Final Report for ECS-9618597

High Performance Associative Memory: Practical Generalizations of the Hamming Net

by

Mohamad H. Hassoun
Department of Electrical and Computer Engineering
Wayne State University
Detroit, MI 48202

and

Paul Watta
Department of Electrical and Computer Engineering
University of Michigan-Dearborn
Dearborn, MI 48128

Project Summary

The goal of this project was to investigate new approaches for designing associative neural memories to overcome some of the well known problems of existing models, such as limited capacity, low or unquantifiable error correction capability, large number of spurious memories, and impractical hardware implementation. In terms of hardware implementation, difficulties arise either from requiring complicated hardware or else requiring an excessive number of interconnection weights (for example, fully interconnected architectures).

To address these problem, we proposed a class of associative memory architectures, performed a theoretical analysis, and did extensive testing and simulation on a difficult and real-world problem: human face recognition. Also as part of this project, we constructed a database of human face images suitable for associative memory research.

In terms of architecture, we proposed a class of associative memory models which generalize the operation of the Hamming associative memory. Four different models were constructed: Grounded Hamming Memory, Cellular Hamming Memory, Decoupled Hamming Memory, and Two-level decoupled Hamming Memory. The grounded Hamming memory is similar to the Hamming associative memory, but allows for a ground state which attracts all states with very low signal-to-noise ratio. The cellular Hamming memory utilizes local Hamming distance measures rather than a global Hamming distance measure, leading to a cellular network architecture which is more amenable to VLSI hardware implementation and fine-grained parallel implementations. The decoupled Hamming memory also uses local Hamming distance computations, but requires less hardware than the cellular Hamming memory because the local windows do not overlap. Finally, the two-level decoupled Hamming memory combines the decoupled Hamming distance computations with a higher-level decision making stage.

For the two-level decoupled Hamming network, and in the case of random and binary memory patterns, we were able to derive a theoretical analysis to characterize its memory capacity and error correction capability as a function of system dimension, window size, and input noise level. The analysis also characterizes the expected number of votes that each image in the memory set will receive.

We did extensive simulation and testing of the proposed models. Simulation results were shown to be in close agreement with the theoretical results. More importantly, the capacity of the two-level decoupled Hamming memory was shown to be substantially larger than other single layer neural net memories, such as the Hopfield network.

In addition to measuring capacity, we applied the two-level memory to an important and difficult real-world problem: human face recognition, and performed extensive tests on the capacity of the two-level memory in the case of storing these highly correlated patterns (images). As part of this project, we formulated a database of face images which were obtained in a controlled setting so that we can measure the classification performance of the system. We collected a database of 200 individuals.

In face recognition applications, the ability of the system to reject unknown individuals is just as important as the ability to correctly identify known individuals. To provide a rejection mechanism, we used a pair of thresholds to determine whether to classify the image or reject it. The results of the face recognition experiments showed that the two-level decoupled Hamming memory gave good correct classification performance on a database consisting of 100 people (400 images total in the memory set). For most of our face recognition experiments, the two-level memory gave correct classification of 100% with a rejection of about 5%. For the false positive experiments (the ability of the system to reject unknown individuals), the system achieves a false positive rate of 0.4%.

Table of Contents

1. Introduction

The goal of this project was to investigate new approaches for designing associative neural memories to overcome some of the well known problems of existing models, such as

- Limited capacity
- Low or unquantifiable error correction capability
- Large number of spurious memories
- Impractical hardware implementation

In terms of hardware implementation, difficulties arise either from requiring complicated hardware or else requiring an excessive number of interconnection weights (for example, fully interconnected architectures).

To address these problem, we proposed a class of associative memory architectures, performed a theoretical analysis, and did extensive testing and simulation on a difficult and real-world problem: human face recognition. Also as part of this project, we constructed a database of human face images suitable for associative memory research. Our contributions in each of these areas are briefly summarized below.

Architecture

To overcome the problem of limited capacity, we proposed a class of associative memory models which generalize the operation of the Hamming associative memory. Five different models were constructed:

- Grounded Hamming Memory
- Cellular Hamming Memory
- Decoupled Hamming Memory
- Two-level decoupled Hamming Memory

The grounded Hamming memory is similar to the Hamming associative memory, but allows for a ground state which attracts all states with very low signal-to-noise ratio. The cellular Hamming memory utilizes local Hamming distance measures rather than a global Hamming

distance measure, leading to a cellular network architecture which is more amenable to VLSI hardware implementation and fine-grained parallel implementations. The decoupled Hamming memory also uses local Hamming distance computations, but requires less hardware than the cellular Hamming memory because the local windows do not overlap. Finally, the two-level decoupled Hamming memory combines the decoupled Hamming distance computations with a higher-level decision making stage.

Although we initially formulated each of these models for binary memory patterns, we were able to extend their operation to the case of real-valued patterns (such as grayscale images).

Theoretical Analysis

For the two-level decoupled Hamming network, and in the case of random and binary memory patterns, we were able to derive a theoretical analysis to characterize its memory capacity and error correction capability as a function of system dimension, window size, and input noise level. The analysis also characterizes the expected number of votes that each image in the memory set will receive. An interesting result emerged: the system will perform extremely well in the case of very large window sizes and very small windows, and performance will decrease for intermediate-sized windows. The exact nature of the relation between system performance and window size is captured by the theoretical analysis.

Testing and Simulation

We did extensive simulation and testing of the proposed models. Simulation results were shown to be in close agreement with the theoretical results. More importantly, the capacity of the two-level decoupled Hamming memory was shown to be substantially larger than other single layer neural net memories, such as the Hopfield network.

In addition to measuring capacity, we applied the two-level memory to an important and difficult real-world problem: human face recognition, and performed extensive tests on the capacity of the two-level memory in the case of storing these highly correlated patterns (images).

In face recognition applications, the ability of the system to reject unknown individuals is just as important as the ability to correctly identify known individuals. To provide a rejection mechanism, we used a pair of thresholds to determine whether to classify the image or reject it. The thresholds were based on the statistics of the images in the database. The results of the face recognition experiments showed that the two-level decoupled Hamming memory gave good correct classification performance on a database consisting of 100 people (400 images total in the memory set). For most of our face recognition experiments, the two-level memory gave correct classification of 100% with a rejection of about 5%. For the false positive experiments (the ability of the system to reject unknown individuals), the system achieves a false positive rate of 0.4%.

Database Construction

As part of this project, we formulated a database of face images which were obtained in a controlled setting so that we can measure the classification performance of the system (as opposed to studying the effects of the pre-processing stages, such as face segmentation, size normalization, etc.). We designed a simple laboratory apparatus which allowed us to snap images in a constrained way, and collected a database of 200 individuals. For each individual, we stored 4 images in the memory set (or database) and 2 images in the test set. The 4 memory set images showed different facial expressions of the subject: blank, smile, angry, surprised. The test set contained a blank expression and an arbitrary expression, where the subject was told to try to fool the system by giving an unusual expression.

Overview

All of the proposed associative memory models have been individually described and analyzed in the literature [1] - [11]. In the remainder of this report, we will give more details concerning the scientific advances made in this project. In Section 2, we will review the proposed associative memory architectures. Section 3 will give a brief summary of the theoretical analysis

that was undertaken. Sections 4 and 5 will give a summary of the experimental results that we obtained, as well as outline how the face database was constructed.

2. Summary of ANM Architecture Designs

In the following, we consider the binary autoassociative memory problem. In this case, the given fundamental memory set is of the form $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^m\}$, where each pattern $\mathbf{x}^i$ is an N-bit binary vector, i.e., $\mathbf{x}^i \in \{0, 1\}^N$, $i = 1, 2, \ldots, m$. The task is to design a system which associates every fundamental pattern with itself. That is, when presented with $\mathbf{x}^i$ as input, the system should produce $\mathbf{x}^i$ at the output. In addition, when presented with a *noisy* version of $\mathbf{x}^i$ at the input, the system should also produce $\mathbf{x}^i$ at the output.

Let the Hamming distance between two binary vectors $\mathbf{x}$ and $\mathbf{y}$ (of the same dimension) be denoted as $d(\mathbf{x}, \mathbf{y})$.

The Hamming associative memory [12-15] is a static model, and operates as follows: For any memory key $\mathbf{x} \in \{0, 1\}^N$, the retrieved pattern is obtained by computing the Hamming distances $d_k = d(\mathbf{x}, \mathbf{x}^k)$, selecting the minimum such distance d_{k^*}, and outputting the fundamental memory $\mathbf{x}^{k^*}$ (closest match). The most attractive feature of this model is its exponential capacity [8] and large error correction capability. There are two serious disadvantages of this model. First, there is no provision for a ground state. Second, the hardware implementation is cumbersome because the computation of the required Hamming distances is a global and sequential operation, and hence not immediately suitable to parallel and distributed processing systems. The following sections detail generalizations of the Hamming memory which overcome these limitations.

2.1 The Grounded Hamming Memory

The operation of the grounded Hamming associative memory model [2, 11] is similar to the Hamming associative memory, but provides for a ground state (the zero state) to attract all outlier or "garbage" inputs. That is, the grounded Hamming associative memory outputs the closest

fundamental memory when there is a sufficiently close match between the memory key and one of the fundamental patterns; otherwise, when no such match occurs, the grounded Hamming memory converges to the ground state. The Hamming memory, on the other hand, has no such ground state, and will produce a fundamental memory at the output no matter how noise-corrupted the memory key. Clearly, there are applications which require such a "no decision" state in the presence of excessive noise. Figure 1 shows conceptually the difference between the Hamming associative memory and the grounded Hamming memory in terms of state space basins of attraction.

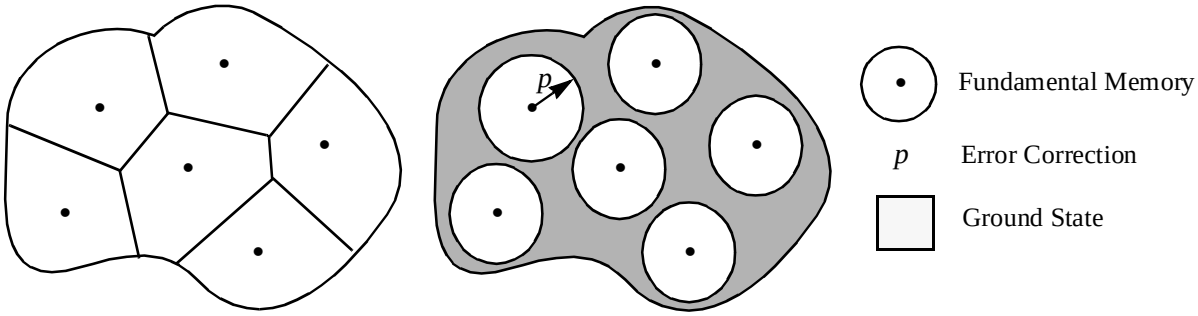


Figure 1. Basins of attraction of the (a) the Hamming memory and (b) the grounded Hamming memory.

The first step in the design of the grounded Hamming memory is to choose some desired level of *error correction*. The easiest way to do this is to choose a value for *p*, the number of correctable bit errors. If the number of correctable bit errors is to be uniform for all memories in the fundamental memory set, then *p* must be chosen such that the following condition is satisfied

$$1 \le p < \frac{1}{2}\min \{d(\mathbf{x}^i, \mathbf{x}^j)\} \qquad (1)$$

where $i \ne j \in \{1, \ldots, m\}$. Once *p* is chosen, the grounded Hamming memory operates as follows. Given an input memory key $\mathbf{x}$, we compute $d(\mathbf{x}, \mathbf{x}^i)$ for each $i = 1, 2, \ldots, m$. If there is an index i^* which satisfies $d(\mathbf{x}, \mathbf{x}^{i^*}) \le p$, then the output of the memory is $\mathbf{x}^{i^*}$; otherwise, the output of the memory is the ground state $\mathbf{0}$. Note that by the condition established in (1), if i^* exits, then

it is unique.

In Watta, Wang, and Hassoun [4], it was shown that an explicit Boolean expression for the grounded Hamming memory operation can be derived. In this case, the system may be implemented with digital hardware, as shown schematically in Figure 2. Here, each $\lambda_{\mathbf{x}^i}(\mathbf{x})$ block is a collection of AND and OR gates; see [4] for details. Unfortunately, this design is not practical for high dimensional systems because it requires an excessive number of gates.

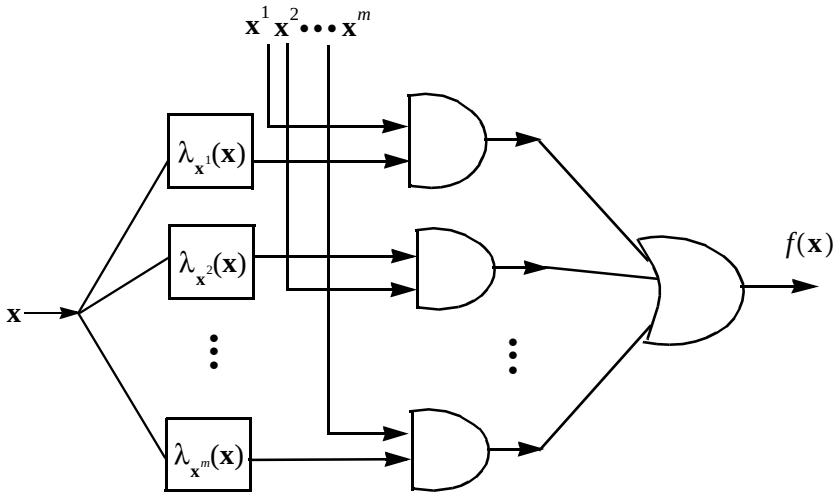


Figure 2. Circuit diagram of the grounded Hamming memory. Each $\lambda_{\mathbf{x}^i}(\mathbf{x})$ is a block of AND and OR gates.

A more efficient design of the grounded Hamming memory may be obtained by using linear threshold gates (LTGs). The neural architecture for this system is shown in Figure 3. Here, each of the *m* LTGs is "tuned" to respond to a single fundamental memory. In this case, we require at most $m(N+1)$ LTG's, which scales linearly in *N*. Note the incredible savings in hardware for this LTG-realization of the grounded Hamming memory as opposed to the digital logic realization, which required an exponential number of gates. In this sense, and as mentioned in [16] in a different context, the LTG is *exponentially more powerful* than digital logic gates.

Another attractive feature of the architecture shown in Figure 3 is the ease with which the size of the basin of attraction for all fundamental memories can be changed. Here, to change *p*, we

need only adjust the threshold of each neuron in the hidden layer. In fact, in this case, we don't require a uniform value of p for all memories: we can define a nonuniform error correction for each fundamental memory, p_j, $j = 1, 2, \ldots, m$, where

$$1 \le p_j < \frac{1}{2}\min \{ d(\mathbf{x}^j, \mathbf{x}^k) : k = 1, \ldots, m, k \ne j \} \qquad (1)$$

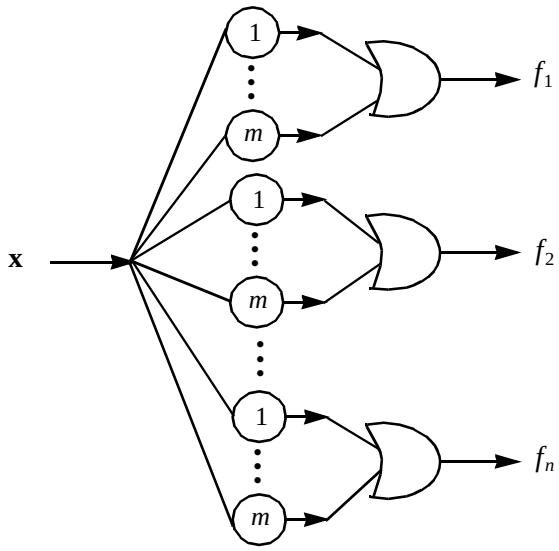


Figure 3. LTG-realization of the grounded Hamming memory.

The state space structure for the grounded Hamming memory with nonuniform error correction is shown schematically in Figure 4. Note that the grounded Hamming memory with uniform error correction uses the most conservative value of p; i.e., $p = \min\{p_j : k = 1, \ldots, m\}$. The nonuniform grounded Hamming memory, though, is able to handle (for some images) much more noise.

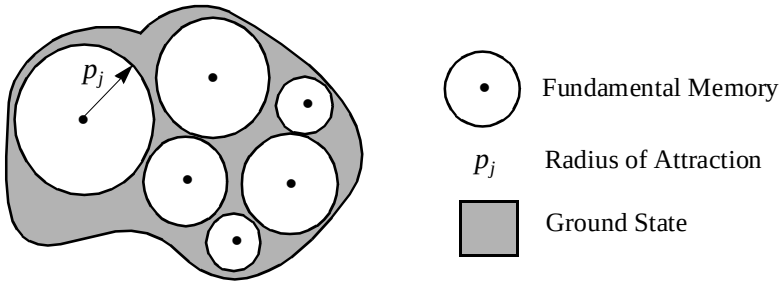


Figure 4. The grounded Hamming memory with nonuniform error correction.

2.2 Cellular Hamming Memory

For the grounded Hamming net formulated above, each output bit is a function of the entire input vector; i.e. $y_i = y_i(x_1, x_2, \ldots, x_N)$ for each $i = 1, 2, \ldots, N$. Substantial savings in hardware may be achieved by restricting the dependence of each output to a small fraction of all possible inputs, giving rise to the notion of a *local Hamming distance* measures. Since our application will be for image processing, we will suppose that the memory input pattern is 2-dimensional. In this case, the cellular Hamming model is actually a two-dimensional *nonuniform* cellular automaton in which neighboring pixels interact locally, as shown in Figure 5(a). Here, the pixel interconnectivity structure is shown as a 2×2 Von Neumann-type neighborhood, but larger neighborhoods, such as the 3×3 Moore neighborhood, or extended Moore neighborhoods may be formed, as well.

Previous studies of 2-dimensional cellular automata [17] concentrated on uniform systems in which each automaton (pixel element) employs the same transition or updating function. Here, to obtain the operation of associative recall, each automaton may employ a different transition function, formulated as follows. First, each pixel senses the state of the pixels in its neighborhood. For example, suppose the pixel marked with a circle in Figure 5(b) is chosen for updating and suppose we employ a 3×3 neighborhood (shown as the shaded region surrounding the updating pixel). The updating pixel updates itself as follows:

(1) Compute the Hamming distance between its neighborhood configuration and the corresponding pixels in all the memory patterns. For example, in Figure 5(b), the distances $\{d_1, d_2, \ldots, d_m\}$ are computed.

(2) Choose the smallest such distance d_{k^*}.

(3) Assume the value of the center pixel in that closest pixel pattern.

A simple heuristic may be employed in case of a tie in the minimum Hamming distance computation. For example, if there is a tie between two images and if the two fundamental memories agree at pixel (i, j), then we simply assign the pixel to the common value. If, however,

they disagree at (i, j), then heuristically, we can keep the input image pixel at its present value. Similar rules can be formulated in the case of a tie among several memory patterns.

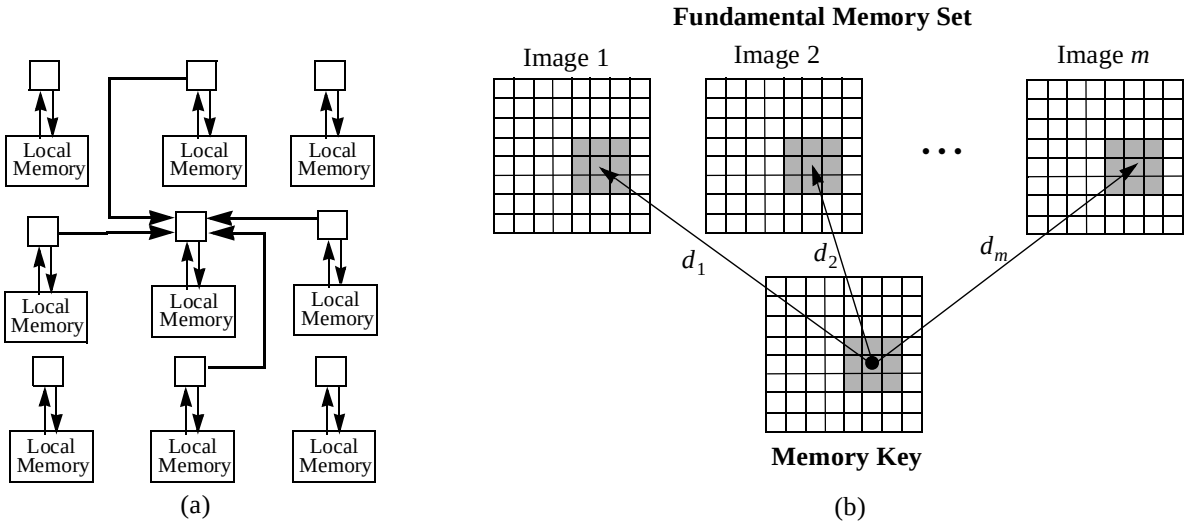


Figure 5. (a) Structure of local interactions for the local Hamming net, assuming a 2×2 neighborhood. (b) Updating of the memory key by local Hamming distance computations (here, a 3×3 neighborhood is used).

There are several ways in which the dynamics can be computed. For *parallel update*, all the nodes in the network update their state at each time instant. For *sequential update*, only one node is updated at each time instant following some fixed ordering of the nodes. *Block sequential update* is a mixture of sequential and parallel update in which a partition is formed on the set of nodes, and the updating of the partitions follows sequentially, while the updating of the nodes within each partition block is performed in parallel. For *random update*, only one randomly chosen node is updated at each time instant.

It is important to note that the full Hamming net can be viewed as a special case of this cellular memory, corresponding to the case where each pixel has a neighborhood consisting of the entire image. We expect the quality of memory retrievals will be best for larger neighborhoods, approaching the (optimal) performance of the Hamming net in the limit of maximum neighborhoods.

2.3 Decoupled Hamming Associative Memory

The local Hamming memory uses overlapping windows and hence requires a lot of hardware. It is possible, though, to use non-overlapping windows, giving rise to the *decoupled Hamming associative memory.*

The decoupled Hamming associative memory localizes the Hamming distance computation by partitioning the input vector into non-overlapping modules or windows, and performing the Hamming memory operation on each module independently. To be precise, suppose we partition the N input variables $X = \{x_1, x_2, \ldots, x_N\}$ of our memory into w modules: $\{X_1, X_2, \ldots, X_w\}$ such that $X_i \subset X$, $\bigcup X_i = X$, and $X_i \cap X_j = \varnothing$, $i \neq j$. To simplify notation, assume that each module has the same number of variables, denoted n. In this case, we have $|X_i| = n$, $i = 1, 2, \ldots, w$, where $w = N/n$ is the total number of windows or modules. Figure 6 shows the structural difference between (a) the full Hamming memory and (b) the decoupled Hamming memory.

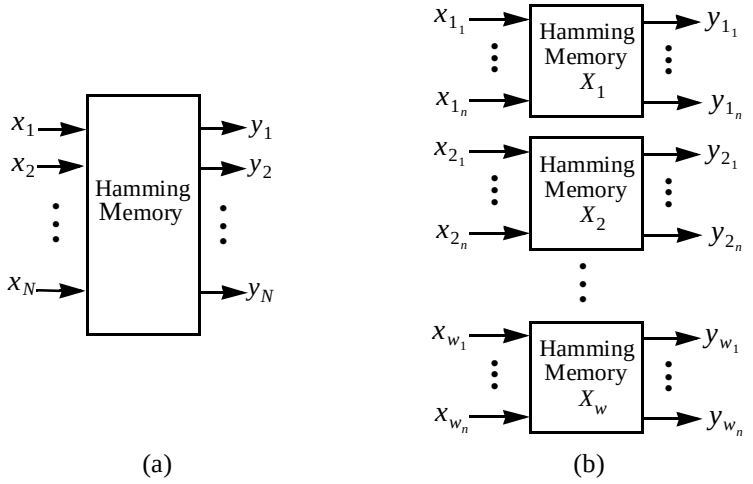


Figure 6. Structure of (a) the full Hamming and (b) the decoupled Hamming memory.

Each module is a local Hamming memory and has its own local memory set, which is obtained by partitioning each fundamental memory $\mathbf{x}^k$ into w memory subvectors:

$\mathbf{x}^k = [\mathbf{x}^k_{(1)}, \ldots \mathbf{x}^k_{(w)}]$, where the *i*th subvector $\mathbf{x}^k_{(i)} \in \{0, 1\}^n$ contains the components of $\mathbf{x}^k$ specified by the variables in the *i*th module X_i. In this case, we can associate with each module its own local memory set of the form $\mathbf{x}_{(i)} = \{\mathbf{x}^1_{(i)}, \ldots, \mathbf{x}^m_{(i)}\}$.

The decoupled Hamming memory operates as follows: The memory key $\mathbf{x}$ is partitioned in the same fashion as the fundamental memories: $\mathbf{x} = [\mathbf{x}_{(1)}, \ldots \mathbf{x}_{(w)}]$, and the *w* module Hamming memories independently (and in parallel) operate on each of the subvectors of **x**, computing the Hamming distances $d(\mathbf{x}_{(i)}, \mathbf{x}^k_{(i)})$, $k = 1, 2, \ldots, m$, and outputting the closest matching pattern.

In the case of 2-dimensional patterns, there are many different topologies possible for the layout of the local Hamming memories. For example, the local Hamming memories may be arranged by row, by column, or in a checkerboard arrangement, as shown in Figure 7(a). Here, the 64×64 binary image is covered with non-overlapping 16×16 windows in a checkerboard-type layout. Each local Hamming memory then computes 256-bit Hamming distances as opposed to 4096-bit Hamming distances for the entire image.

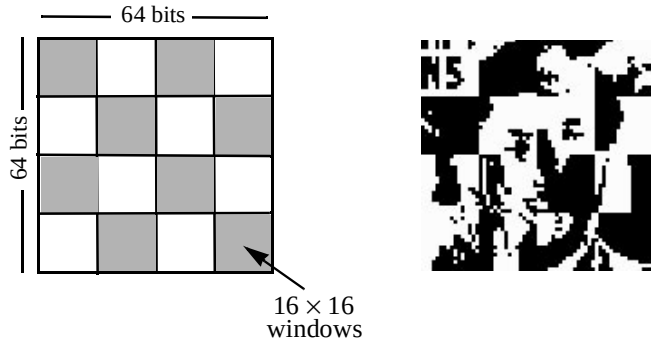


Figure 7. (a) Structure of local memories arranged as an array of non-overlapping 16×16 windows. (b) A spurious memory.

One clear advantage of the decoupled Hamming memory over the full Hamming memory is retrieval speed. Since all modules can perform their computations in parallel, a *w*-fold speedup in retrieval time can be achieved by dedicating a processor to each module. A disadvantage of this stringent parallelism, though, is that the decoupled Hamming memory may retrieve a pattern

which was not part of the memory set; i.e., spurious memories are possible. For image processing applications, for example, the decoupled memory may converge to the correct fundamental image, but contain scattered "chunks" of other images, as shown in Figure 7(b). The full Hamming network, on the other hand, never retrieves spurious memories.

2.4 The Two-Level Decoupled Hamming Associative Memory

To overcome the spurious memory problem of the previous section, a two-level structure can be used which consists of a decoupled Hamming memory along with a higher-level decision network. The architecture of this memory (in the case of 2-dimensional memory patterns) is shown in Figure 8(a). Here, each local Hamming memory or module X_i computes the closest matching pattern and sends the index I_i of the best match pattern to the decision network. The decision network examines the indices $I_1, I_2, \ldots, I_w$ of all the modules and computes a single best match index I^*. Each memory module then outputs its portion of the fundamental memory $\mathbf{x}^{I^*}$; that is, each module outputs $\mathbf{x}^{I^*}_{(i)}$, $i = 1, 2, \ldots, w$. Since the decision network forces all modules to output the same fundamental memory, the spurious memory problem of the previous section is eliminated.

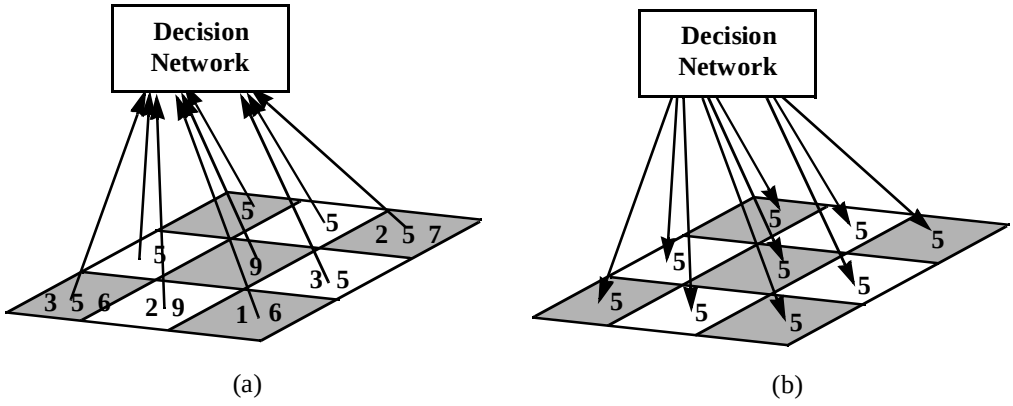


Figure 8. Structure of the two-level decoupled Hamming network. Numbers in (a) represent the index of the closest matching pattern(s), and (b) shows the result after the voting.

For example, in Figure 8(a), the window in the upper left hand corner of the image best matches image 5 in the memory set, while the window in the lower right hand corner best matches images 1 and 6 (there is a tie in the Hamming distance). The decision network examines all the votes from the local windows, determines that 5 is the most prevalent, and forces all windows to output its portion of image 5, as shown in (b).

There are many ways to design the decision network. In the simplest case, a majority rule is used, in which I^* is chosen to be the most frequent index among $I_1, I_2, \ldots, I_w$. Utilizing the emerging theory of *classifier combination* [18] and *sensor fusion* [19], more sophisticated decision rules can be formulated. In this case, it may be desirable for each module to send an ordered list of, say, the best 3 indices $I_{i_1}, I_{i_2}, I_{i_3}$ to the decision network. For very noisy patterns, the second and third choices of each module may contain useful information which can be exploited with an appropriate combination scheme.

As with the single layer decoupled Hamming network, it is easy to see that the 2-level decoupled Hamming network reduces to the full Hamming network in the case of a single module: $w = 1$. But unlike the single layer model, the 2-level decoupled Hamming memory also reduces to the full Hamming network in the other extreme case: $w = N$. So the 2-level decoupled Hamming network achieves the optimal performance of the Hamming memory for both the maximum and minimum number of modules.

For intermediate window sizes, the capacity of the two-level decoupled Hamming memory is not as large as the full Hamming memory. But even so, the two-level decoupled Hamming memory with intermediate window size has a much higher capacity and much more error correction than most of the standard neural-based associative memories, such as the correlation-recorded Hopfield network [13, 14, 20], and other recording algorithms for the same single-layer Hopfield-type neural structure.

Besides its performance advantages over standard neural net models, the two-level decoupled Hamming net is ideal for parallel hardware implementation. Since the first level is modularized,

the computation can be done in parallel. Indeed, special purpose hardware consisting of a dense array of digital signal processors already exists which can perform the required computations efficiently (see, for example, [21]).

3. Summary of Theoretical Results

In this section, we give a theoretical analysis of the capacity of the two-level decoupled Hamming network. The complete derivation of these results can be found in [1].

3.1 Expected Number of Votes

We assume that the memory set consists of uniformly random vectors. That is, each bit of a fundamental memory has a 50 percent chance of being 1 and 50 percent chance of being 0. In addition, it is assumed that the memory key is a corrupted version of one of the fundamental memories. In particular, the memory key is obtained by adding an amount ρ of uniform random noise to one of the fundamental memories—called the *target memory*; i.e., with probability ρ, each bit of the target image is flipped from its original value. Each of the remaining $m-1$ fundamental memories will be called a *non-target* memory, or *other* memory (short for "other than the target").

The analysis will proceed by first computing the probability that the given local window votes for the target memory and the non-target memories, then the number of votes for the target memory and non-target memories will be computed, and finally, the capacity will be estimated by computing the probability that the target memory gets the highest number of votes.

3.2 Probability of Voting for the Target and a non-Target Memory

In reference to Figure 9, let us fix a window in the memory key (say, the highlighted window), and let us fix the corresponding window in each of the fundamental memories. In addition, of the $m-1$ non-target memories, let us focus our attention on a single one of them, say the ith memory.

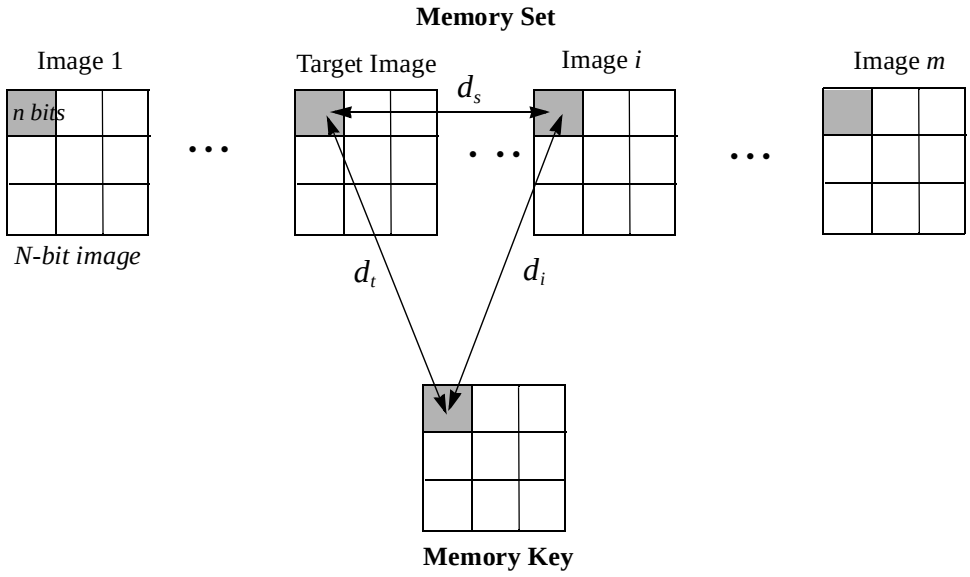


Figure 9. The fundamental memory set and memory key for the 2-level decoupled Hamming memory. The Hamming distances d_t, d_i, and d_s are indicated.

Let d_t denote the Hamming distance between the highlighted local window of the memory key and the corresponding window of the target memory, let d_i denote the Hamming distance between the highlighted local window of the memory key and the corresponding window of the *i*th (non-target) image in the memory set, and let d_s denote the Hamming distance between the highlighted local window of the target image and the corresponding window of *i*th image.

Clearly, since each of the fundamental memories is a uniform random binary vector, then the probability that $d_s = j$ bits (where $0 \le j \le n$) follows a binomial distribution of the form

$$Prob(d_s = j) = \binom{n}{j}\left[\frac{1}{2}\right]^j\left[\frac{1}{2}\right]^{n-j} = \binom{n}{j}\left[\frac{1}{2}\right]^n \quad (1)$$

where $\binom{n}{j}$ is the number of combinations of *n* items chosen *j* at a time, and is given by

$$\binom{n}{j} = \frac{n!}{j!(n-j)!}$$

Also, since the memory key is obtained by uniformly perturbing the target image with an amount of noise ρ, then the probability that $d_t = k$ bits ($0 \le k \le n$) also follows a binomial

distribution of the form

$$Prob(d_t = k) = \binom{n}{k}\rho^k(1-\rho)^{n-k} \tag{2}$$

Another quantity that will be of interest is $Prob(d_i = j \,|\, d_t = k)$. That is, assuming $d_t = k$ bits, we want to compute the probability that $d_i = j$ bits. The target image and the *i*th memory image are created completely independently of each other. If some of the bits of the target image are subsequently flipped (which is how the memory key is created), it is still independent of the *i*th memory image; hence $Prob(d_i = j \,|\, d_t = k)$ has the same binomial distribution as $Prob(d_i = j)$:

$$Prob(d_i = j \,|\, d_t = k) = Prob(d_i = j) = \frac{\binom{n}{j}}{2^n} = \binom{n}{j}\left[\frac{1}{2}\right]^n \tag{3}$$

Note that each of the other non-target memories in the memory set follows the same distribution, since each of the fundamental memories was created independent and uniformly random. Hence, Equation 3 holds for each non-target memory *i* in the memory set.

Now for which memory will the highlighted window vote? Well, for each $k \in \{0, 1, \ldots, n\}$ the target memory gets a vote if $d_t = k$ **and** $d_{i'} \geq k$ for *all* other $m-1$ memories (here, i' ranges over all indices $1, \ldots, m$ excluding the index of the target image). The probability that this occurs is given by

$$P_t = P_t(n, \rho, m) = \sum_{k=0}^{n} Prob(d_t = k)\left[\sum_{j=k}^{n} Prob(d_i = j \,|\, d_t = k)\right]^{m-1}$$

Substituting Equations 2 and 3 into the above Equation, we get

$$P_t(n, \rho, m) = \sum_{k=0}^{n} \binom{n}{k}\rho^k(1-\rho)^{n-k}\left(\sum_{j=k}^{n} \binom{n}{j}\left[\frac{1}{2}\right]^n\right)^{m-1}$$

On the other hand, for each $k \in \{0, 1, \ldots, n\}$ the *i*th (non-target) memory gets a vote if $d_t = k$ **and** $d_i \leq k$ **and** $d_{i'} \geq d_i$ for all other $m-2$ memories in the memory set (here, i' ranges over all

indices $1, \ldots, m$ excluding index i—for the ith memory—and the index of the target memory). Hence, the probability that the ith image gets a vote $P_i = P_i(n, \rho, m)$ is given by

$$P_i = \sum_{k=0}^{n} Prob(d_t = k) \left\{ \sum_{j=0}^{k} Prob(d_i = j \,|\, d_t = k) \left[\sum_{s=j}^{n} Prob(d_i = s \,|\, d_t = k) \right]^{m-2} \right\} \tag{4}$$

Substituting Equations 2 and 3 into the above Equation, we get

$$P_i(n, \rho, m) = \sum_{k=0}^{n} \binom{n}{k} \rho^k (1-\rho)^{n-k} \left[\sum_{j=0}^{k} \binom{n}{j} \left[\frac{1}{2}\right]^n \left(\sum_{s=j}^{n} \binom{n}{s} \left[\frac{1}{2}\right]^n \right)^{m-2} \right]$$

3.3 Number of Votes for the Target and non-Target Images

Thus far, we computed the probability that a single window will vote for the target and/or one of the non-target memories. The decision network of the two-level decoupled Hamming memory counts up the votes for each of the w windows covering the memory key and then chooses the fundamental memory with the most votes. Hence, the question here is: What is the total number of votes received by the target and each of the non-target memories?

Let N_t denote the total number of votes received by the target memory and N_i the number of votes received by the ith non-target memory. Both N_t and N_i are random variables which follow a binomial distribution. The expected value and variance of the number of votes for the target are given by

$$\mu_t = E[N_t] = P_t \frac{N}{n}$$

$$\sigma_t^2 = V[N_t] = P_t(1 - P_t) \frac{N}{n}$$

Similarly, the expected value and variance for the number of votes received by the ith non-target memory is given by

$$\mu_i = E[N_i] = P_i \frac{N}{n}$$

$$\sigma_i^2 = V[N_i] = P_i(1 - P_i) \frac{N}{n}$$

By the central limit theorem, and assuming a large number of windows, the probability distribution of the random variables N_t and N_i approaches a normal distribution. In this case, the (approximate normal) density function f_t for the number of votes for the target image is given by

$$f_t(x) = \frac{1}{\sqrt{2\pi}\sigma_t} e^{\frac{-(x-\mu_t)^2}{2\sigma_t^2}} \tag{5}$$

and the density function f_i for the number of votes for the *i*th non-target memory can be approximated by

$$f_i(x) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{-(x-\mu_i)^2}{2\sigma_i^2}} \tag{6}$$

The normal approximations of Equations (5) and (6) are useful in obtaining numerical values for N_t and N_i in the case of large *N*.

3.4 Estimation of Memory Capacity

In the previous subsection, we determined the expected number of votes for the target image and the expected number of votes for the *i*th non-target image. Of course the two-level decoupled Hamming memory retrieves the correct (target) image when N_t is larger than N_i for *all* of the $m-1$ non-target memories in the fundamental memory set. That is, N_t must be larger than N_1 and N_2 and . . . and N_{m-1}, where we have assumed, without loss of generality, that the target memory is the *m*th memory (last memory) in the fundamental memory set. To compute the probability of correct retrieval, then, we must determine the maximum of the "other" or non-target votes; hence, we define a quantity N_i^{max}

$$N_i^{max} = max\{N_1, N_2, \ldots, N_{m-1}\} \tag{7}$$

It can be shown that the maximum of a collection of continuous random variables is also a random variable; furthermore, the cumulative distribution function (cdf) of the max random variable is given by the product of the individual cdf's of the random variables being maximized

[22].

In this case, we have $m-1$ random variables which follow the continuous (and approximate) probability density functions $f_1, f_2, \ldots, f_{m-1}$ given in Equation 6. Suppose that the corresponding cumulative distribution functions are denoted by $F_1, F_2, \ldots, F_{m-1}$, respectively. Then, since each of these $m-1$ distributions are identical, we have

$$F_{max}(x) = F_1(x)F_2(x)\ldots F_{m-1}(x) = [F_i(x)]^{m-1} \tag{8}$$

The probability density function of N_i^{max} can be obtained by differentiating Equation 8

$$f_{max}(x) = \frac{d}{dx}F_{max}(x) = (m-1)[F_i(x)]^{m-2}f_i(x) \tag{9}$$

Using the density function in Equation 9, the expected value of N_i^{max} can be computed as follows:

$$\overline{N}_i^{max} = E[N_i^{max}] = \int x f_{max}(x)dx \tag{10}$$

Finally, the probability of correct retrieval is the probability that $N_t > \overline{N}_i^{max}$, i.e., $P_{cor} = Prob(N_t > \overline{N}_i^{max})$ which is computed below

$$P_{cor}(n, N, \rho, m) = \int_{\overline{N}_i^{max}}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_t} e^{\frac{-(x-\mu_t)^2}{2\sigma_t^2}} dx \tag{11}$$

Using the standard normal distribution, Equation 11 can be recast as

$$P_{cor}(n, N, \rho, m) = \int_{\frac{\overline{N}_i^{max}-\mu_t}{\sigma_t}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2} dz \tag{12}$$

And using the standard error function

$$erf(x) = \int_0^x \frac{2}{\sqrt{\pi}} e^{-z^2} dz$$

Equation (12) can be written as

$$P_{cor}(n, N, \rho, m) = \frac{1}{2}\left[1 + erf\left(\frac{\mu_t - \bar{N}_i^{max}}{\sqrt{2}\sigma_t}\right)\right] \quad (13)$$

Equation 13 gives the probability that the target image will be retrieved as a function of system dimension *N*, local window size *n*, noise level ρ, and total number of stored patterns *m*. Numerical estimates of the capacity m^* of the memory, then, can be determined by fixing values for *n*, *N*, and ρ, and computing Equation 13 for increasing values of *m*.

4. Summary of Experimental Results: Random Binary Images

In this section we will summarize the performance of the two-level decoupled Hamming associative memory on binary images.

4.1 Expected Number of Votes

In order to compare with the results of our theoretical analysis, it is assumed that all memory patterns are 2-dimensional images and are generated randomly from a uniform distribution. As with the above theoretical analysis, it is assumed that one of the memory images (the target image) is selected and corrupted with an amount ρ of uniform random noise ($0 \le \rho \le 0.5$). This corrupted image is used as the memory key, and it is desired that the system produce the target image at the output; i.e. retrieve the target image.

Figure 10(a) shows a comparison of the simulation and theoretical results for the expected number of votes for the target N_t, an arbitrary other image (other than target) N_i, and the maximum among these other images N_i^{max}. For these simulations, $m = 10,000$ images, the noise level is set at $\rho = 0.4$, and the local window size is $n = 2 \times 2$. The top Figure shows the simulation results, and the bottom plot shows the corresponding theoretical distributions for these quantities, as given in Equations 5, 6, and 9. In (a), the image size is set at $N = 64 \times 64$, while in (b), the image size is $N = 128 \times 128$.

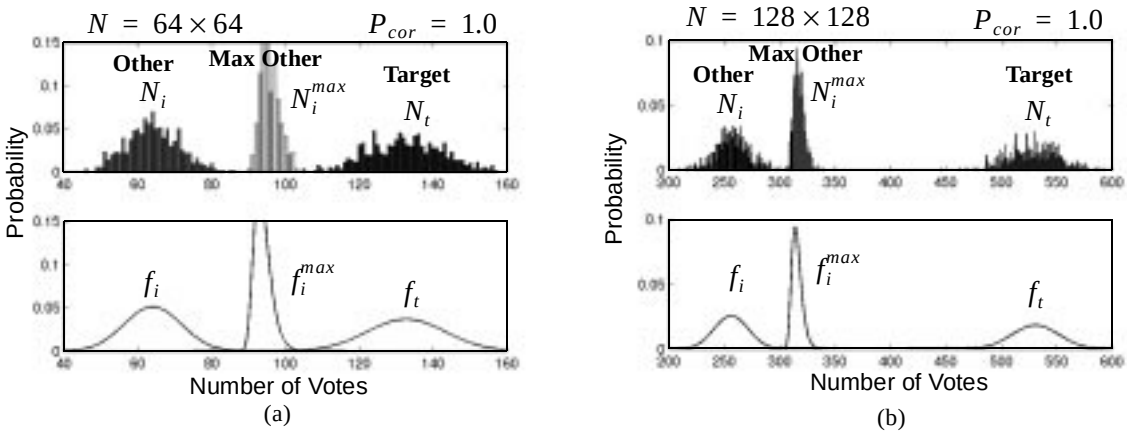


Figure 10. Comparison between simulation and theory for the case of a 2×2 window size, input noise $\rho = 0.4$, memory size of $M = 10000$, and an image size of (a) 64×64 and (b) 128×128.

The simulation results were obtained as follows. The corrupted target memory (with 40% random noise) was input to the two-level Hamming memory, and the number of votes received by the target, a randomly chosen other (non-target) image, and the maximum among the non-target images were recorded. This process was repeated 500 times. In each case, if the system retrieved the target image, the retrieval was considered a success; otherwise it was counted as a failure. The probability of correct retrieval was simply the number of successes out of 500 trials [8].

Clearly, the theoretical results provide a good model for the underlying distributions. Note that in both cases, there is sufficient separation in the distributions of N_t and N_i to successfully perform the classification, and hence the probability of correct retrieval is unity: $P_{cor} = 1$.

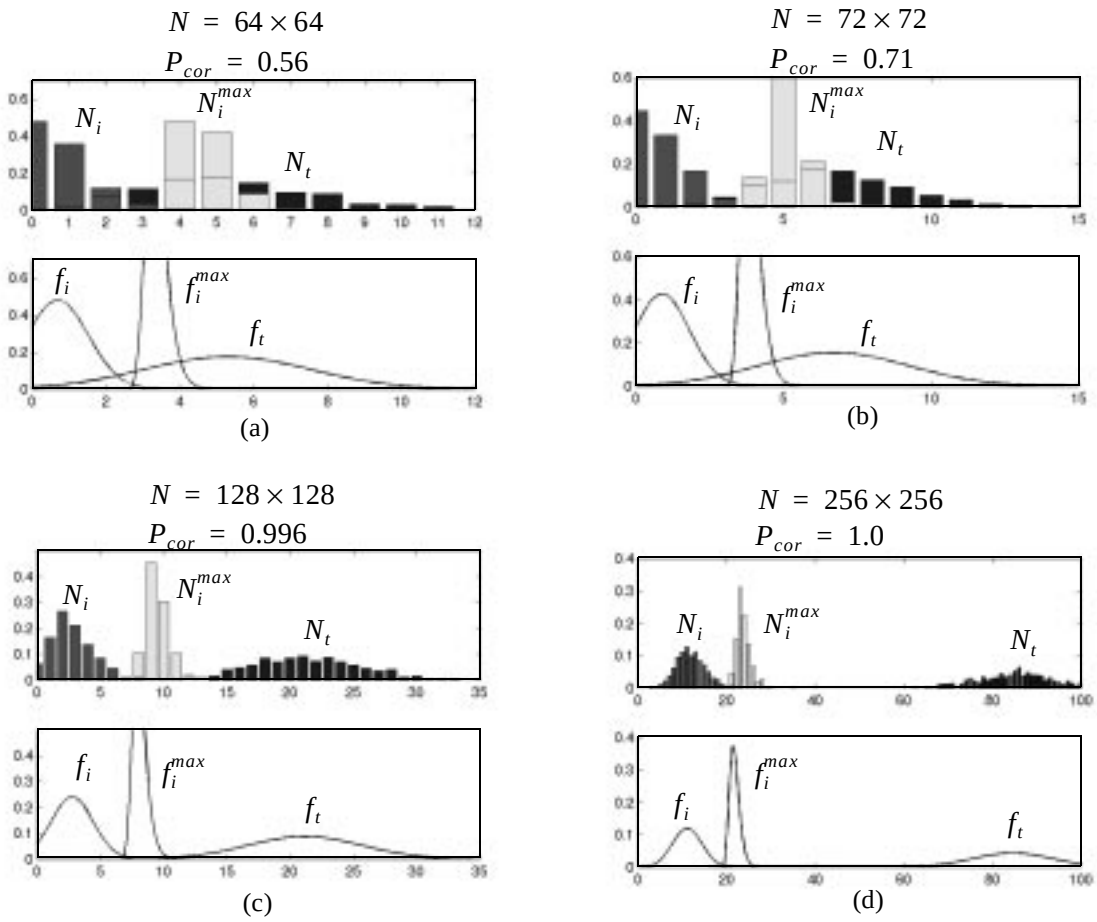


Figure 11. Comparison between simulation and theory for the case of a 4×4 window size and noise of $\rho = 0.4$ and a memory size of $M = 1000$, and an image size of (a) 64×64, (b) 72×72, (c) 128×128, and (d) 256×256.

Figure 11 shows a case where there is overlap between the distributions. Here, the number of images in the memory set is $M = 1000$, the input noise is $\rho = 0.4$, and the local window size is $n = 4 \times 4$. Figure 11(a) shows the results in the case of an image size of $N = 64 \times 64$. Here, the probability of correct retrieval is $P_{cor} = 0.56$. Figure 11(b) shows the results for $N = 72 \times 72$. In this case, the probability of correct retrieval is $P_{cor} = 0.71$. Similarly, (c) and (d) show the results for larger image sizes: $N = 128 \times 128$, and $N = 256 \times 256$. As expected, for a fixed number of memory patterns, as the image size increases, the separation between N_i and N_t increases.

4.2 Probability of Correct Retrieval

Fixing the system dimension at $N = 64 \times 64$ and the noise level at $\rho = 0.4$, the simulations of the previous section were repeated for various values of the number of fundamental patterns *m* from $m = 10$ to $m = 10,000$. Figure 12 shows the probability of correct retrieval vs. the number of stored patterns for various local window sizes. The dashed line gives the simulation result, and the solid lines gives the theoretical values from Equation 13.

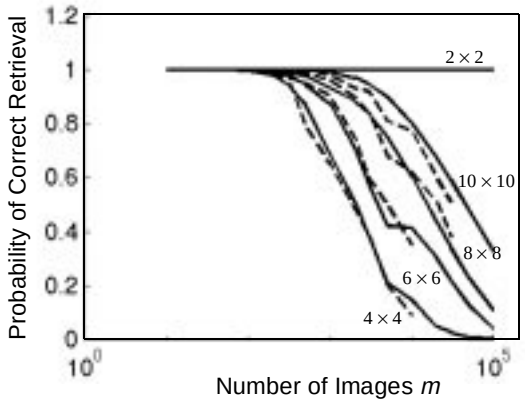


Figure 12. Probability of correct retrieval vs. number of stored patterns *m*.

As noted earlier, the two-level Hamming network reduces to the full Hamming memory in both of the extreme cases for the window size. This is illustrated in Figure 12, where the small 2×2 neighborhood size gives very good performance. As the neighborhood size increases, though, the performance degrades. In particular, the 4×4 window gives the worst performance. By increasing the window size above 4×4, the performance improves.

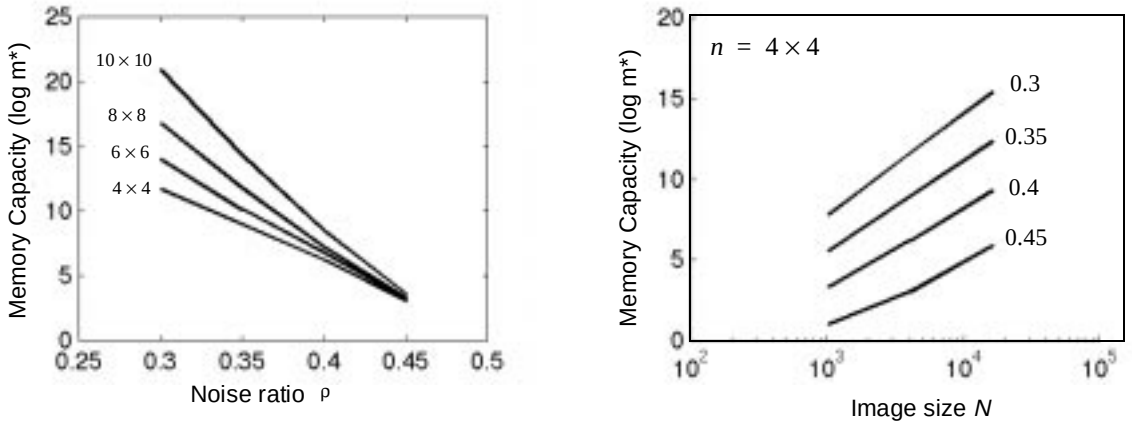
4. 3 Capacity and Error Correction

Capacity is usually defined as the maximum number of patterns that can be stored such that when presented at the input, the memory retrieves the correct pattern. This notion of capacity, though, does not address the issue of error correction. That is, if the input pattern is different from the fundamental pattern by 1 pixel, can the memory still retrieve the correct pattern?

The notion of capacity that will be used here is more stringent: Assuming an input noise level of ρ, how many input patterns can be reliably stored?

Plots of the capacity of the two-level decoupled Hamming memory can be generated as follows. First, values for *n*, *N*, and ρ are fixed. Then, starting with a small value for *m*, P_{cor} is computed using Equation 13. Initially, with such a small *m*, P_{cor} is a near 1.0. As *m* is slowly increased, P_{cor} decreases in value. This process of decreasing *m* is continued until P_{cor} falls below 0.99. In this case, the largest value of *m* which gives $P_{cor} \geq 0.99$ is taken as the capacity m^* of the memory.

Figure 13(a) shows a plot of the capacity of the 2-level decoupled Hamming memory vs. the input noise level for various window sizes: 4×4, 6×6, 8×8, and 10×10. For these experiments, the image size was fixed at $N = 64 \times 64$. As expected, the capacity decreases as the noise increases. Figure 13(b) shows a plot of capacity vs. image size for various values of input noise: $\rho = 0.3$, 0.35, 0.4, and 0.45. Again, the image size is fixed at 64×64.



(a) (b)

Figure 13. (a) Capacity vs. noise level for window sizes: 4×4, 6×6, 8×8, and 10×10. (b) Capacity vs. image size for various noise levels: 0.3, 0.35, 0.4, 0.45.

5. Summary of Experimental Results: Grayscale Face Images

To our knowledge, there has been no systematic study of the memory capacity of associative memory models for human face recognition. One of the reasons for this is that a large enough

database of normalized images does not exist. As part of this project, we collected a large database of faces and used it to study the performance of various associative memory models and classifier systems.

In a practical face recognition scheme, there are two main computational phases. The first phase involves snapping the image and performing any necessary preprocessing computations to make the image suitable for input to the recognition system. Typical preprocessing computations include segmentation of the face part of the image from a larger image, alignment of the face image to eliminate rotation and shift, and intensity and size normalization. The second phase is the recognition task, where the image is classified as either a known individual or else rejected as not known to the system.

Note that most of the sophisticated classification algorithms use template matching in some form or another. Hence, the capacity results that we obtain will provide a baseline measure with which all other algorithms can be compared.

5.1 Collection of a Face Database

In order to focus our research efforts on the associative memory aspect of face recognition (the second phase of the recognition process), we formulated a database of face images which was collected in a laboratory setting under semi-controlled conditions [7]. In this case, a minimal amount of pre-processing is required before the image can be used in an automatic face recognition scheme.

The specifications for the database described in this paper are as follows:

•Two image sets should be collected: a *database*, which can be used for designing the recognition and classification system, and a *test set* which is used to test the system.

•The database and test set should contain several images of each individual, showing different facial expressions.

•All images should be the same size with the face centered in the image.

•The database should contain subjects of different ethnicities, samples of both men and women, and samples from different age groups.

•Variations due to lighting effects, head tilt, shift, rotation, and scaling should be minimized as much as possible.

To eliminate the need for sophisticated preprocessing, a simple apparatus was constructed which fixes the head of the person in the center of the image. Figure 14 shows the apparatus, which consists of a wooden beam mounted on a tripod. Attached to one end of the beam is a frame in which the subject puts his or her face while the picture is snapped. On the other end of the beam is a video camera, which is used to snap the images. The camera is controlled by software, which manages the image snapping process and the process of storing the image in the computer's memory. The dimension of the snapped image is 82×115.

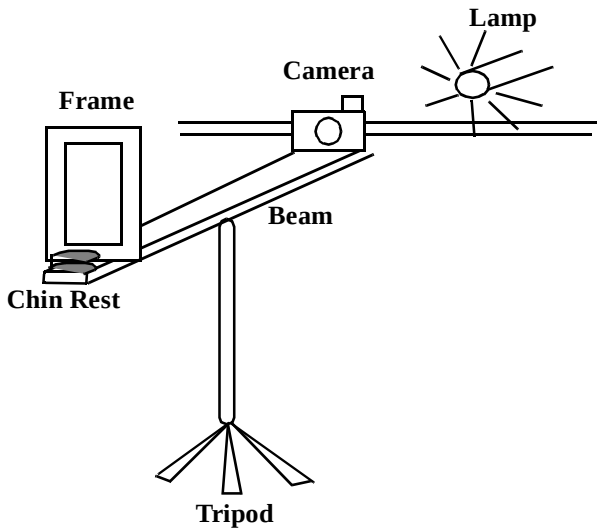


Figure 14. A schematic diagram of the experimental setup which is used to snap the images. The subject sits in a chair (not shown) which is positioned in front of the frame.

Two sets of images were collected: a *database* and a *test set*. In the database, each person is represented by 4 images which show different facial expressions: a blank expression, smile, angry, and surprised. The test set consists of 2 images for each person: a blank facial expression and an arbitrary expression. In the case of the arbitrary expression, the subject is told to try to fool

the system by making an unusual expression. Note that the blank test image is different from the blank memory image. The training set consists of 4 additional blank images of each subject. Hence, for each subject, a total of 10 images are taken.

Images of 200 different people were collected over a 1 year period. In order to study the false positive performance of the system, we separate the 200 people into two 100-person databases: DB-1 and DB-2. Note that all images for a given person were taken on the same day and under roughly the same lighting conditions. The age of the subjects ranged from 13 years old to 50 years old. In addition, samples were taken over a variety of ethnic backgrounds, including European-American, African-American, East Indian, Middle-Eastern, and Asian.

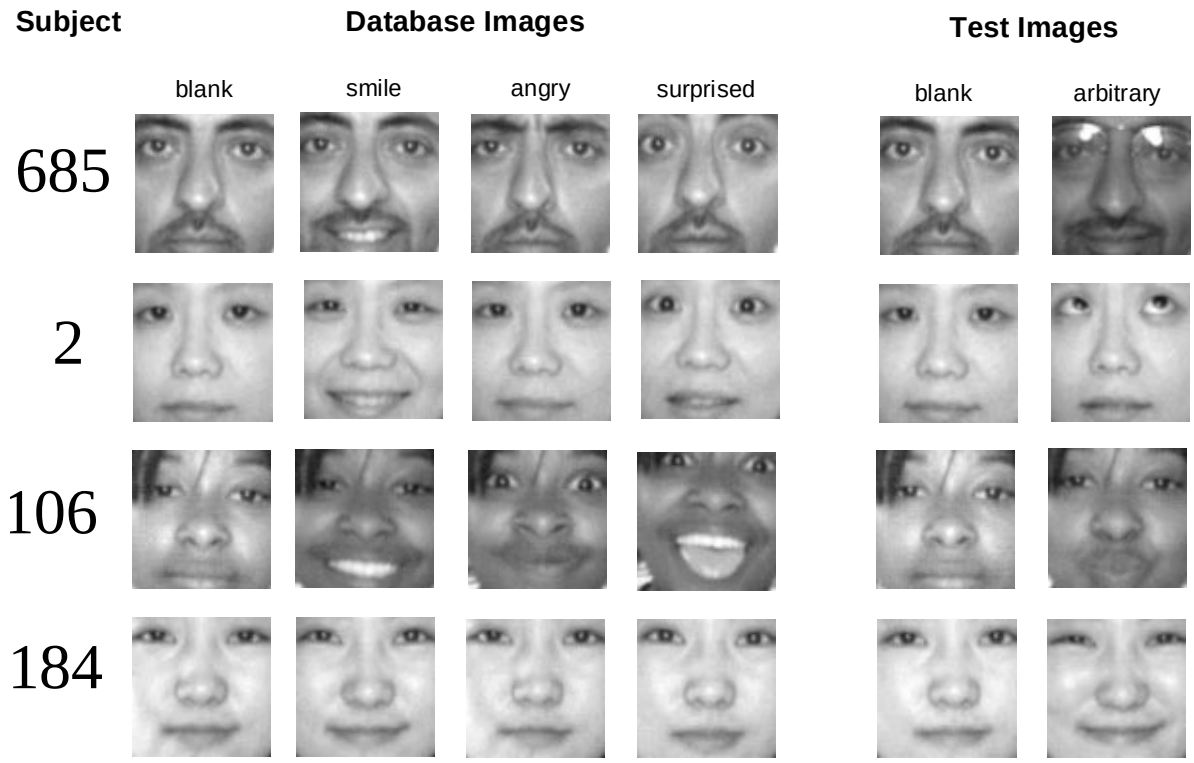


Figure 2. Samples of the 72x72 database and test images for different people (the 4 additional blank images are not shown). In total, there are 200 individuals in the database.

In many face recognition schemes, it is desired that the subject's hair not be part of the image, because the system can be fooled if the subject changes hair style, etc. To eliminate the hair

artifacts, we cropped the 82×115-dimensional images down to 72×72.

5.2 Correct Classification Experiments

Classification and rejection experiments were run for both DB-1 and DB-2 using four different algorithms: a nearest neighbor classifier [6], the two-level decoupled Hamming network [1], eigenfaces [23-24], and a wavelet-based classifier [25]. A 6x6 window size was used by the two-level decoupled Hamming network.

The correct classification results are shown in Table 1(a) for DB-1 and (b) for DB-2. In this case, all of the results are out of 100 blank expression test images and 100 arbitrary expression test images.

DB-1	Correct classification	
	Blank	Arbitrary
Nearest Neighbor	99-0-1	68-0-32
Voting Network	97-0-3	65-0-35
Eigenfaces	87-0-13	47-1-52
Wavelet	99-0-1	64-1-35

(a)

DB-2	Correct classification	
	Blank	Arbitrary
Nearest Neighbor	98-0-2	68-0-32
Voting Network	97-0-3	62-1-37
Eigenfaces	97-1-2	48-1-51
Wavelet	97-0-3	60-0-40

(b)

Table 1. Correct classification performance for 4 different classifiers: a nearest neighbor classifier, voting network, eigenfaces, and wavelets on (a) the DB-1 database and (b) DB-2. In each case, the database consists of 100 people and 4 images per person (different expressions). The results are reported as: C-M-R, where C is the number of images correctly classified, M is the number of images misclassified, and R is the number of images rejected.

For each experiment, 3 numbers are reported: C - M - R, where C indicates the number of images correctly classified, M is the number misclassified, and R is the number rejected, respectively. So when storing DB-1 and testing with the blank test images, the voting network gives 97/100 images correctly classified, 0/100 images misclassified, and 3/100 images rejected.

The test set of arbitrary face expressions is much more difficult, and in this case, the results show 65/100 correctly classified, 0/100 images misclassified, and 35/100 images rejected.

5.3 False Positive Experiments

Table 2 shows the results of the false positive experiments where DB-1 is tested with images from DB-2. In this case, we desire all of the test images to be rejected by the system. Table 3 indicates the number of DB-2 images that were not rejected. For example, the nearest neighbor classifier failed to reject 6 blank expression samples (out of 600 test images), 1 smile image, and 1 angry expression image. This gives a false positive identification rate of 8/1000 = 0.8%. Notice the strong performance of the voting network in rejecting unknown individuals: 4/1000 = 0.4%.

DB-1 rejecting DB-2	False positives					
	Blank	Smile	Angry	Surprised	Arbitrary	Total
Nearest Neighbor	6/600	0	1	0	0	7/1000
Voting Network	2/600	0	2	0	0	4/1000
Eigenfaces	34/600	5	7	4	4	54/1000
Wavelet	21/600	5	3	3	0	32/1000

Table 2. Number of false positive images when DB-1 is used as the memory set and DB-2 is used as the test set.

Finally, Table 3 shows the results when DB-2 is stored and tested with the images from DB-1. The results are similar to those in Table 3.

These results are encouraging and show that the two-level decoupled Hamming network performs well in comparison to some other standard pattern classification techniques, such as the eigenfaces method and the classical nearest neighbor method.

	False positives					
	Blank	Smile	Angry	Surprised	Arbitrary	Total
Nearest Neighbor	6/600	1	1	0	0	8/1000
Voting Network	3/600	2	0	1	0	6/1000
Eigenfaces	37/600	3	5	5	4	54/1000
Wavelet	33/600	6	6	2	3	50/1000

Table 3. Number of false positive images when DB-2 is used as the memory set and DB-1 is used as the test set.

6. List of Publications Resulting from this Grant

6.1 Publications in Refereed Journals

Ikeda, N., Watta, P., Artiklar, M, and Hassoun, M., (2001). "A Two-level Hamming Network for High Performance Associative Memory," Neural Networks, 14, 1189-1200.

P. Watta and M. H. Hassoun, (2001). "Generalizations of the Hamming Associative Memory," *Neural Processing Letters*, 13(2), 183-194.

N. Ikeda, P. Watta, and M. H. Hassoun, (1999). "Performance of the Two-Level Parallel Hamming Associative Memory," *Transactions of the Institute of Electronics, Information, and Communications Engineers D-II*, **J82**(9), 1528-1532 (in Japanese).

P. Watta, K. Wang, and M. H. Hassoun, (1997) "Recurrent Neural Nets as Dynamical Boolean Systems with Application to Associative Memory," *IEEE Transactions on Neural Networks*, **8**(6), pp. 1268-1280.

6.2 Publications in Refereed Conference Proceedings

X. Mu, Mehmet Artiklar, Metin Artiklar, P. Watta, and M. Hassoun (2001). "A Training Algorithm for Robust Face Recognition," *Proceedings of the International Joint Conference on Neural Networks*, July 14-19, 2001, Washington, DC., 2877-2882.

M. Artiklar, A. Masadeh, M. H. Hassoun, and P. Watta, (2000). "The Effect of Expression in a Database of Face Images," *Proceedings of the 43rd Midwest Symposium on Circuits and Systems*, August 8-11, 2000, Lansing, MI.

P. Watta, M. Artiklar, A. Masadeh, and M. H. Hassoun, (2000). "Construction and Analysis of a Database of Face Images which Requires Minimal Preprocessing," *Proceedings of the IASTED Conference on Modeling and Simulation MS'2000*, May 15-17, 2000, Pittsburg Pennsylvania, 465-469.

P. Watta, N. Ikeda, M. Artiklar, A. Subramanian, and M. H. Hassoun, (1999). "Comparison Between Theory and Simulation for the 2-Level Decoupled Hamming Associative Memory," *Proceedings of the IEEE International Conference on Neural Networks*, IJCNN'99, July 10-16, 1999, Washington, DC., Paper #JCNN 0337.

M. Artiklar, M. Hassoun, and P. Watta, (1999). "Application of a Post-processing Algorithm for Improved Human Face Recognition," *Proceedings of the IEEE International Conference on Neural Networks*, IJCNN'99, July 10-16, 1999, Washington, DC., Paper #JCNN 2166.

N. Ikeda, P. Watta, and M. H. Hassoun, (1998). "Capacity Analysis of the Two-Level Decoupled Hamming Associative Memory," *Proceedings of the International Joint Conference on Neural Networks*, IJCNN'98, May 4-9, 1998, Anchorage, Alaska, pp. 486-491.

P. B. Watta, M. Akkal, and M. H. Hassoun, (1997). "Decoupled Voting Hamming Associative Memory Networks" *Proceedings of the IEEE International Conference on Neural Networks*, ICNN'97, June 9-12, 1997, Houston, Texas, pp. 1188-1193.

7. List of Students Supported from This Grant

Metin Artiklar: Research Assistant and Ph. D. Candidate. He will be completing his Ph. D. dissertation in August 2002. His dissertation is entitled: "Capacity Analysis of Voting Networks with Application to Human Face Recognition"

Xiaoyan Mu: Ph. D. Candidate.

Mehmet Artiklar: Ph. D. student

Ambiga Subramanian: Student assistant

Ahmed Masadeh: Student assistant

Mohammed Akkal: Student assistant

8. Summary and Future Work

Strictly speaking, the capacity of the two-level Hamming memory is $m^* = 2^N$. That is, the memory can store and exactly retrieve all 2^N patterns when presented with the perfect (uncorrupted) pattern at the input. Compare this result to the well known capacity result for the Hopfield memory $m^* = 0.15N$. Of course the price we pay for this much larger capacity is in terms of storage and retrieval time.

The two-level Hamming memory requires that all *m* fundamental patterns be stored, whereas the Hopfield network requires storage of an $N \times N$ weight matrix. There are applications, however, where the two-level Hamming memory requires less storage than the Hopfield weight matrix. For example, suppose we want to store a 512×512 image of each of the 9,000 undergraduate students at the University of Michigan-Dearborn. In this case, the two-level

decoupled Hamming memory requires about 2.5 GB of storage (of size character), whereas the Hopfield weight matrix requires the storage of about 68×10^9 weights. Since weights are usually stored as floating point numbers, the Hopfield weight matrix would require about 136 GB of memory (assuming 2 bytes for each floating point number).

Clearly, whenever $mN < N^2$, the two-level decoupled Hamming memory requires less storage than a single layer neural associative neural memory (ANM). Of course, the single layer ANM is the simplest network architecture possible, and more sophisticated multilayer associative neural memories usually require considerably more than N^2 weights.

The notion of capacity discussed in this paper focused on error correction capability, which is a more practical concept than the typical definition of capacity. This paper demonstrated that the two-level decoupled Hamming memory provides a large amount of error correction.

In future work, we plan to continue and extend our associative memory model work in the area of face recognition in the following directions:

1. Improve the process and apparatus used to collect face images.

2. Collect a larger database of normalized face images.

3. Improve the performance of the associative memory and classification algorithms.

4. Measure the performance of various associative memory models and classification systems as a function of how many people and images are stored in the database and for 3 different types of experiments: *Correct classification experiments*. Measure the ability of the system to correctly classify test images of people who are in the database. *False Positive experiments.* Measure the ability of the system to reject individuals who are not in the database. *Classification over time experiments.* Measure the ability of the system to correctly classify some of the individuals in the database over an extended period of time, say 6 months or a year.

5. Formulate a notion of capacity of associative memory in terms of results of the above 3 experiments, and using simulations, determine the capacity of various associative memory models and classification systems in the presence of these highly correlated memory patterns.

References

1. Ikeda, N., Watta, P., Artiklar, M, and Hassoun, M., (2001). "A Two-level Hamming Network for High Performance Associative Memory," *Neural Networks*, 14, 1189-1200.

2. P. Watta and M. H. Hassoun, (2001). "Generalizations of the Hamming Associative Memory," *Neural Processing Letters*, 13(2), 183-194.

3. N. Ikeda, P. Watta, and M. H. Hassoun, (1999). "Performance of the Two-Level Parallel Hamming Associative Memory," *Transactions of the Institute of Electronics, Information, and Communications Engineers D-II*, **J82**(9), 1528-1532 (in Japanese).

4. P. Watta, K. Wang, and M. H. Hassoun, (1997) "Recurrent Neural Nets as Dynamical Boolean Systems with Application to Associative Memory," *IEEE Transactions on Neural Networks*, **8**(6), pp. 1268-1280.

5. X. Mu, Mehmet Artiklar, Metin Artiklar, P. Watta, and M. Hassoun (2001). "A Training Algorithm for Robust Face Recognition," *Proceedings of the International Joint Conference on Neural Networks*, July 14-19, 2001, Washington, DC., 2877-2882.

6. M. Artiklar, A. Masadeh, M. H. Hassoun, and P. Watta, (2000). "The Effect of Expression in a Database of Face Images," *Proceedings of the 43rd Midwest Symposium on Circuits and Systems*, August 8-11, 2000, Lansing, MI.

7. P. Watta, M. Artiklar, A. Masadeh, and M. H. Hassoun, (2000). "Construction and Analysis of a Database of Face Images which Requires Minimal Preprocessing," *Proceedings of the IASTED Conference on Modeling and Simulation MS'2000*, May 15-17, 2000, Pittsburg Pennsylvania, 465-469.

8. P. Watta, N. Ikeda, M. Artiklar, A. Subramanian, and M. H. Hassoun, (1999). "Comparison Between Theory and Simulation for the 2-Level Decoupled Hamming Associative Memory," *Proceedings of the IEEE International Conference on Neural Networks*, IJCNN'99, July 10-16, 1999, Washington, DC., Paper #JCNN 0337.

9. M. Artiklar, M. Hassoun, and P. Watta, (1999). "Application of a Post-processing Algorithm for Improved Human Face Recognition," *Proceedings of the IEEE International Conference on Neural Networks*, IJCNN'99, July 10-16, 1999, Washington, DC., Paper #JCNN 2166.

10. N. Ikeda, P. Watta, and M. H. Hassoun, (1998). "Capacity Analysis of the Two-Level Decoupled Hamming Associative Memory," *Proceedings of the International Joint Conference on Neural Networks*, IJCNN'98, May 4-9, 1998, Anchorage, Alaska, pp. 486-491.

11. P. B. Watta, M. Akkal, and M. H. Hassoun, (1997). "Decoupled Voting Hamming Associative Memory Networks" *Proceedings of the IEEE International Conference on Neural Networks*, ICNN'97, June 9-12, 1997, Houston, Texas, pp. 1188-1193.

12. Hamming, R. (1986). *Coding and Information Theory*. Prentice-Hall, Englewood Cliffs, NJ.

13. Hassoun, M. H., ed. (1993). *Associative Neural Memories: Theory and Implementation*. Oxford University Press, New York.

14. Hassoun, M. H. (1995). *Fundamentals of Artificial Neural Networks*, MIT Press, Cambridge, Mass.

15. M. H. Hassoun and P. B. Watta, (1996). "The Hamming Associative Memory and its Relation to the Exponential Capacity DAM," *Proceedings of the IEEE International Conference on Neural Networks*, ICNN'96, June 3-6, Washington, D.C., pp. 583-587.

16. K. Siu, V. Roychowdhury, and T. Kailath, (1995). *Discrete Neural Computation: A Theoretical Foundation*, Prentice-Hall, Englewood Cliffs, New Jersey.

17. S. Wolfram, *Cellular Automata and Complexity: Collected Papers*, (1994). Addison-Wesley, Reading, Massachusetts.

18. C. Bishop, *Neural Networks for Pattern Recognition*, (1995). Oxford University Press, New York.

19. T. Ho, J. Hull, J., and S. Srihari, (1994). "Decision Combination in Multiple Classifier System," *IEEE Transactions on Neural Networks*, **16**(1), 66-75.

20. Hopfield, J., (1984). "Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons," *Proc. Natl. Acad. Sci., USA, Biophys*. **81**, 3088-3092.

21. V. Pulkki, and H. Taneli, (1996). "An Implementation fo the Self-Organizing Map on the CNAPS Neurocomputer," *Proceedings of the IEEE International Conference on Neural Networks*, ICNN'96, June 3-6, Washington, D.C., pp. 1345-1349.

22. Port, S. (1994). *Theoretical Probability for Applications*, John Wiley and Sons, New York.

23. Turk, M., and Pentland, A. (1989). "Face Processing: Models for Recognition," *Proceedings of the SPIE Intelligent Robots and Computer Vision VIII: Algorithms and Techniques*, Vol. **1192**, 22-32.

24. Turk, M., and Pentland, A. (1991). "Eigenfaces for Recognition," *J. Cognitive Neuroscience*, **3**(1), 71-86.

25. Jacobs, C., Finkelstein, A., and Salesin, D. (1995). "Fast Multiresolution Image Querying," Proceedings of SIGGAPH 95, in *Computer Graphics Proceedings, Annual Conference Series*, pp. 277-286, August 1995, Los Angeles, CA.