, for some $k \in \{0, 1, \dots, m-1\}$, $\lambda_i = 0$, $\forall i \neq k$, and from expressions (4) and (5) we get, respectively, $\Omega\alpha^T = (\alpha_k \alpha_k^T) \alpha_k^T$ and $\Omega\alpha^T = \alpha_k^T$. In fact, it is not difficult to prove that the cross-talk term does not exist if, and only if, $\alpha \in V$ and that the same result holds for case 3 in Table I. In other words, at most a recalling with a scaling factor exists if, and only if, $\alpha \in V$ in cases 1 and 3 of Table I.

A more general problem is that of heteroassociation [Kosko 1988], where elements

in V should be associated with m elements in a different space, say \mathbf{R}^p . Although we are not interested in discussing this situation, the matrix memory $\mathbf{\Omega}$ has a size $p \times n$ (it is not an order n matrix any more) and is computed through the formula

$$\mathbf{\Omega} = \sum_{i=0}^{m-1} \gamma_i^T \alpha_i$$

where $\mathbb{V} = \{\gamma_0, \dots, \gamma_{m-1}\}$ is a finite set of patterns in \mathbf{R}^p . Like in the autoassociation problem, the cross-talk term does not appear when V defines an orthonormal basis and α belongs to V .

Since we are requiring a perfect recalling in general, the problem consists in removing the cross-talk term. In the next two sections we discuss some simple solutions to this problem for cases 1 and 3 of Table I.

3. HYBRID NEURAL NETWORK

Usually, in the recall process, the memory matrix $\mathbf{\Omega}$ is directly multiplied by the input pattern α ; i.e., without giving the explicit linear combination in terms of the elements of V . On the other hand, one simple way to get rid of the cross-term consists in finding the minimum Euclidean distance among the input pattern α and the elements in V . This can be done through a competitive neural network.

We define a winner takes-all competitive neural network with a number of neurons equal to the dimensionality of the subspace generated by V ; in other words, such that $\forall \alpha_i \in V$ there does exist a single neuron in the competitive neural network that, after competition, fires 1 while the rest of neurons fire 0's. It is clear that the output of such a neural network will be a canonical vector of the space \mathbf{R}^m .

Assume that the canonical basis of the Euclidean space \mathbf{R}^m is given by the finite set $\mathcal{V} = \{e_0, \dots, e_{m-1}\}$, where $e_i \in \mathbf{R}^m$ has a 1 in i th entry and 0's otherwise. Every one of these canonical vectors should be associated with just one of the m vectors of V . Therefore, we have changed the original association problem by one of heteroassociation. However, the input patterns to be learned define now an orthonormal basis, which is an important requirement to get rid of the cross-talk term.

If $C: \mathcal{L}(V) \rightarrow \mathcal{V}$ denotes the competitive neural network and

$$H(x_i) = \delta_{i, \min\{\max\{j | x_j = \max\{x_j\}\}}}$$

its activation function, where $\delta_{i,j}$ is the well known Kronecker's delta function, then $\mathbf{H}(\mathbf{I}(\boldsymbol{\alpha})) = \mathbf{e}_i$ describes the whole behavior of C , for some winning neuron i , where $\mathbf{I}: \mathbf{R}^n \rightarrow \mathbf{R}^m$ defines the integration function as the negative of the Euclidean distance between $\boldsymbol{\alpha}$ and every pattern in V . If we design the competitive network in such a way that $\mathbf{H}(\mathbf{I}(\boldsymbol{\alpha}_i)) = \mathbf{e}_i$, then we require that the heteroassociative network, defined by the memory matrix $\boldsymbol{\Omega}$, associates \mathbf{e}_i with $\boldsymbol{\alpha}_i$.

Therefore, the matrix memory $\boldsymbol{\Omega}$ is then computed for an heterogeneous associative memory network as given in Expression (6)

$$\boldsymbol{\Omega} = \sum_{i=0}^{m-1} \boldsymbol{\alpha}_i^T \mathbf{e}_i \quad (6)$$

For example, assume that $\boldsymbol{\alpha}_k$ is the nearest pattern to the input pattern $\boldsymbol{\alpha}$. Therefore

$$\boldsymbol{\Omega} \mathbf{H}(\mathbf{I}(\boldsymbol{\alpha})) = \boldsymbol{\Omega} \mathbf{e}_k$$

and, by substituting Expression (6) we get

$$\boldsymbol{\Omega} \mathbf{H}(\mathbf{I}(\boldsymbol{\alpha})) = \sum_{i=0}^{m-1} \boldsymbol{\alpha}_i^T \delta_{i,k}$$

which is exactly $\boldsymbol{\alpha}_k$, and the cross-talk term has been removed.

At this stage all seems to be fine, but when we try to apply the whole thing to some practical situations (as, for example, image association) some problems arise. For example, in dealing with GIF images with a 360 x 300 resolution, we get input patterns of dimensionality 1 x 108,000 which implies some very serious problems of computer's memory (a memory matrix $\boldsymbol{\Omega}$ of order 108,000!). Therefore, the main assumptions in the discussion above are that there are no restrictions in computer's memory and computer's speed, that the number of images to work with in an association problem is a small one, and that the dimensionality of the input patterns is not bigger enough.

4. INPUT DIMENSION

To solve the problem of dimensionality already mentioned in previous section, we need to look at Expression (3) again. Still considering the cases 1 and 3 of Table I, assume that by some process we obtain an ordered set W of orthogonal patterns, which are in a one-to-one correspondence with elements in V ; in other words, pattern $\beta_i \in W$ is orthogonal to pattern $\alpha_k, \forall k \neq i$. Later on we will prove that such a finite set does exist, indeed.

Given a pattern α , the set of patterns W will let us to compute the value of the components in basis V by using Expression (3) as follows

$$\lambda_k = \frac{\alpha \beta_k^T}{\alpha_k \beta_k^T} \quad (7)$$

Considering that we are dealing with a problem of basis transformation (from canonical basis of \mathbf{R}^n to V , which generates \mathbf{R}^m as a subspace embedded in \mathbf{R}^n), through Expression (7) we can compute the $m \times n$ matrix transformation, \mathbf{P} , as follows

$$[\mathbf{P}]_{i,j} = \frac{\epsilon_j \beta_i^T}{\alpha_i \beta_i^T} \quad (8)$$

where the canonical basis of \mathbf{R}^n is given by the set $\mathcal{U} = \{\epsilon_0, \dots, \epsilon_{n-1}\}$.

In case of $m = 1$, $\alpha \in \mathbf{R}^n$ and belonging to the subspace generated by $V = \{\alpha_0\}$, where $\alpha_0 \in \mathbf{R}^n$, then $\alpha = \lambda \alpha_0$. In this situation, there is no need to look for patterns orthogonal to the elements of V , since computing the $1 \times n$ matrix transformation \mathbf{P} is straightforward; namely, $\mathbf{P}: \mathbf{R}^n \rightarrow \mathcal{L}(V)$ is defined as $\mathbf{P} = \frac{\alpha}{\alpha_0 \alpha_0^T}$, which means that the component of α in V basis is $\mathbf{P} \alpha^T = \frac{\alpha \alpha^T}{\alpha_0 \alpha_0^T}$. Just to make clear this point, consider the following simple example.

Example

Let $n = 4$ and $m = 2$; i.e., we are considering the four-dimensional and two-dimensional Euclidean spaces as the working space and the embedded subspace, respectively. Furthermore, assume that $V = \{(1, 2, 1, 0), (0, 1, 1, 2)\}$ and let $\alpha \in \mathbf{R}^4$ be an element of the subspace generated by V , i.e. $\alpha = \lambda_0(1, 2, 1, 0) + \lambda_1(0, 1, 1, 2)$.

Following Expression (8), and after considering $\beta_0 = (1, -2, 0, 1)$ as orthogonal to $\alpha_1 = (0, 1, 1, 2)$, and $\beta_1 = (-1, 0, 1, 1)$ as orthogonal to $\alpha_0 = (1, 2, 1, 0)$, the 2 x 4 matrix transformation \mathbf{P} is equal to

$$\begin{pmatrix} -\frac{1}{3} & \frac{2}{3} & 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \end{pmatrix}.$$

With \mathbf{P} , we can define now a modified autoassociative neural network in $\mathcal{L}(V)$, rather than \mathbf{R}^n , as follows. The neural network consists of just two layers with a number of neurons equal to the dimensionality of $\mathcal{L}(V)$ (m) and weight matrix given by \mathbf{P} and the pseudo-inverse matrix $(\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T$, respectively. The activation function H in the first layer is defined as before; namely, $H(x_i) = \delta_{i, \min\{\max\{j | x_j = \max\{x_j\}\}}$, which produces one single neuron to fire.

Doubtless, working with the pseudo-inverse matrix has an important theoretical value, but from a practical point of view, in doing so, we go back to the main problem of weight matrix size posed in the previous section. Therefore, from a practical point of view, we consider only the first layer as the output layer as well, where the output pattern gives the components of some $\alpha_k \in V$, which can be identified through the index of the firing neuron.

Consider again the previous example, and assume that the noisy pattern $\alpha = (1, 4, 3, 4)$ needs to be recalled. Since the weight matrix is \mathbf{P} , the components in basis V are the following

$$\begin{pmatrix} -\frac{1}{3} & \frac{2}{3} & 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \end{pmatrix} \begin{pmatrix} 1 \\ 4 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

and, after applying the activation function, we obtain the output pattern $(0, 1)$, which means that the neural network associates α with α_1 .

Since the fact of working with a set W of orthogonal patterns plays an important rôle in what we have made, it is convenient to prove that this set indeed exists. We do this by proving the following theorem

THEOREM 4.1. *Given a set V of m linearly independent vectors and a linear manifold $\mathcal{L}_k(V)$ generated by $V \setminus \{\alpha_k\}$, $V = \{\alpha_0, \dots, \alpha_{m-1}\}$ and $\mathcal{L}_k(V) = \sum_{i \neq k}^{m-1} \lambda_i \alpha_i$, where $\alpha_i \in \mathbf{R}^n$, $2 \leq m \leq n$ and $\lambda_i \in \mathbf{R}$, $\forall k \in \{0, 1, \dots, m-1\}$ there does exist a vector β_k which is orthogonal to $\mathcal{L}_k(V)$.*

Proof.

Suppose that $\mathcal{L}(V)$ is the m - dimensional linear manifold generated by V , since $m \leq n$, $\mathcal{L}(V) \subseteq \mathbf{R}^n$. If $\alpha \in \mathcal{L}(V)$, then α can be written as a linear combination of vectors in V : $\alpha = \sum_{i=0}^{m-1} \lambda_i \alpha_i$. In other words, α can be written as a combination of α_k , for some k , and a vector in $\mathcal{L}_k(V)$:

$$\alpha = \lambda_k \alpha_k + \sum_{i \neq k}^{m-1} \lambda_i \alpha_i.$$

If $\mathbf{x} = \sum_{i \neq k}^{m-1} \lambda_i \alpha_i$, then $\mathbf{x} \in \mathcal{L}_k(V)$ can be written as a matrix multiplication $\mathbf{x}^T = \mathbf{A}\boldsymbol{\lambda}^T$, where \mathbf{A} is an $n \times (m-1)$ matrix and $\boldsymbol{\lambda}$ an $1 \times (m-1)$ vector such that

$$\mathbf{A} = [\alpha_0^T, \dots, \alpha_{k-1}^T, \alpha_{k+1}^T, \dots, \alpha_{m-1}^T]$$

and

$$\boldsymbol{\lambda} = (\lambda_0, \dots, \lambda_{k-1}, \lambda_{k+1}, \dots, \lambda_{m-1}).$$

Notice that the expression $\mathbf{x}^T = \mathbf{A}\boldsymbol{\lambda}^T$ gives the transformation of components from $\mathcal{L}_k(V)$ space to \mathbf{R}^n ; however, through some elementary row operations to be defined later we can obtain the opposite. First of all, since $m \leq n$, then $m-1 < n$ and this means that the row rank of \mathbf{A} is smaller than the number of rows in \mathbf{A} . The following elementary row operations reduce the number of non-null rows in \mathbf{A} and produce a linear system equivalent to $\mathbf{x}^T = \mathbf{A}\boldsymbol{\lambda}^T$ [Hoffman 1971]:

- (1) Substitution of row i by k times row i , where $k \neq 0$,
- (2) Swapping row i and j , and
- (3) Substitution of row i by itself plus k times row j , where $k \neq 0$.

After a finite number of elementary row operations on the augmented matrix $[\mathbf{A}, \mathbf{x}^T]$ we can obtain 0's in the last row of \mathbf{A} , but a linear function $f(\mathbf{x})$ in the

column corresponding to \mathbf{x}^T , which means that $f(\mathbf{x}) = 0$. This equation defines the linear manifold $\mathcal{L}(V)$ where the coefficients of the variables in \mathbf{x} define the orthogonal vector to $\mathcal{L}(V)$.

5. CONCLUSIONS

The simple way of getting rid of the cross-talk term in an autoassociative neural network, with continuous-valued input signal, through the inclusion of a non-linear operation on the input pattern (a competitive layer) as shown in Section 3, has critical consequences when the computers memory and speed's memory are not appropriate, or the number and dimension of the input patterns are bigger enough.

This problem can be reduced in a ratio $m : n$, where m is the dimension of the subspace $\mathcal{L}(V)$ of \mathbf{R}^n , when the competitive and associative parts are embedded into one-single layer neural network at the expense of increasing the complexity in the calculations of the weight matrix. However, this complexity can be reduced by mean of a computer implementation of the process followed to prove Theorem 4.1. Since, from this point of view, the proof of Theorem 4.1 not only proves that the set W of orthogonal vectors exists, but also shows one process of construction.

REFERENCES

- HARYONO, R. S. AND PHIEN, H. N. 1994. Properties of non-orthogonal input pattern pairs in bidirectional associative memory and strategies for their successful recall operation. In *Proceedings of the Second International Conference on Expert Systems for Development*. IEEE Computer Society Press, Los Alamitos, California, 148–152.
- HAYKIN, S. 1994. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, Upper Saddle River, New Jersey.
- HOFFMAN, K. 1971. *Linear Algebra*. Prentice-Hall, Upper Saddle River, New Jersey.
- KOSKO, B. 1988. Bidirectional associative memories. *IEEE Transactions on Systems, Man, and Cybernetics* 18, 1 (Jan.), 49–60.
- ZURADA, J. M. 1992. *Introduction to Artificial Neural Systems*. West, St. Paul, Minnesota.

July/25/2002