

Stabilizing Bidirectional Associative Memory with Principles in Independent Component Analysis and Nullspace (PICANS)

James P. LaRue

james@jadcosignals.com

Yuriy Luzanov

Yuriy.luzanov@rl.af.mil

ABSTRACT

A new extension to the way in which the Bidirectional Associative Memory (BAM) algorithms are implemented is presented here. We will show that by utilizing the singular value decomposition (SVD) and integrating principles of independent component analysis (ICA) into the nullspace (NS) we have created a novel approach to mitigating spurious attractors. We demonstrate this with two applications. The first application utilizes a one-layer association while the second application is modeled after the several hierarchal associations of ventral pathways. The first application will detail the way in which we manage the associations in terms of matrices. The second application will take what we have learned from the first example and apply it to a cascade of a convolutional neural network (CNN) and perceptron this being our signal processing model of the ventral pathways, i.e., visual systems.

Section 1: Introduction

After over 30 years of research there is still a long way to go for computer vision systems to be able to achieve reliable performance in scene analysis and object categorization and identification in unconstrained video and imagery data. At the same time a lot of progress has been made in mapping the functionality of biological vision systems, which are very robust at tasks with which computer vision struggles.

Several models have been developed over the years to describe the information processing that takes place in the visual systems of mammals, with a lot of focus on primates including humans. Evidence from psychophysical and physiological studies points out a hierarchical organization of the visual system. In the ventral stream starting with V1 through V2, V4 and up to IT and PFC, a gradual increase in the complexity of the stimulus required to elicit neuronal response was observed. Several feed-forward computational models of the ventral stream have been formulated notably the HMAX¹ design and the convolutional neural network (CNN)² model. However, these models have not been successfully applied as feedback processors due to the lack of data available to model the feedback paths from higher to lower areas in the visual system. To address this shortcoming we have utilized bidirectional associative memory (BAM) to model both the bottom-up and top-down influences of higher areas of the brain on feature selectivity in the ventral pathway.

BAM is an extension to the early 1980's independent research with unidirectional autoassociators^{3,4}, in that BAM includes feedback in addition to feed forward processing. Bart Kosko, who had spoken with Stephen Grossberg regarding the autoassociators and who also knew of Kohonen's earlier research with correlation matrices⁵, reintroduced the idea of BAM in matrix form in the later part of the decade⁶. He demonstrated the success of the algorithm but warned of the problems involving inversion and stability. These warnings were reiterated in more recent research⁷.

The goal of our AFOSR project was to model feature-based attention through association of features at different stages of a hierarchical model with higher level representation of objects. The envisioned end-state was to better suppress the features irrelevant to the current categorization/identification process and negate effects of clutter in the stimulus image. In our second application the CNN is our choice for modeling the hierarchical neural response and will be used as the basis for the feed-forward processing. The reason for this choice is due to its historical association with the MNIST Data⁸. The purpose of this paper is to offer an innovative extension to the way in which the Bidirectional Associative Memory (BAM) should be implemented in practice and in general.

Most of the work presented in this paper is in the area of basic Research and Development (R&D) and was performed in 2011-2012 at the Air Force Research Laboratory (AFRL) Information Directorate in Rome, NY and was sponsored by the Air Force Office of Scientific Research. The remaining work was performed at the DARPA/GMU Innovation House program where there was more emphasis on immediate transition into the commercial sector. The overall paper is broken into five sections. This is the first section – introduction. The second section will describe how a BAM estimator is formed which should explain why it is intuitively referred to as The Additive Model³. The third section will show how we managed the matrix methods available to us to make improvements to the estimator. The fourth section will show how a BAM estimator is formed from the layers of the CNN which we hold to be a bio-inspired model of the ventral pathways. Finally, section five will provide summary and conclusions.

In the end we want to emphasize that our solution addresses the 30 year old problem of stability with BAM. We will show that by transforming the output vectors of a hierarchical neural network into an associative memory matrix and then by analyzing the matrices utilizing the singular value decomposition (SVD) in conjunction with

independent component analysis (ICA) with a watchful eye on particular nullspace (NS) vectors, convergence to the expected state can be made more robust.

Section 2: Kosko Example

In chapter three of Kosko's book, *Neural Networks and Fuzzy systems*, he describes how to construct BAM connection matrices. In summary, one has a set of associations in mind. An association is between two feature vectors, not necessarily the same size and not necessarily in binary form. For each association a matrix is formed by taking the outer product of the two vectors. Thus for N associations there are N matrices. These N matrices are then added together under normal matrix algebra rules to form the final matrix, hence the alternate reference to this model as being the Additive Model.

The forward direction starts with multiplying the matrix with a feature vector to create an output (associated feature) vector. The backward direction is evoked by applying the output vector to the transpose of the matrix which in turn forms another vector. This bidirectional process is meant to be repeated several times before convergence occurs albeit it may be to something either expected or not expected. Terms like adaptive resonance or attractor are used when a feature converges to itself and its associated vector while terms like spurious attractors or unintended fixed points are used when the vector does not converge into a desired associated pair.

Having described Kosko's process we turn to our first example. From the MNIST data base we collected 863 sets of the numbers 1 through 8. For this example we will focus on set number 196. A threshold was applied to convert all pixel values into a binary state. Figure 1 shows the resulting 28 x 28 pixel images. Figure 2 shows how the images appear after each of these images was unwrapped into vectors of length 784. We will refer to these vectors as $X_1, X_2, \dots, X_8$ as needed but they directly correspond to the MNIST numbers 1, 2, ..., 8. Four associations were established linking 1-2, 3-4, 5-6, and 7-8 through their outer products yielding four 784 x 784 association matrices. The four minor matrices were added together to form the final additive matrix M . Figures 3 shows the four individual matrices on the left and the composite (additive) matrix on the right.

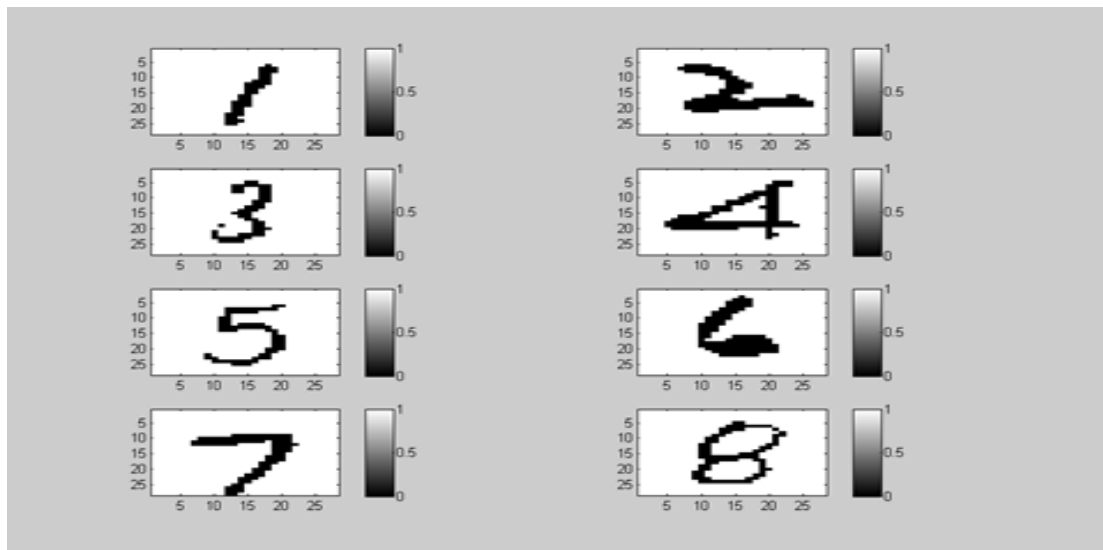


Figure 1: Binary MNIST dataset images, result of thresholding

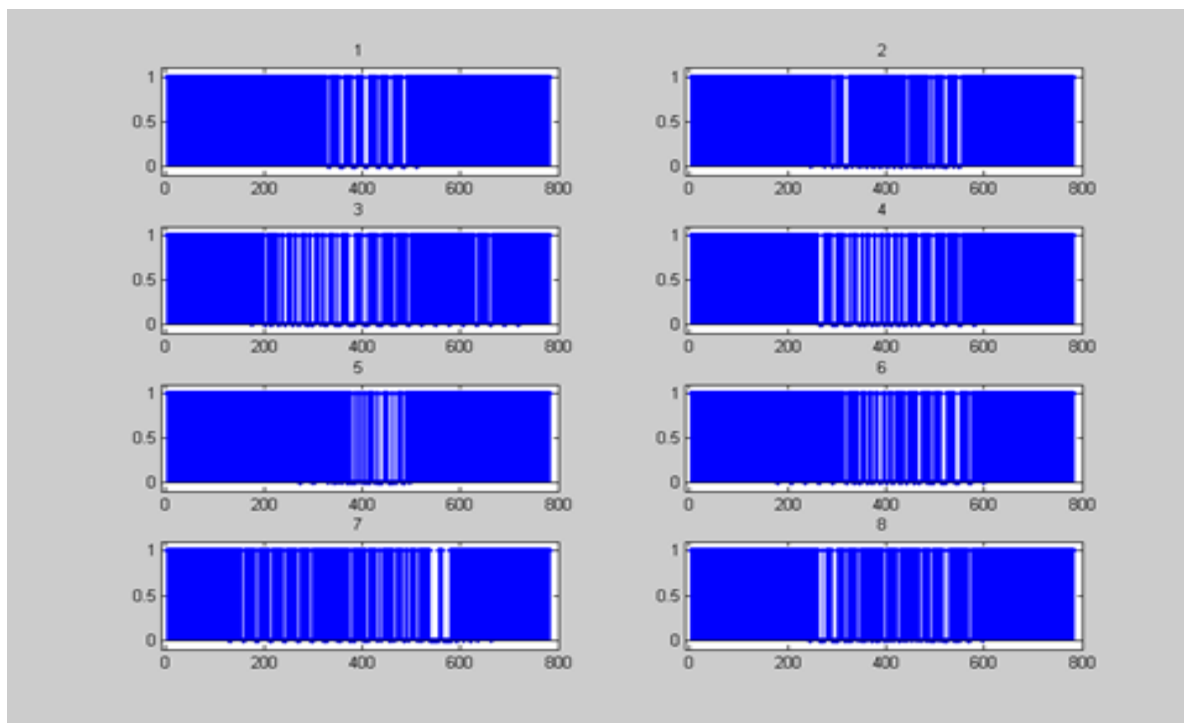


Figure 2: Binary images of '1' to '8' unwrapped into vectors

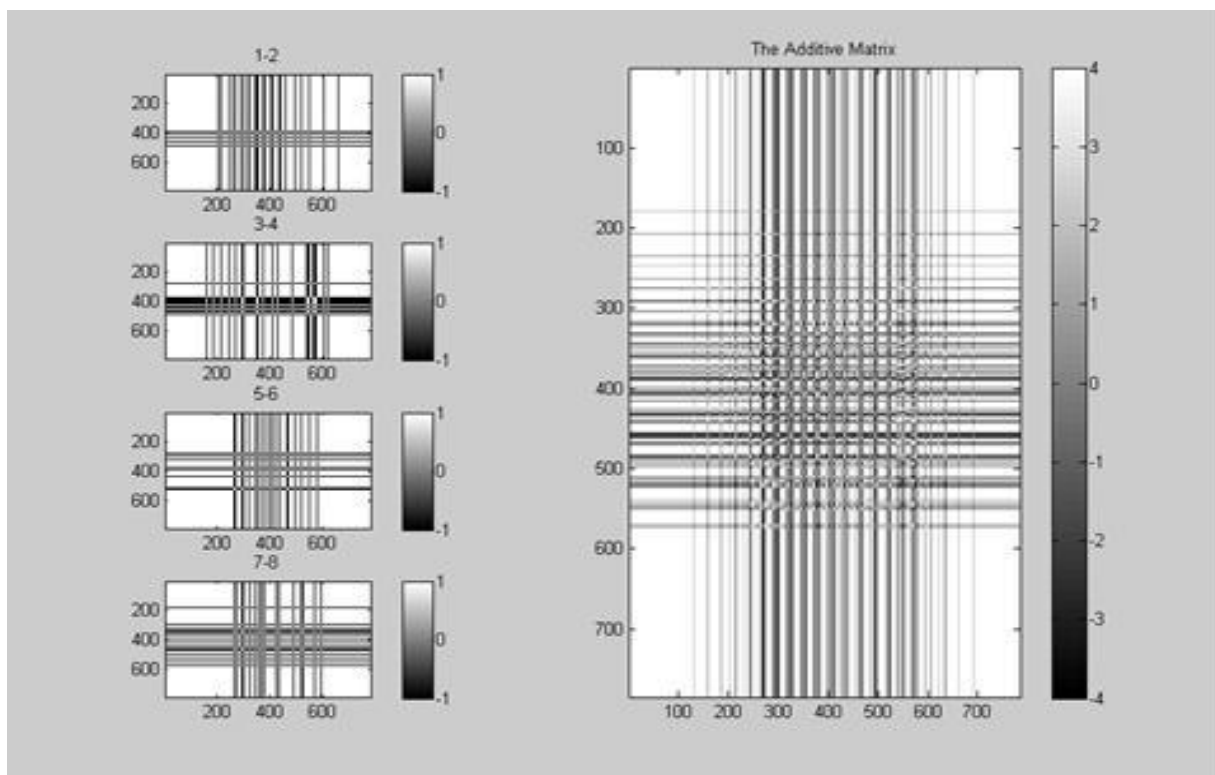


Figure 3: Associations formed through outer products of input/output pairs

If the minor association matrices were built from components that were orthogonal to each other then it would be trivial to apply X_1 to the input and receive X_2 at the output and vice-versa. That is, given,

$$M = X_1X_2^T + X_3X_4^T + X_5X_6^T + X_7X_8^T, \quad [1]$$

$$\text{then } X_1^T M = X_2^T, \quad [2]$$

since the other terms vanish.

$$\text{Similarly, } X_2^T M^T = X_1^T. \quad [3]$$

Hence, if orthogonality was assumed we would expect an input of '1' in vector form to result in an output of '2' in vector form. We would only need to reshape the output into a 28 x 28 matrix to see images in Figure 1. Note that the X_1 vector is transformed by M while the X_2 vector is transformed by M^T due to the way we constructed M . We define an iteration to be a single loop involving M and M^T .

Since orthogonality is not assumed, then only through multiple iterations may one find a steady state in which an input vector finds an associated pairing. In actual experimentation with all 863 sets we found very few examples where the number of iterations required for convergence to a steady state to exceed three iterations. However, as noted above, the steady state may not be the desired state. Hence the focus in this paper is to show how our novel approach may mitigate the case of convergence to something other than the expected pairing.

In examples given by Kosko⁶ and later by Chartier & Boukadoum⁷, convergence to an expected association occurred in many cases. However, in testing with the MNIST data base⁸ this was found to be the exception rather than the rule; emphasizing the strict adherence to the construction and iteration process. As the previous researchers have alluded to in their research there is an issue with thresh-holding the output vector after it is transformed with either M or M^T .

The amplitude of an output vector for our MNIST example can grow to exceed the computer's allotted memory in short time. Thus, we follow Kosko's lead and chose our threshold at zero where anything greater than zero was set to one and anything less than or equal to zero was set to 0.

Let us quickly review again Figures 1-3. Figure 1 shows the numerals as an image. Their pixel values are either 1 or 0 as indicated in the color-scale. Figure 2 shows a stem plot of the unwrapped images with the same amplitudes. The amplitudes change as we look at Figure 3. The minor matrices on the left have amplitudes 1 or -1. This transformation from 1/0 to 1/-1 is a Kosko suggestion since multiplication in the 1/0 state admits one way to reproduce a 1 but three ways to produce a 0. In the contrasting 1/-1 state the number of ways to return a 1 or -1 are split evenly. Thus using Kosko's suggestion the additive matrix on the right ranges from -4 to 4. Once the process begins the matrix values do not change. The vectors that pass through the M or M^T matrix can take on large values but they are reassigned to either a (threshold dependent) 1 or 0. Note that in Figures 1 and 3 the color-scale appears as a grayscale but the values in the plots are discrete integers.

The main result of this paper is discussed in the next section. Before we do that we wish to discuss the next three figures which show results using three variations of the constructed M matrix. In most of the remaining figures in this paper a color-scale has been added. We purposely have shifted between grayscale and color-scale for both clarity and attention to detail in the accuracy of what we are showing.

We now discuss Figures 4, 5, and 6. These three figures show results using three variations of the constructed M matrix; we will save the way in which these matrices were derived for the next section. We have included a grayscale yet keep in mind that at this point the pixels values are 1/0 in value. In each figure there are four rows of plots. The first and third rows of each figure are the expected numeral pairs based on our construction while the second and fourth rows are the actual outputs after passing through enough matrix transformations to cause a steady state convergence to, albeit, not always the intended pair

For purposes of maintaining clarity for the reader let us return briefly to Figures 2 and 3. These stem plots represent the unwrapped images in Figure 1. To start our iterative process we took as our initial input vector one of the vectors on the left. For example, consider the top left image in Figure 1. Its corresponding vector form is the top left stem plot in Figure 2. This vector is applied iteratively to the additive matrix as shown on the right side of Figure 3 and described in equations [1] and [2] above. With each iteration we return an estimate of some association. The closely timed but independent argument by Cohen-Grossberg and Hopfield is that this input vector should 'resonate' with the additive matrix to produce the associated pair.

Now let us turn our attention to the first column in Figure 4. Recall that the first and third rows are the desired pairs. Hence the images of '1' and '2' in the first and third rows correspond to those in Figure 1. The images in rows 2 and 4 of Figure 4 are the estimates. Our very own ventral pathways in the Lateral Occipital Complex allow us to see the close correspondence despite the differences between the expected and actual 1-2 pairs. Turning to the 3-4 pair we see that '3' was estimated well but that there is a mixture of numerals in the '4' estimate. Finally, with respect to the 5-6 and 7-8 pairs, both 4th row estimates are reasonable but both 2nd row estimates have estimated what appears to be a '3'.

Figures 5 and 6 are nearly identical at first review. Each Figure shows that the estimates of the 1-2, 5-6, and 7-8 associations are very close, and in fact are perfect with respect to the desired pairs. Upon closer inspection one sees that both 3-4 pair estimates are flawed. In Figure 5 the 3-4 estimate appears to be a flawed 1-2 pair while in Figure 6 the '3' estimate has minor flaws and the '4' estimate appears to be mixture of (possibly) '2' and '4'. The estimates arrived at in Figures 5 and 6 were born from a treatment to a Kosko formulated additive M matrix. Figure 5 used a matrix variation we will refer to as M14 and Figure 6 used a matrix variation we will refer to as M18. The reason for this naming convention will become clear in the next section.

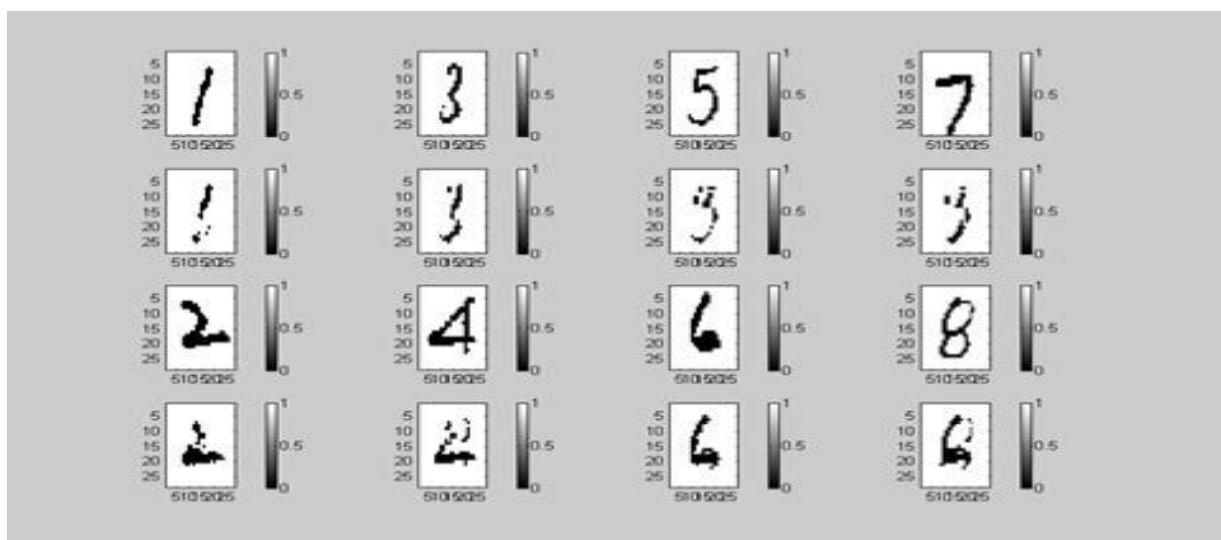


Figure 4: Results of BAM associations using Kosko formulated additive M matrix

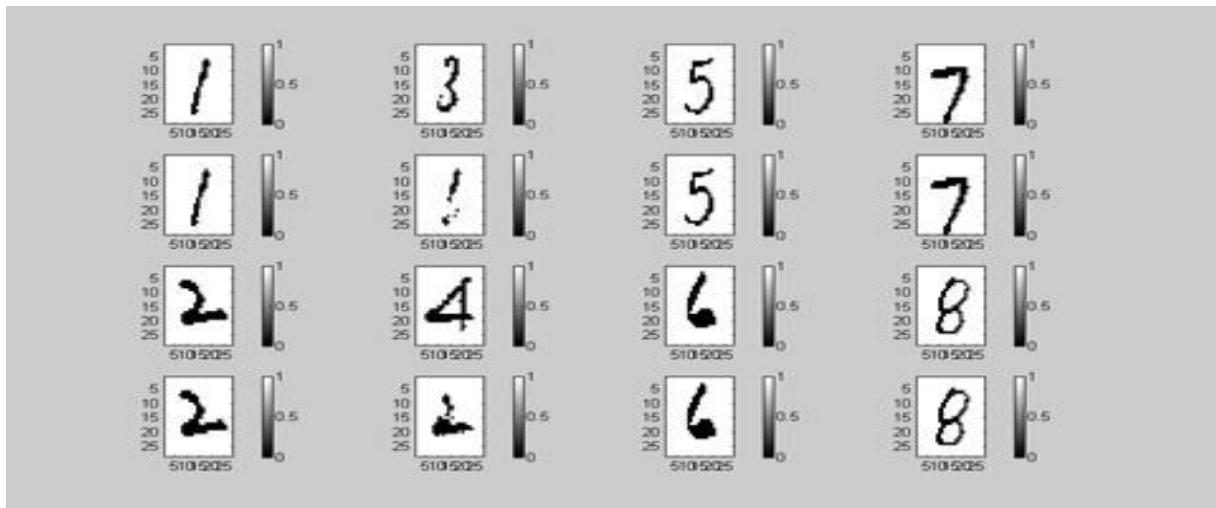


Figure 5: Results of BAM associations using a modified M14 matrix

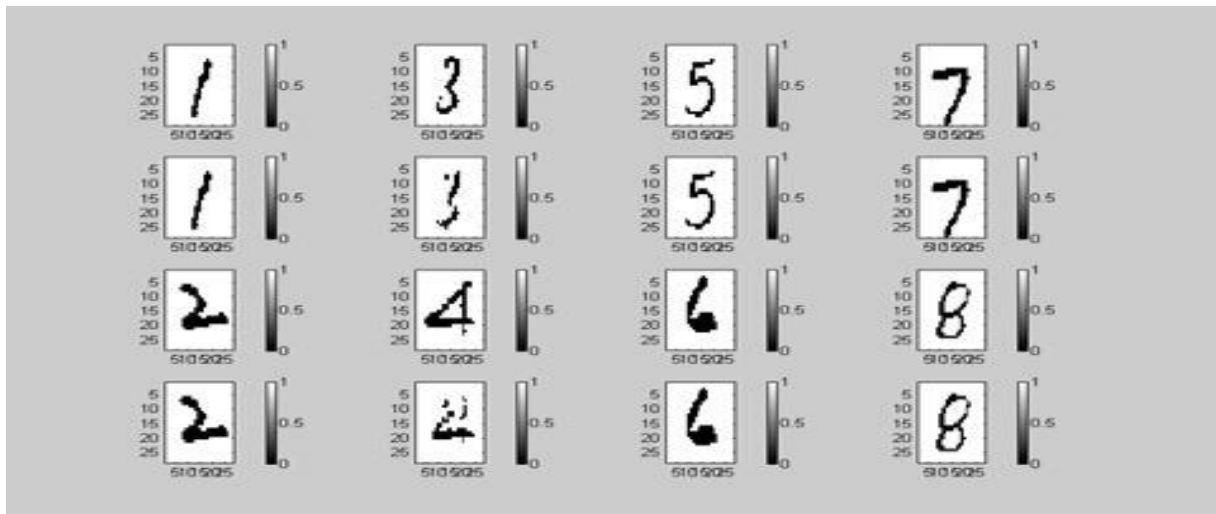


Figure 6: Results of BAM associations using a modified M18 matrix

Section 3: PICANS and Managing Neural Outer Products in Bidirectional Associative Memory

Independent Component Analysis (ICA) is an extension of Principal Component Analysis (PCA). Our motivation to incorporate this approach was based on the experienced success in a specific application to additive (mixture) matrices encountered in digital modulation schemes. A possibly better known context in which to visualize this problem comes from the so-called 'cocktail party problem'. In this problem several people are speaking simultaneously and the goal is to separate the speakers. In general this can be best achieved when the number of speakers is matched by the number of microphones and that the number of speakers is known. In our case we can equate the speakers and microphones as an additive mixture of a known number of associations. While the ICA

algorithm is generally thought of as a two-step process; the first step whitens the data and the second step estimates mixture matrix; at this point we have only utilized the first step of whitening, saving the estimate of the mixture matrix for a later time.

The ICA whitening step requires that the matrix under consideration be broken down into either an eigenvalue decomposition (EVD) or a singular-value decomposition (SVD). Since our matrix is not symmetric and we are interested in bidirectionality we used the SVD. Also, in general, the associations that can be formed can lead to a rectangular matrix which eliminates using the EVD.

In signal processing, white noise in the time domain is, for better or for worse, associated with a constant (or flat) response in the frequency domain. In the context of ICA, whitening similarly means the flattening of the principal singular values in the SVD. The result of this flattening is the equating of the associated left and right singular vector structures. The SVD process produces singular vectors that are orthogonal to each other and are normalized to one. It is the singular values that set the individual vectors apart with respect to their individual affect to the reconstruction of the additive matrix. That is, given M , the SVD develops two matrices of singular vectors where the left and right vectors which are scaled by the corresponding singular values in a third matrix. In classical SVD form we write $M=USV^T$ where U and V are the sets of left and right singular vectors and S is the diagonal matrix of (scaling) singular values. We will refer to the left singular vectors by the notation U_i , the singular values by S_i , and the right singular vectors by V_i .

In Figure 7 we show the additive M matrix from Figure 3, its first 20 singular values S_1-S_{20} , and its first four each of left and right singular vectors. The only difference between the two figures is the choice of color bar. In Figure 7 we choose to utilize a color scale instead of gray scale for reasons of consistency and clarity as we proceed with the discussion.

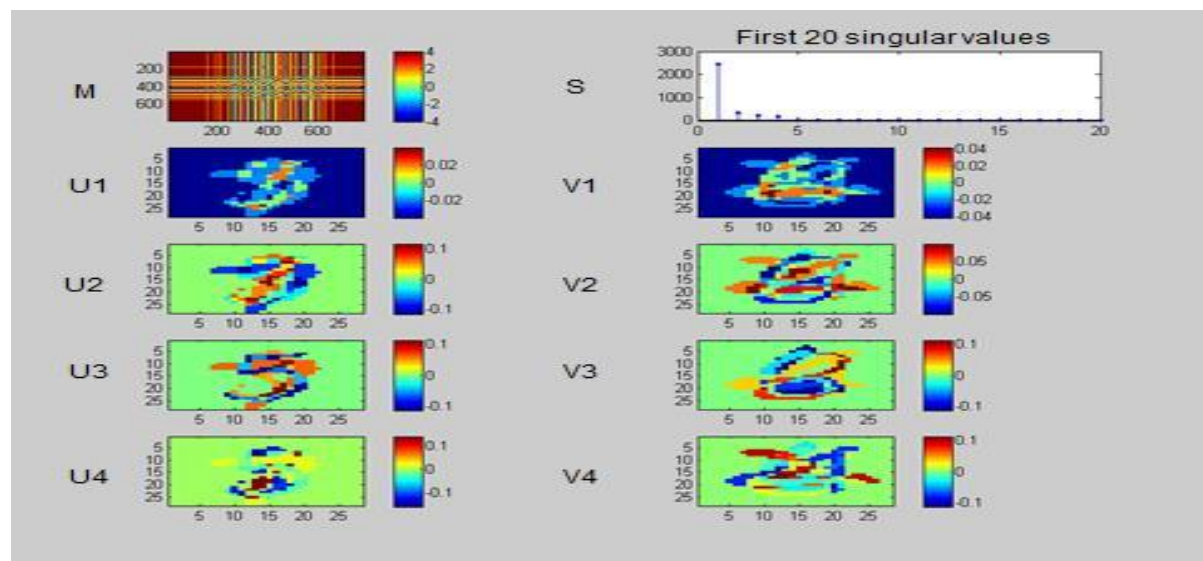


Figure 7: Analysis of the principal components of the association matrix M

The top right plot is a stem plot of the singular values, which by definition are positive numbers. The first singular value is assigned a value near 250. The next three singular values are small in comparison and are on the order of amplitude 50. The remaining 16 singular values have amplitude on the order of $1e-11$. In most cases where compression techniques are used these 16 singular values and their associated vectors would be ignored. Indeed, the four non-zero singular values (S_1-S_4) indicate that M has a rank of four. This makes sense since we have four

associations that made up this matrix. Since the dimensions of the matrix are 784 x 784, this means there are 780 sets of triplets of a left vector, right vector, and scalar that can be eliminated. From a compression standpoint one may say there are four principal components and 780 nullspace components each from the U and V matrices.

In a compression algorithm that is looking to simply reconstruct the matrix rather than use it in a bidirectional iterative fashion there is no problem with this approach. However, in our adaptive resonance setting, leveling out the playing field, so to speak, through whitening, has made the difference between Figures 4 and 5. Also, although ICA and PCA in general focus on the largest singular values, we will show that including small singular values can make the difference between Figures 5 and 6.

Continuing with Figure 7 the plots in rows 2-5 represent some of the 784 x 1 dimensional singular vectors reshaped into a 28 x 28 dimensional matrix as was done in Figures 1 and 2. Regarding the second row, the image on the left is the 28 x 28 reshaping of the left singular vector corresponding to the largest eigenvalue while the image on the right is its corresponding right singular vector, also reshaped. Jumping ahead to Figure 11 the top left stem plot is this left singular vector before being reshaped and shown in Figure 7.

The next three rows of plots correspond to the vectors associated with the next three largest singular values. There are many observations to be made. The color-scale is becoming more important to the understanding of what the SVD does and what it does not. It does find four pairs of vectors (four because the rank of the matrix is calculated to be four) which when associated through the outer products and scaled by the singular vectors in the top right stem plot reconstructs the top left matrix M. In mathematical notation

$$\sum_{i=1}^4 U_i S_i V_i^T \approx M \quad [4]$$

It does not separate out the pairs 1-2, 3-4, 5-6, and 7-8. It would only do so if Eqs. [2] and [3] held true for all pairs. Instead, the U_i and V_i images in Figure 7 show that each principal component upon being reshaped into a 2-D image reveals an overlapping of the numbers 1,3,5, and 7 in each of the figures on the left and the numbers 2,4,6, and 8 in each figure on the right. Thus any of the stem plots in Figure 11 are normalized mixtures of the plots in Figure 2.

We chose to reprint the M matrix with a color map to contrast it against the reshaped singular vectors. As a reminder, the M plots are determined from a set of discrete integers ranging from -4 to 4. We count over 20 distinct numbers for any of the principal components. Hence the SVD does return a set of components which are constrained to be orthonormal and since Eqs. [2] and [3] would hold in this case we can conclude that these components must be mixtures.

But what if we look at the next four singular values? Why would we look at the next four singular values? We stated that the rank of this matrix is four. Compression algorithms rely on a rank estimation algorithm to indicate what the rank of a matrix is in order to only include those vectors deemed significant; any other vectors are deemed part of the nullspace and viewed as insignificant. And as we stated above, that's fine if the task is to reconstruct the original matrix, but it may be a little naïve to ignore the non-significant singular values which are paired with what is consequently thought of as insignificant singular vectors. However, these vectors are still viable modes, and when we use the matrix in this iterative fashion the BAM process can and does resonate with some of these seemingly insignificant modes.

Figures 8 and 9 are involved in our discussion of the U_i and V_i components, for $i=5:8$. Figure 8a shows the M matrix. Figure 8b focuses on the singular values from 5 through 20. Observe that there are several values above 0 however the y-axis does indicate that the values are on the order of $1e-11$. The corresponding sets of reshaped left and right

vectors are plotted in Figure 8 with $U_5 - U_8$ on the left and $V_5 - V_8$ on the right. A quick observation of these images reveals a faint mixture in each similar to that seen in Figure 7. Upon closer inspection we notice the range of the color-scale scale to be much greater than the ones in Figure 7. But if we look at the first column we observe some darker pixels which are an artifact of the normalization constraint. In Figure 9 we show on the left a 28x28 image based on the reshaped V_8 . The image on the right is arrived at by eliminating the first column of the reshaped V_8 , thus reconfiguring it into a 28x27 matrix. Notice the scale has changed and the structure of the mixture is more apparent. Now, do we need this and other insignificant components to 'reconstruct' the matrix M ? The answer is 'no'. However, do we need these components in a BAMs environment? The answer is 'sometimes'.

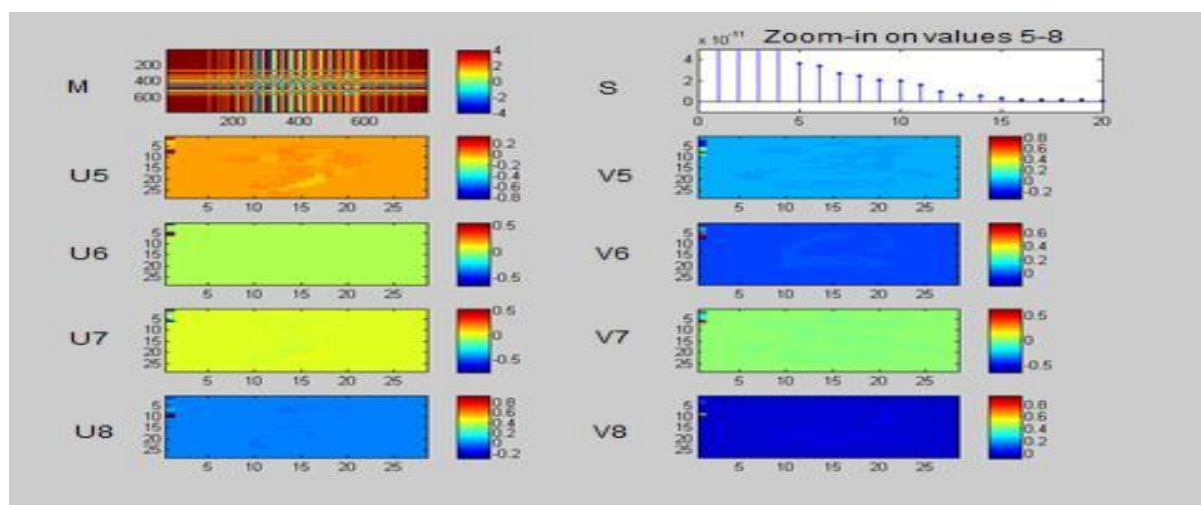


Figure 8: Analysis of the null space of the association matrix M

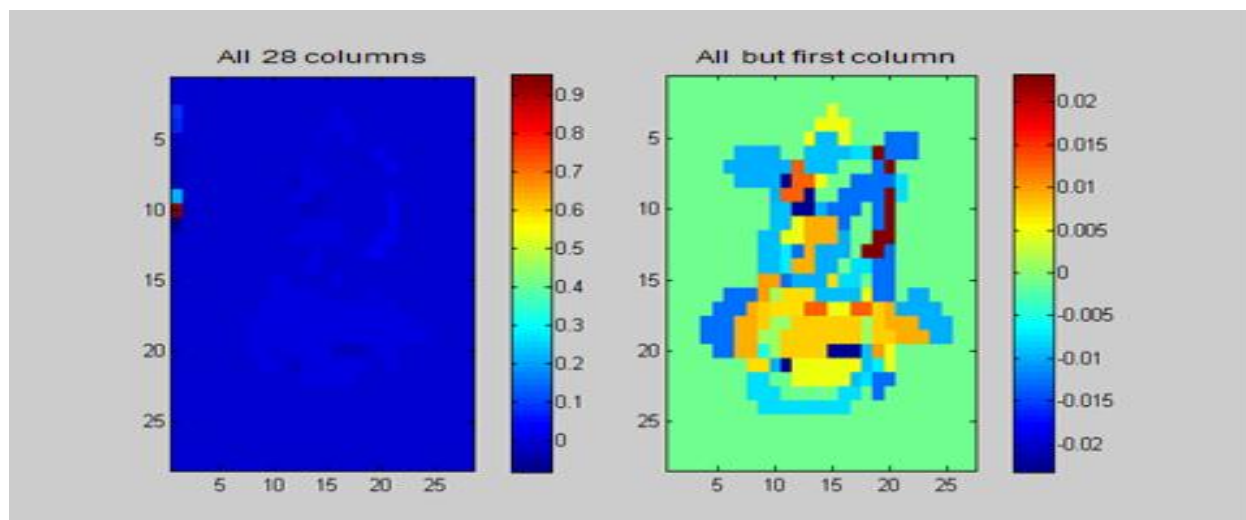


Figure 9: Contribution of the U_8 and U_8 singular vector components to the association matrix M

We are coming to the end of this section. We have shown how we constructed the M matrix. We have shown what the first four principal components look like along with their singular values and we have done the same with the next four 'insignificant' principal components and singular values. The ICA approach relies on the whitening of a

number of singular values; that number being equal to the estimated rank of M. Our M14 matrix was formed from the first four components. The four singular values were reassigned to the value of 1. Thus

$$M14 = \sum_{i=1}^4 U_i V_i^T \quad [5]$$

Now, each of these vectors are orthogonal and already normalized to one, hence, applying the SVD to M14 results in four principal components with four equal singular values as should be expected.

Our M18 matrix was formed from the first eight components. The eight singular values were reassigned to the value of 1. Thus

$$M18 = \sum_{i=1}^8 U_i V_i^T \quad [6]$$

Similarly, each of these vectors are orthogonal and normalized to one, hence, applying the SVD to M18 results in eight principal components with eight equal singular values, again, as should be expected.

Figure 10 shows the original matrix M, and the two reconstructed matrices M14 and M18 on the left along with their corresponding singular values on the right. Thus we can compare the original unequal singular values from M, the four equalized singular values from M14, and the eight equalized singular values from M18. Of course the images of M14 and M18 are scaled differently. For completeness we have plotted in Figure 11 the first eight U_i vectors from each of the three SVD's. Figure 12 shows the same vectors but looks at the components from 200 to 600 only in order to show the underlying structures.

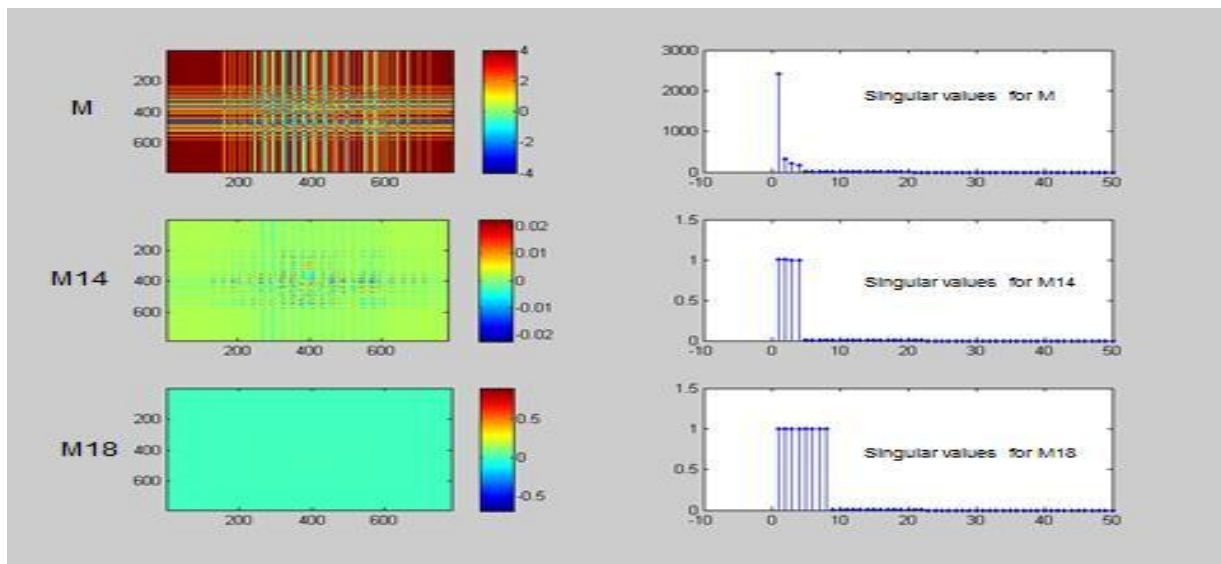


Figure 10: Comparison of the original association matrix M and modified matrices M14 and M18

We utilized the three sets of vectors in Figure 11 to produce the BAM results shown in Figures 4, 5, and 6. As observed Figure 4 has some success, Figure 5 is a dramatic improvement over Figure 4, and Figure 6 is a slightly better improvement over Figure 5. However, we did test all 863 sets from MNIST and determined that the most consistent BAM estimator was based on M14 and not M18. We give the name PICANS to this approach to reflect the use of principal component analysis through the SVD along with independent component analysis and nullspace analysis.

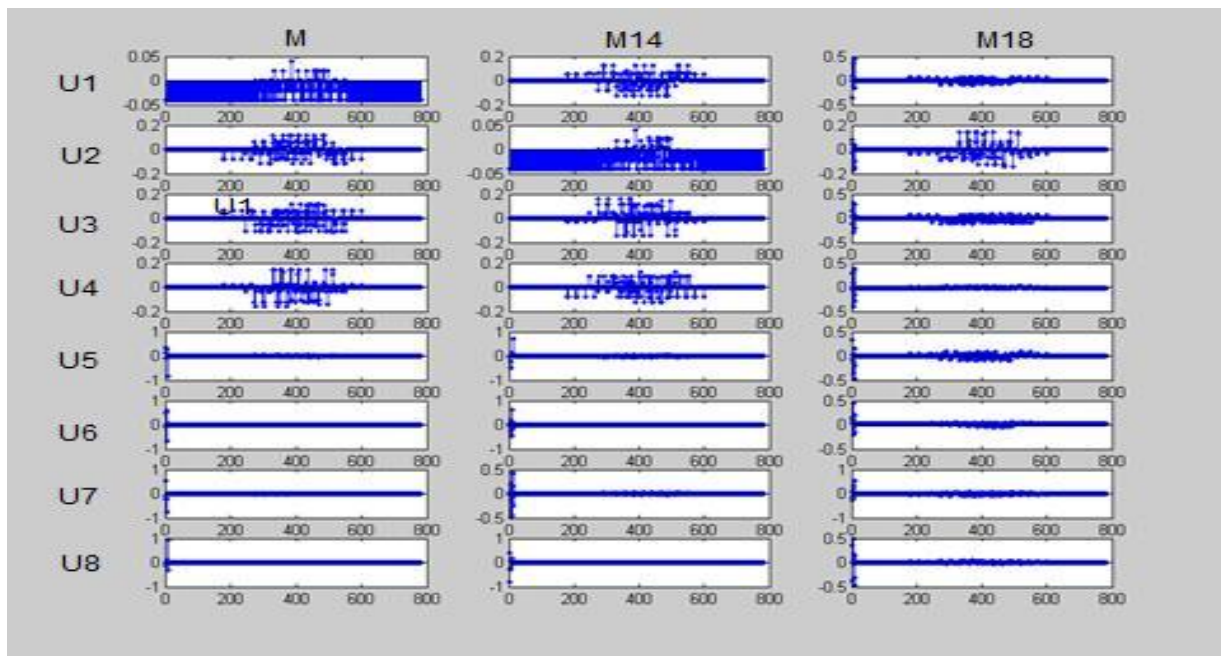


Figure 11: Comparison of the first eight U singular vectors of each of the three association matrices

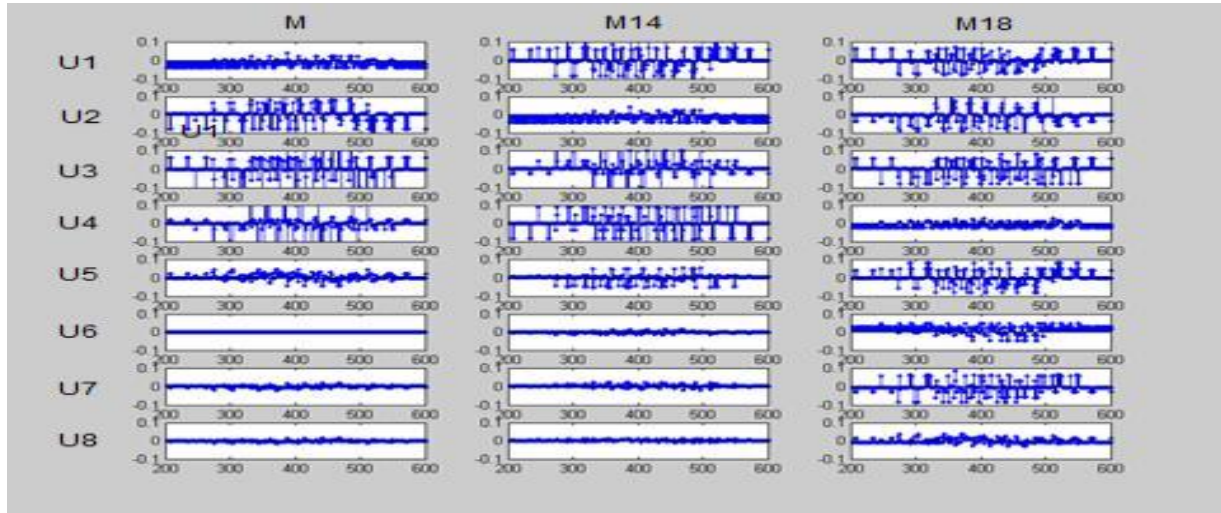


Figure 12: Comparison of the first 8 V singular vectors of each of the three association matrices

These results, if the reader follows our procedure, can be readily reproduced. The point of this paper is to show that by utilizing the whitening step from the ICA approach BAM is a more robust tool. Clearly, there is a lot of work to do; clearly, least squares and assuming Gaussian distributions, for this application of the SVD, is not optimal for BAM. Also, since this application involves resonating modes, one should pay careful attention to the seemingly insignificant ones; one has only to recall the analogy of the Tacoma Narrows Bridge collapse.

Section 4: BAM Applied to Ventral Pathways

In contrast to the single layer example given in section three, section four will demonstrate how to modulate the activity of several contiguous layers. The CNN and the BAM will act as co-modulators in the context of a biologically inspired classification algorithm which models the multiple hierarchical levels associated with the ventral pathways. As was first mentioned in our introduction many biologically inspired models have not been successfully applied as feedback processors. This section asserts that with the integration of BAM into a feed-forward only hierarchical construct like the CNN the viability of introducing feedback can be realized.

To form the intermediate sets of vectors from which we would make the associations we used the CNN. The CNN was designed to produce the feature vectors for LGN, V1, V2, V4, IT, PFC where these labels correspond to the ventral pathways. Figure 13 depicts this design. It is from these six sets of vectors we form the five sets of BAM matrices. BAM 1 interconnects the input LGN and V1, BAM 2 interconnects V1 and V2, BAM 3 interconnects V2 and V4, BAM 4 interconnects V4 and IT, and BAM 5 interconnects IT to the PFC decision.

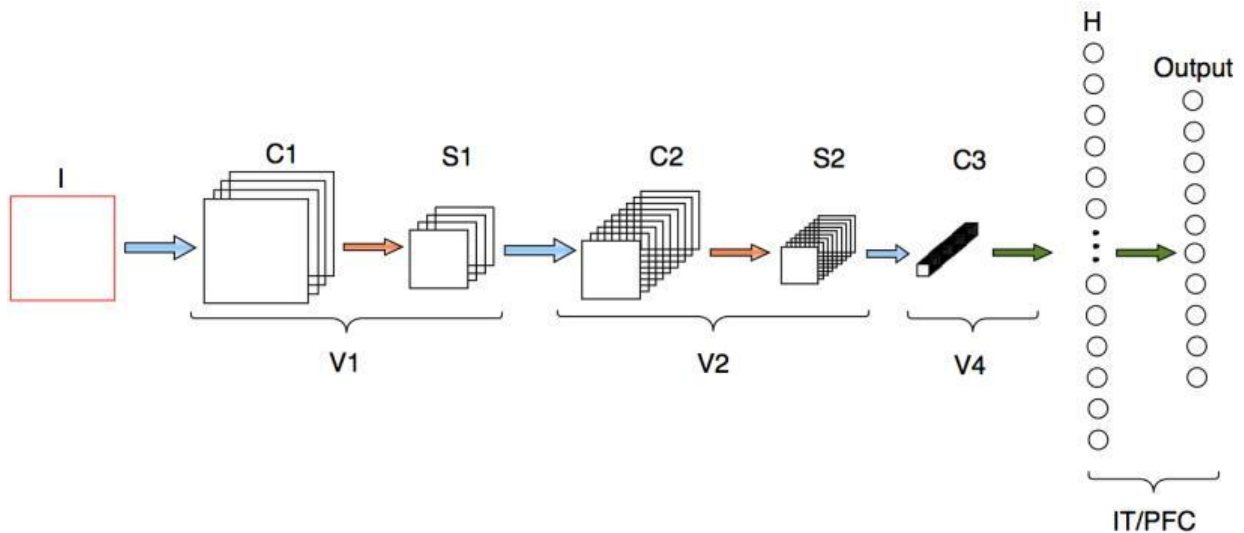


Figure 13: Architecture of the CNN/Perceptron system

To train and test the system we again used the MNIST data set. In contrast to the Kosko example where we associated the number '1' with the number '2', here we associated each of the numbers 0-9 with the corresponding columns of the identity matrix. That is, the input image of 0 was associated to the output decision vector of [1 0 0 0 0 0 0 0 0], while the input image of 1 was associated to the output decision vector of [0 1 0 0 0 0 0 0 0] and so on.

To gather a representation for each of six ventral pathway feature vectors we ran a CNN training algorithm for each number 0-9. We took the results from each of the six layers to form the necessary sets of vectors to form the five contiguous BAM matrices. The lengths of the vectors are built into the design of the CNN. Figure 14 shows what the feature vectors look like corresponding to each layer and each number 0-9. Thus, for example, the top left figure labeled 'LGN' is a 784x10 matrix where each column is the feature output vector for each of the ten numbers as determined by the CNN at that layer. The lengths of the vectors are as follows: LGN – 784, V1 – 2352, V2 – 400, V4 – 120, IT – 200, and PFC – 10.

The first BAM was formed by associating the 10 columns of the LGN matrix with the corresponding 10 columns of the V1 matrix, or in matrix notation $BAM1=(LGN)(V1)^T$. This is equivalent to addition of the 10 associations formed by the outer product of the individual columns. This procedure is repeated over the next four layers to form five BAM matrices. The dimensions of the five matrices are 784 x 2352, 2352 x 400, 400 x 120, 120 x 200, and 200 x 10. Figure 15 shows the five BAM matrices connected in series. Note that our choice of notation for V1-V4 will not use subscripts. This is done to separate Ventral Pathways inspired matrices from the V matrices normally associated with the SVD.

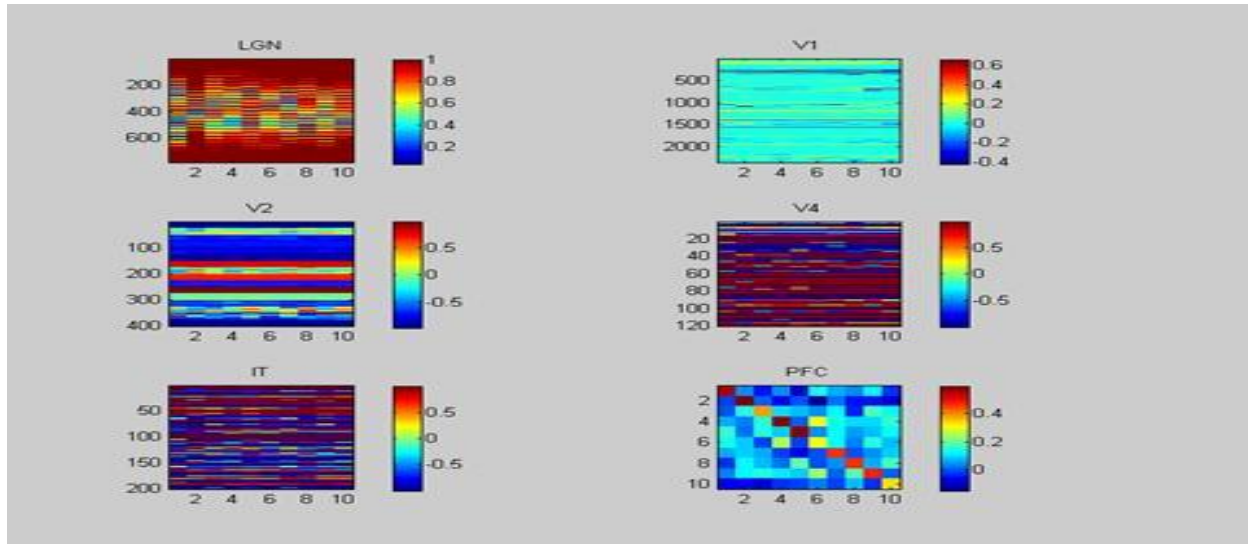


Figure 14: Feature vectors for each of the hierarchical layers for numbers '0' through '9'

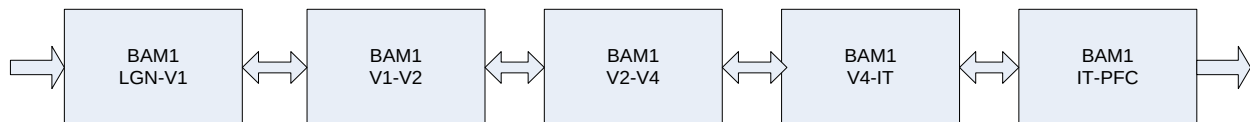


Figure 15: BAM association matrices for each of the hierarchy levels

It is interesting to note that while the LGN is simply related to the input image the other feature vectors are the result of several steps of internal processing; for example convolution, down sampling, weight adjustments, biases, and activation functioning. Although a CNN must execute of these steps on its way to classifying an object, each BAM in a sense jumps over these processes by using the output vectors only. Also, although the output vectors from the CNN and BAM processes are similar they are in fact different enough to where we can claim an increase in accuracy by using both algorithms together to make a classification decision. This will be shown in section 5.

For the results given in this paper we multiplied the five intermediate additive matrices to form one matrix. That is, we set

$$M = (BAM1)(BAM2)(BAM3)(BAM4)(BAM5) \quad [7]$$

By forming M in this way we have relegated BAM to behaving as a feed forward process. Although in Section 3 full bidirectionality was used in the relatively simple single layer application to obtain the results shown in Figures 4-6, we decided to utilize the forward only mode of the BAM process and report on the existence of the success of the

BAM algorithm in the context of multiple layers. We hope that showing these results will motivate future research into combining BAM matrices in both feedback and feed forward modes and integrating portions of the CNN.

We performed two tests. The first test used the full M matrix while the second test used a construction similar to the M18 construction; we will call these matrices M and M18 to be consistent. Figure 16 shows on the left the five original BAM matrices which were multiplied together to form a full M matrix. On the right we show the five PICANS derived BAM matrices that went into forming M18. The scales are different due to the scaling of the singular values.

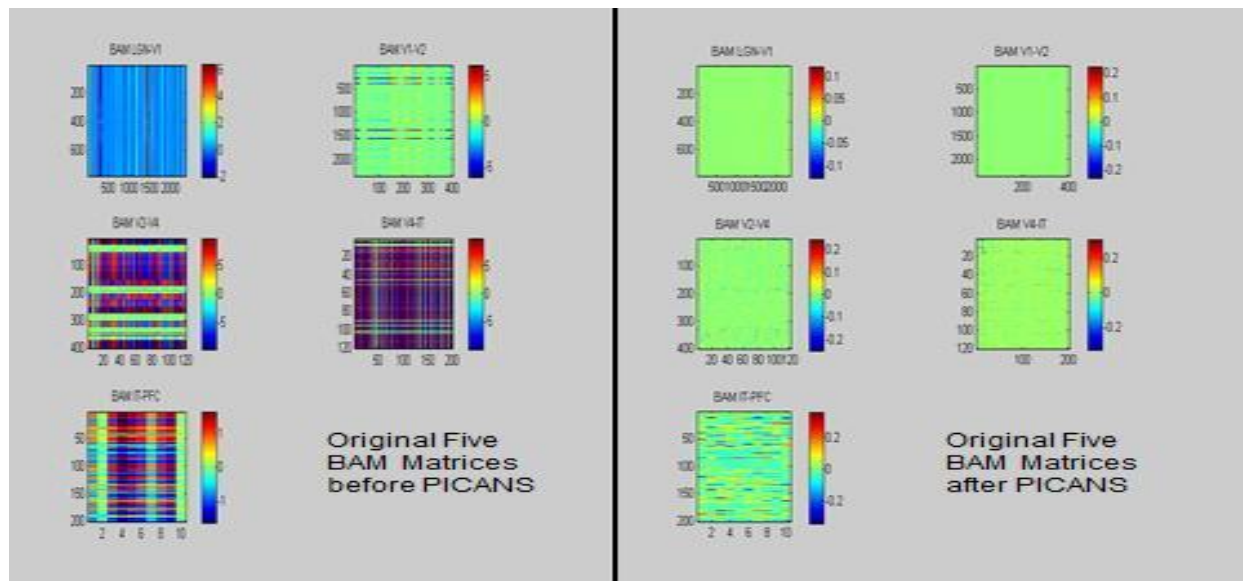


Figure 16: Comparison of the hierarchical BAM matrices before and after applying the PICANS algorithm

For the test data we chose from the MNIST data base the first 100 sets each of the categories 0-9 for a total of 1000 tests. That is we took 100 samples for the number '0', then we took 100 samples of the number '1', and so on up to '9'. Each of the 1000 samples were introduced to the BAM LGN-V1 matrix and then followed the cascade of BAM matrices before ending at the PFC decision.

In Figure 17 we chose to display the results in two forms for the purpose of making general observations. On the left are four images that show results for all 10 categories simultaneously while on the right the four plots show the single output decision for each category. (Note that we have plotted the results for 0-9 as categories 1-10.)

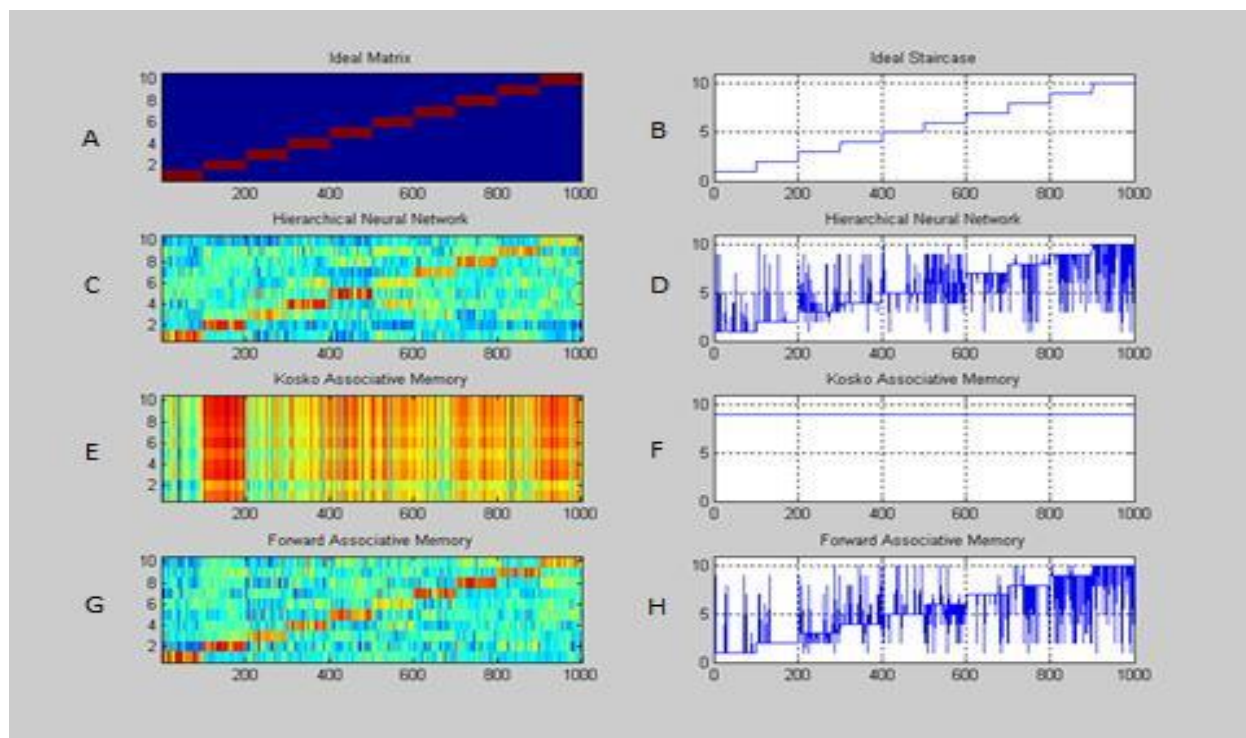


Figure 17: Results of applying CNN, BAM and CNN/BAM hybrid to MNIST dataset

The first row (17 A-B) describes how the ideal case would appear in matrix and plot form. Here we would expect a block diagonal output indicating each decision was made correctly and furthermore only that decision was expected because there are no off-diagonal indications. Hence we see in the plot that 100 zeros were found when fed in 100 '0' images and at no other time was a '0' found.

The second row (17 C-D) shows the results from using the CNN algorithm. The block diagonal is not perfect, and actually running at about 79% accuracy. We have included this as a baseline indicator to compare to the BAM results. No parameters were adjusted to improve performance.

The third row (17 E-F) shows the results in using the Kosko formulation for an M matrix. (17 E) shows structures but nothing indicating a block diagonal. The plot on the right shows that no matter which number was used as the input the digit '8' (category 9) scored the highest. This is similar to the results we saw in Figure 4 where both the digits 5 and 7 were attracted to the digit 3.

The fourth row (17 G-H) shows the results from using the M18 matrix resulting from the PICANS adjustment to M. (17G) shows a block diagonal pattern like that seen for the CNN. This test also ran at about 83% in accuracy. While the improvement over the CNN results is slightly better we are very satisfied that the PICANS approach is comparable at all. More significant is that the decisions found contributing to each accuracy score were not identical and thus motivated us to form a hybrid scoring which we discuss in the next section.

We have shown that by invoking our PICANS approach, which manages the outer products of the neural outputs from each layer which form the basis for associated memory, the results indicate the success of the algorithm.

Section 5: Summary and Conclusions

Our results show that by managing the principal components, modeled in the way of ventral pathways of neurons, the outer product associations stand a better chance of converging to stable and expected end states. We utilized the PCA approach by taking the SVD, incorporating equalization through the ICA, and then extending the association space out into the nullspace. Here we simply added several nullspace components, equally weighted, to the principal components of the association matrix. But, as the nullspace becomes better understood there should be a more refined process to counter the unintended and spontaneous inversions of the associations as well as to improve the stable state convergence properties.

We have demonstrated the success of PICANS BAM process in several ways. First we showed how to use the PICANS process in a single layer BAM association, applied to hand written numeral character recognition from the MNIST database. Second, a hierarchical set of associations was built using a hybrid of CNN and BAM. The resulting hierarchical system was tested with the handwritten character database, MNIST. It was also compared to the performance of a BAM and a CNN used independently.

ACKNOWLEDGEMENTS

We wish to acknowledge the support of Dr. Jay Myung, program manager for the AFOSR Joint Program in Cognition, Decision, Human System Interaction, Robust Decision Making and Computational Intelligence, and Mr. Mike Geertsen, program manager for the DARPA Visual Media Reasoning program.

REFERENCES

- [1] M. Riesenhuber and T. Poggio, (1991b). Hierarchical Models of Object Recognition in Cortex. *Nature Neuroscience*, 2:1019-1025.
- [2] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition" *Proceedings of the IEEE*, v. 86, pp. 2278-2324, 1998.
- [3] Cohen, M.A. and Grossberg, S. (1983), Absolute stability of global pattern formation and parallel memory storage by competitive neural networks, *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13, 815-826.
- [4] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci.*, vol. 79, pp. 2554–2558, 1982.
- [5] Kohonen T., "Correlation matrix memories", *IEEE Trans. Comput*, vol. C-21, pp. 353-359, 1972.
- [6] Bart Kosko, " Bidirectional Associative Memories", *IEEE Transactions On Systems, Man, and Cybernetics*, vol. 18, no.1, January/February 1988
- [7] S. Chartier and M. Boukadoum, "A bidirectional heteroassociative memory for binary and grey-level patterns," *IEEE Trans. Neural Netw.*, vol. 17, no. 2, pp. 385–396, Mar. 2006.
- [8] Y. LeCun, Courant Institute, Corinna Cortes, Google Labs, New York, "The MNIST database of handwritten digits," <http://yann.lecun.com/exdb/mnist>