Neural Processing Letters (2006) 23:257–271
DOI 10.1007/s11063-006-9001-1

An RCE-based Associative Memory with Application to Human Face Recognition

XIAOYAN MU[1], MEHMET ARTIKLAR[2], PAUL WATTA[3]
and MOHAMAD H. HASSOUN[2]

[1] *Department of Electrical and Computer Engineering, Rose-Hulman Institute of Technology, Terre Haute, IN 47803, USA. e-mail: mu@rose-hulman.edu*

[2] *Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, USA. e-mail: mcayan@hotmail.com, hassoun@eng.wayne.edu*

[3] *Department of Electrical and Computer Engineering, University of Michigan-Dearborn, Dearborn, MI 48128, USA. e-mail: watta@umich.edu*

Abstract. Many models of neural network-based associative memory have been proposed and studied. However, most of these models do not have a rejection mechanism and hence are not practical for many real-world associative memory problems. For example, in human face recognition, we are given a database of face images and the identity of each image. Given an input image, the task is to associate—when appropriate—the image with the corresponding name of the person in the database. However, the input image may be that of a stranger. In this case, the system should reject the input. In this paper, we propose a practical associative memory model that has a rejection mechanism. The structure of the model is based on the restricted Coulomb energy (RCE) network. The capacity of the proposed memory is described by two measures: the ability of the system to correctly identify known individuals, and the ability of the system to reject individuals who are not in the database. Experimental results are given which show how the performance of the system varies as the size of the database increases—up to 1000 individuals.

Key words. associative memory, neural network, automated human face recognition, capacity, image processing, RCE, false acceptance, database

1. Introduction

In the associative memory problem, we are given a memory set of M prototype patterns or input-output pairs of the form $\{(\mathbf{x}_i, \mathbf{y}_i) : i = 1, 2, \ldots, M\}$, where $\mathbf{x}_i \in R^N$, $\mathbf{y}_i \in R^p$. The goal is to design a system which can store each of the $(\mathbf{x}_i, \mathbf{y}_i)$ associations in memory so that when pattern $\mathbf{x}_i$ is presented as input (the memory key), the system reliably retrieves the pattern $\mathbf{y}_i$. In addition, the system should retrieve $\mathbf{y}_i$ even when $\mathbf{x}_i$ is corrupted with various types of noise. But when the memory key is not sufficiently close to any of the stored prototypes, the system should reject the input as being noise [7].

Human face recognition can be seen as an associative memory problem whereby the memory set consists of a database of face images paired with the identity (name) of each individual. The input key is then a two-dimensional face image, and the retrieved pattern should be one of the following:

(1) the name of the person represented by the image
(2) reject state (either the input image is not a face, or else, it is a face of an individual who is not in the database).

Many face recognition systems rely on template matching in order to determine if the input should be associated with one of the known individuals or rejected. For example, in a typical eigenface approach [2, 13], both the memory set images and the input image are mapped onto a lower dimensional eigenspace. But once that mapping is done, template matching is used to match the vector of eigen coefficients of the input to the closest prototype vector of eigen coefficients. In the Gabor features-based approach of Wiskott et al. [17], local features are extracted from both the memory set images and the input image, and local distance calculations are made. A global distance measure between the memory pattern and the input is obtained by summing the local distances [17, 18]. A local template matching approach which uses a voting mechanism to combine the results of the local computations was proposed in Ikeda et al. [8].

In its simplest form, the design of a template matching-based system involves storing all of the fundamental input-output pairs $(\mathbf{x}_i, \mathbf{y}_i)$ in the computer's memory. Then for a given input key $\mathbf{x}$, the system simply determines the closest matching stored input pattern $\mathbf{x}_i$ and then outputs the corresponding output pattern $\mathbf{y}_i$. Such a template matching approach can be modified to provide for a reject or no-decision state when the input pattern is not sufficiently close to any of the stored prototypes [15], as well as modified to allow for local and parallel distance computations [8].

Of course, to determine how close two patterns are, a suitable metric is needed. Many distance functions have been proposed and used in the literature [9, 16, 17]. The associative memory model developed in the next section is quite flexible and can be used with any suitable distance measure.

The performance of an associative memory is measured by its *capacity*, which is how many patterns can be stored in the system. Various definitions of capacity have been proposed in the neural network literature [6]. Most definitions, though, only measure how well the system can retrieve the correct pattern when presented with one of the memory keys (or distorted version of one of the keys) of the fundamental memory set. In this paper, we define capacity as a measure of how many patterns can be *reliably* stored in the system. In particular, following the methodology used for face recognition [11], we will measure capacity by measuring the results on two different types of experiments:

1. **Correct classification experiments**. Measure the ability of the system to correctly recognize and identify individuals who are in the database.
2. **False acceptance experiments.** Measure the ability of the system to reject all images of individuals who are not part of the database.

In this paper, we develop an associative memory model similar in structure to the restricted Coulomb energy (RCE) network. We will test the performance of the proposed associative memory on a difficult and practical problem: human face recognition. We study how the performance of the system scales as a function of the size of the memory set. The contribution of this paper is to present an associative memory model that can retrieve the correct pattern (when appropriate), but also reject patterns with very low signal to noise ratio.

2. An RCE-type Associative Memory

In this section we describe a restricted Coulomb energy (RCE)-type neural network which operates as an associative memory suitable for the face recognition problem.

2.1. ARCHITECTURE

Suppose that the database of face images DB contains K different face image samples of each of the M individuals (for example, different facial expressions or lighting conditions, etc.):

$$\mathbf{DB} = \{(\mathbf{x}_{mk}, ID_m) \colon m = 1, 2, \dots, M;\ \ k = 1, 2, \dots, K\}$$

Here, $\mathbf{x}_{mk}$ is the kth image (or feature vector) of person m, and ID_m is the name of person m.

Typically, the RCE network is a unit allocating system whereby the number of units is increased as training proceeds [5, 12]. We will simplify the training by just allocating one hidden unit for each prototype image in the database. Effectively, this network acts as a template matching system (with K prototypes per class) with rejection capabilities. The network structure is shown in Figure 1.

Associated with each hidden layer neuron in Figure 1 is a threshold or radius which is used to determine whether the input is sufficiently close to the prototype so that the unit should fire. The output z_{mk} of the mkth hidden unit (with radius r_{mk}) is given by:

$$z_{mk} = \begin{cases} 1 & \text{if } d(\mathbf{x}_{mk}, \mathbf{x}) \leqslant r_{mk} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The outputs y_m are obtained by ORing together the outputs of the hidden units:

$$y_m = z_{m1} \vee z_{m2} \vee \cdots \vee z_{mK} \tag{2}$$

That is, $y_m = 1$ if any of the hidden units $z_{m1}, z_{m2}, \dots, z_{mK}$ fire; otherwise $y_m = 0$. For an input key of say the mth person, we want the output vector $\mathbf{y}$ to consist of all 0s except for a single 1 at the mth position.

One of the advantages of using locally-tuned hidden units (as in an RBF or RCE network) is that the system is able to reject outliers [7]. Here, the proposed

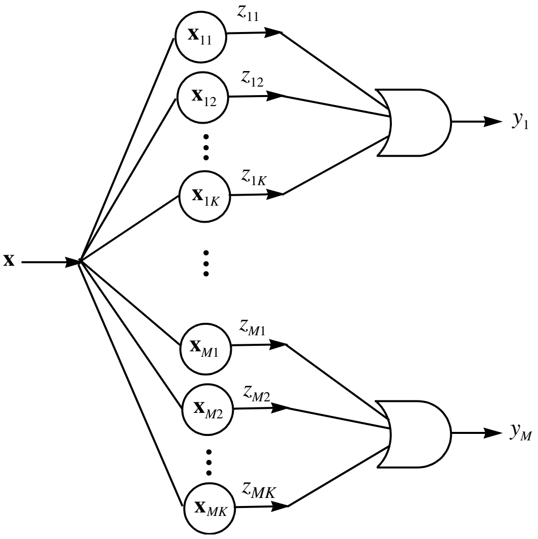

Figure 1. Architecture of the RCE network.

system will reject the input in the following cases: (1) More than 1 output unit fires, and (2) none of the output units fire.

The only tunable parameters of the RCE associative memory are the hidden unit radii. The next section gives one way to compute suitable values for these parameters from the given fundamental memory set (training set).

2.2. TRAINING

The radius r_{mk} gives a measure of how large a region the mkth hidden unit covers in the underlying input space (space of all memory keys). The goal is to set r_{mk} as large as possible, but not too large, otherwise the system will reject a large number of inputs (because multiple units will likely fire). We propose a simple method for computing the hidden unit radii from the given fundamental memory patterns. For the mkth hidden unit, we compute the (between-class) distances between $\mathbf{x}_{mk}$ and all the other images in the given database (excluding the images of person m):

$$d_{ij} = \{d(\mathbf{x}_{mk}, \mathbf{x}_{ij}) : i = 1, \dots, M;\ \ i \neq m;\ \ j = 1, \dots, K\} \tag{3}$$

And then choose the minimum such distance

$$d_{i^*j^*} = \min\{d_{ij}\} \tag{4}$$

The number $d_{i^*j^*}$ represents the largest radius that unit mk can have, because at that radius, another (incorrect) unit (namely, the i^*j^* unit) will also fire. Hence we set the radius r_{mk} as a fraction of that radius:

$$r_{mk} = \alpha d_{i^*j^*} \tag{5}$$

where $0 < \alpha < 1$. Later, we will show that α can be used as a system parameter to adjust the trade-off between correct classification performance and false acceptance performance.

3. Shift Invariance

Noise is always present in any practical image capturing system. Unfortunately, template matching-based computations are sensitive to, and can be adversely affected by, even small amounts of noise. For example, Figure 2 shows the distance between two identical face images as the first image is shifted in various directions (up and down; left and right) and compared to the first. Notice that even a 1-pixel shift in the image can yield a large change in the distance.

There are several ways to make the RCE-based network less prone to errors in small shifts (translations) and rotations in the image. For example, in the elastic graph matching technique [3], the distance between the input and each of the database images is computed on a rectangular grid of points and neighbors of the grid points. The collection of grid points which yields the smallest distance is chosen (deformed template).

A simple and computationally efficient method to provide invariance to small amounts of image shift was given in [1]. Here, to compute the distance between two images, **s** and **t**, an iterative procedure is used to optimize the alignment between the two images. First, we select a center subwindow from each image, as shown in Figures 3(a) and (b). Call these center sub windows $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{t}}$. The distance between the two windows $d(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})$ is computed and recorded as the current best distance. Next, the position of the window $\tilde{\mathbf{s}}$ is held fixed, but the position of the window $\tilde{\mathbf{t}}$ is varied. In particular, the location of $\tilde{\mathbf{t}}$ is shifted by 1 pixel in 4 directions:

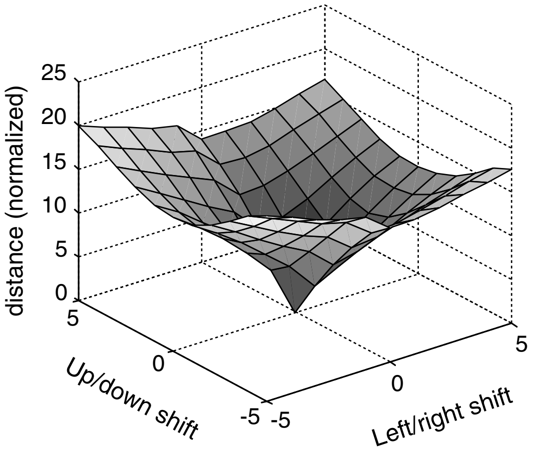

Figure 2. The distance between two identical images as the first image is shifted in various directions (measured in pixels) and compared to the second.

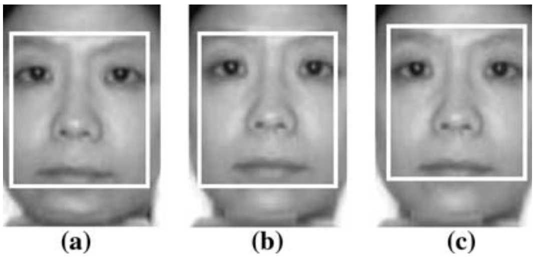

Figure 3. Shift optimization procedure to compute the distance between two images: **s** shown in (a) and **t** shown in (b). After the optimization, the window has moved slightly up and to the right, as shown in (c).

up, down, left, and right. In each case, the resulting distance with $\tilde{\mathbf{s}}$ is computed. If the smallest of these 4 distances is smaller than the best recorded distance, then we shift $\tilde{\mathbf{t}}$ in that direction, update the best recorded distance, and repeat the process. The process terminates when none of the 4 distances yield a smaller value than the best recorded distance. In addition, the process terminates when a maximum number of shifts have occurred. In the experiments reported here, a maximum of 5 steps are used. In Figure 3, the final position of the window in (b) is shown in (c). Notice that the window moved slightly up and to the right.

The nature of the optimization problem and the proposed strategy can best be seen by viewing an exhaustive map of the distances between the two windows as a function of window position. Such a map is given in Figure 4. In this table, each number represents the (normalized) distance between the two windows as a

		Horizontal shift								
		-4	-3	-2	-1	0	+1	+2	+3	+4
Vertical shift	-4	29.4	28.5	27.5	27.3	27.3	28.5	29.4	30.5	31.5
	-3	27.9	27.0	25.8	25.4	25.6	19.4	15.8	29.1	30.6
	-2	27.0	25.4	24.2	23.5	23.7	18.0	10.9	10.1	29.9
	-1	25.8	23.7	22.0	21.4	21.4	12.4	11.6	16.8	29.2
	0	25.0	22.3	19.6	18.0	17.7	13.9	19.3	23.2	27.6
	+1	24.0	21.2	17.5	17.8	17.8	18.0	19.9	21.7	25.1
	+2	23.7	19.7	18.4	19.2	18.6	18.6	18.1	20.2	23.2
	+3	24.2	20.8	19.1	19.9	19.8	19.9	24.1	25.3	31.9
	+4	25.4	22.8	19.8	22.4	24.0	26.2	29.0	28.5	37.9

Figure 4. This diagram shows the city-block distance (normalized) between a database image and shifted versions of an input image.

function of window position. The center of the table (with value 17.7) is the distance between the two windows at their default positions (no shift applied). As $\tilde{\mathbf{t}}$ is shifted 1 pixel to the right, the distance drops to 13.9. From there, shifting the input image one pixel up results in a distance of 12.4. This process continues until the distance can no longer be decreased by shifting in any of the 4 surrounding directions (or until a maximum number of steps has been taken). Here, the shifting process decreased the distance between the two images from 17.7 to 10.1 in five steps.

Of course, in template matching-based recognition, we desire a metric which gives small distances for like patterns and large distances for dissimilar patterns. The above shift optimization procedure aids in reducing the distance between two images of the same person. Unfortunately, this process can also reduce the distance between images of different people. The conjecture here (supported by extensive simulations) is that even though the shifting process decreases the distance between different people, it tends to do so by a lesser amount than the distance improvement for images of the same person. Hence, there is an overall increase in the discrimination ability of the metric.

The shift processing algorithm is used in computing the distance between images both during the training phase and the recall phase. Note that it is pointless to try to shift images that are very dissimilar from each other. Hence, to reduce computation time, we only allow the top 40 matches to participate in the shifting process. The shift processing algorithm occurs right before the hidden layer stage (see Figure 1). Hence, for a given input image $\mathbf{x}$, all distances $\hat{d}(\mathbf{x}_{mk}, \mathbf{x})$ are computed. The 40 such smallest distances are selected and shift-optimization against each of these 40 closest units to obtain a (possibly) smaller set of distances: $d(\mathbf{x}_{mk}, \mathbf{x})$. From there, Equation 1 is used, and the output is computed.

The above shift processing algorithm helps to mitigate the effects of small rotations, as well as translations. Further discussion of this topic can be found in Mu [10].

4. The Database of Face Images

A detailed discussion of the construction of the face database that is used in these experiments can be found in [14]. Briefly, the database consists of 1000 different men and women of various ages (between 15 and 65 years old). The images were collected at an urban University and hence the database contains samples from a variety of ethnic backgrounds. All images have been normalized to a uniform size and scale. For each person, there are four training images that show different facial expressions: blank, smile, angry, and surprised. There are four additional blank expression training images available for each person (that may or may not be used in the database). To test the system, an additional blank expression image is available for each person in the database. Note that the database and test images were captured on the same day and under the same conditions.

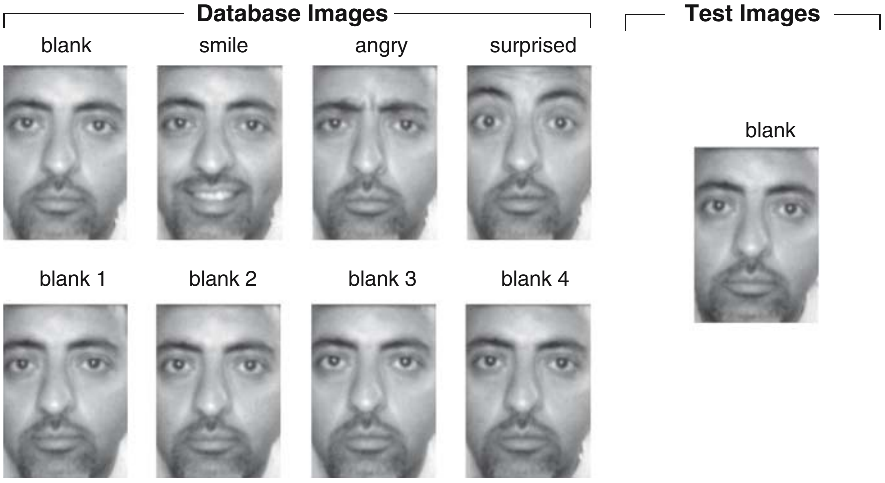

Figure 5. Sample 82 × 115 images of one of the subjects in the database. In total, there are 1000 individuals in the database.

Both the database and test images were snapped at a dimension of 82 × 115 and stored as 8-bit gray scale (256 levels). The shift optimization is performed on a sub window size of 72 × 72 (centered in the image). Figure 5 shows a set of 82 × 115 sample images for one of the subjects in the database.

5. Experimental Results

As mentioned above, the RCE architecture can be used with any distance function. In the results that follow, we use a simple city-block distance function. Here, the distance between two N-dimensional vectors $\mathbf{s}$ and $\mathbf{t}$ is given by:

$$d(\mathbf{s},\mathbf{t}) = \sum_{i=1}^{N} |s_i - t_i|$$

Unless otherwise state, the fundamental memory set consists of the four facial expression images (blank, smile, angry, and surprised) of each of the M known individuals. So there's a total of $4M$ images in the database, and hence there's $4M$ hidden units in the RCE network. In the training phase, the database images are used to determine the radius $r_{mk},\ m = 1, 2, \ldots, M; k = 1, 2, 3, 4$, for each of the hidden units.

In order to assess the performace of the proposed RCE model, we will consider two types of features: simple pixel features and eigenface features. In the case of the pixel features, the patterns $\mathbf{x}_{mk}$ in the memory set (and hidden units) are just the images themselves. In the case of eigenface features, we first compute eigenfaces over the given image database using the standard approach [13]. The patterns

$\mathbf{x}_{mk}$ are then the eigen coefficients representing each database image. In either case, the computation for training and recall are identical and given by Equations 1–5.

As mentioned above, two measures are used to assess system performance. First, in the *correct classification experiments*, we measure the ability of the system to correctly identify known people. Here, the input is a test set image. Three possibilities can occur here: (1) the input image is rejected, (2) the input is classified correctly, and (3) the input is associated with the wrong person (misclassified). For these experiments, we will report the percent rejection and the percent error (misclassification). Note that the percent correct classification and the percent error sum to 100%, hence percent correct classification is easily determined from the given quantities.

The second measure is the ability of the system to reject unwanted inputs. In these *false acceptance experiments*, we test the systems with images of people who are not in the stored database. There are only 2 possible outputs here: (1) the input is properly rejected, or (2) the input is incorrectly identified with one of the people in the database. Since these two quantities sum to 100%, we will only report the second: % error.

5.1. CORRECT CLASSIFICATION EXPERIMENTS

Table 1 shows the results of the correct classification experiments for the RCE-based associative memory as the memory set size scales up from $M = 100$ to $M = 1000$. Note that each number in Table 1 (except for the $M = 1000$ case) is an average over five different trials. In each trial, a different database was (randomly) chosen from the available 1000 individuals. The threshold scaling parameter was set at $\alpha = 0.8$.

The results show that the memory with pixel features scales well and gives an error rate of 0.1% (or less) with a reject rate of less than 5%. The performance

Table 1. Classification results for the blank facial expression test set for various number of people in the database.

Num People M	Pixel Features		Eigenfaces	
	% Reject	% Error	% Reject	% Error
100	3.4	0	2	0
200	2.7	0	5.5	0.5
300	2.7	0	5.5	1.5
400	3.4	0	5.5	2
500	3.5	0	5.5	2.5
600	4.0	0.1	5.6	2.5
700	3.5	0.1	6.7	2
800	4.2	0.1	6.2	2.2
900	4.2	0.1	7.6	2.1
1000	4.5	0.1	7.8	2

using eigenface features is not as good as the pixel features due to the loss of information from the dimensionality reduction.

5.2. FALSE ACCEPTANCE EXPERIMENTS

Next we ran false acceptance experiments and tested the system with 200 randomly chosen individuals who are not present in the database to see how well the system rejects these images. In this case, we used all available training and test images (9 images total) for each test person. Note that we could not perform this experiment on the $M=1000$ database due to lack of additional images, and could only test with 100 people for the $M=900$ database.

The results of the false acceptance experiments are shown in Table 2. Curiously, for the pixel features, the number of false matches decreases as the number of people in the database increase, and is only 0.4% for $M=800$ and $M=900$. The reason for this is that as more images are added to the database, the hidden layer thresholds necessarily become tighter as Equations 2 and 3 are computed over more images. Another reason is that as more and more images are added to the database, it is more likely that more than one output unit will fire. But, by our simple decision rule, if more than one output fires, we reject the input.

In the remaining experiments, we will focus on just the pixel features.

5.3. THE CORRECT CLASSIFICATION/FALSE ACCEPTANCE TRADE-OFF

A plot of both the classification (rejection) and false acceptance results is shown in Figure 6.

From this figure, we see that the general trend in rejection performance increases linearly (after $M=200$). We did not plot the number of misclassified images, since it was so low: there was at most one misclassified image for each test. However, it is expected that as M increases, the number of misclassified images will also begin to rise. A larger database is needed in order to perform such experiments.

Table 2. False acceptance rate (FAR) as a function of memory size.

Num People M	Pixel Features % FAR	Eigenface Features % FAR
100	2.9	5
200	1.6	5.5
300	1.1	5.3
400	1.0	6.7
500	1.2	6.2
600	0.7	5.3
700	0.6	5.2
800	0.4	5.3
900	0.4	5.3

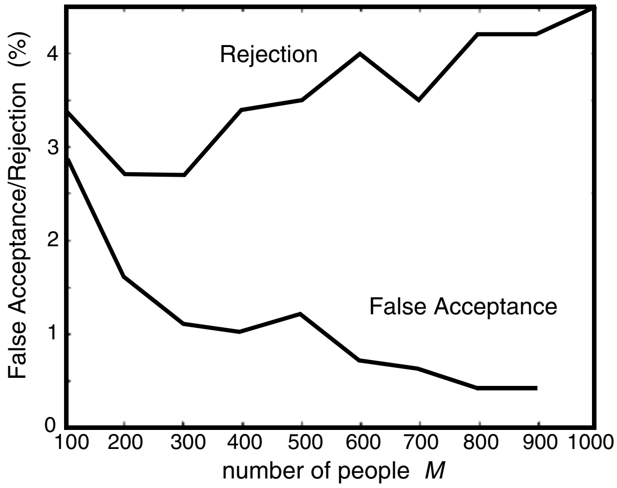

Figure 6. Results of the rejection performance and false acceptance performance of the system as a function of database size.

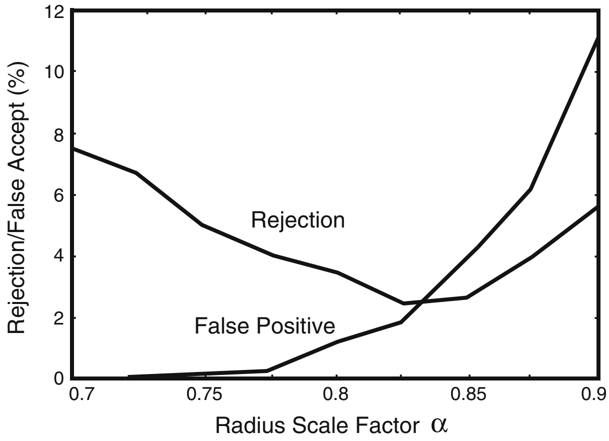

Figure 7. Rejection and false acceptance as a function of α.

5.4. THE EFFECT OF RADIUS SIZE α

For all the results given above, a threshold scaling parameter of $\alpha = 0.8$ was used. Figure 7 shows how the rejection and false acceptance results vary as a function of α. Of course to get good false acceptance performance, we require a very small value for α. But setting α too low causes a lot of rejections (in the correct classification experiments) because the condition for a hidden unit to fire (Equation 1) is too stringent. On the other hand, setting α too high also causes a lot of rejections because it is more likely that more than one output unit will fire.

To analyze the rejection performance of the system, we measured the number of rejections due to the fact that *no hidden units were activated*, as well as the number

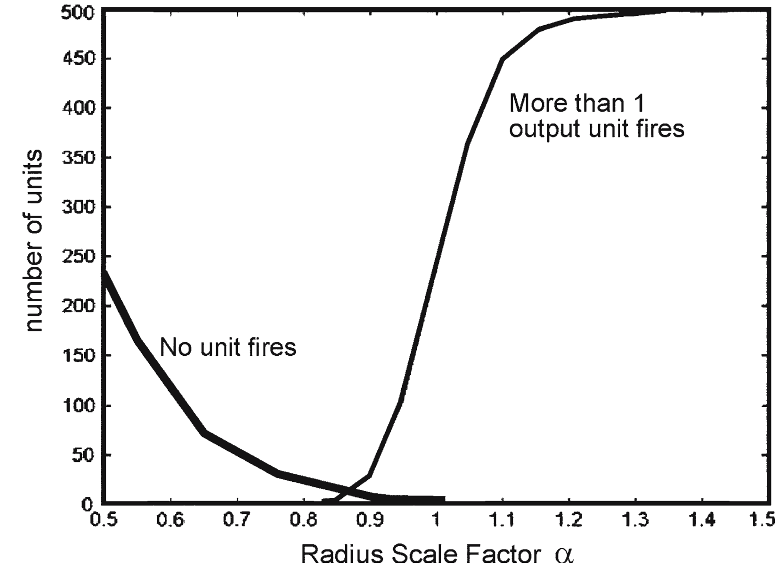

Figure 8. Number of times no unit fires (thick) and more than one unit fires (thin) as a function of α. The sum of these is the number of images rejected by the RCE network.

of rejections due to the fact that *more than one output unit was activated.* Figure 8 shows the result of this experiment on a database of $M = 500$ individuals. The sum of both of these quantities is the number of images rejected by the system.

5.5. PERFORMANCE ON DIFFERENT TYPES OF DATABASES

For all the above simulations, the memory set consisted of the four expression images: blank, smile, angry, and surprised. Since the test image shows a blank expression, it is conceivable that the results would be better by using the four blank training images instead of the four expression images. We performed three different experiments on the 500 person database: (1) store the four expression images, (2) store the four blank images, and (3) store all eight images per person. The results are shown in Table 3. Note that, as above, these results are averaged over five different runs.

Table 3. Performance of the RCE network when $M = 500$ individuals are stored, and different databases are used.

Memory set $M = 500$	Classification %		False acceptance %
	Reject	Error	
4 Expression images	3.5	0	1.2
4 Blank images	5.1	0	0.8
All 8 images	1.6	0	1.0

Surprisingly, the database of the four expressions gives lower rejection than the database of the four blank images. We can conclude that the blank test images sometimes contain things in common with the expression images; for example, one may have a relatively blank expression with just a hint of a smile. Clearly, the best result is to store eight images per person. However, the price to be paid for using eight samples per person rather than four samples is in terms of storage (double the size) and retrieval time (double the time).

5.6. ADDING AND DELETING ASSOCIATIONS

An important design consideration of associative memories is how easy it is to add and delete associations from the memory set. For the given RCE network, it is easy to delete individuals from the database—simply delete the hidden units allocated to that person (or else shrink the radius to 0).

In terms of adding individuals to the database, the fact that the number of false matches decreases as M increases (see Figure 6), allows us to use a modular approach for constructing and training large databases. For example, suppose we start with a $M=400$ person database. Now suppose we want to add an additional 400 people to the database. Rather than retraining on the whole 800 person system, we can just formulate and train a separate 400-person network to handle the new patterns. Then the outputs of the original network and the new network can be combined with a simple decision: If both networks reject the input image or both networks classify the input image, then the image is rejected. However, if one of the networks classifies the pattern and the other network rejects it, then we accept that classification. Figure 9 shows a schematic diagram of this modular design.

Note that we expect to use such a modular approach for large size databases ($M=1000$ and above) where the number of false matches are sufficiently low. We did, however, test the idea on the $M=400$ database. In this case, two 400-people RCE networks were independently trained and then the outputs combined with the simple decision rule described above. The results are shown in Table 4. The correct classification results show remarkable similar performance. The false acceptance performance is a bit worse for the two 400-person network, but not by much.

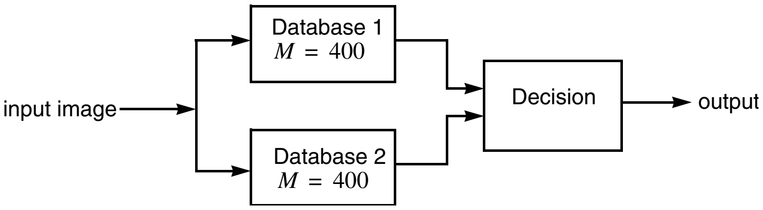

Figure 9. Block diagram of the modular construction.

Table 4. Performance of the modular RCE network (two 400-person databases) compared to the full network (a single 800 person database).

Memory set	Classification %		False Acceptance %
	Reject	Error	
800 person database	4.2	0.1	0.40
Two 400-person databases	4.6	0.4	0.65

6. Summary

This paper introduced an RCE-based model of associative memory. The capacity of the memory for the human face recognition problem was explored. Capacity here is defined in terms of the trade-off between correct classification performance and false acceptance performance. In the correct classification experiments, we want to minimize the number of misclassifications as well as the number of rejections. Simultaneously, though, we want to minimize the number of false matches for people who are not in the database. The results given in Figure 7 show that it is not possible to simultaneously optimize both capacity measures. In terms of design, one would specify the maximum amount of misclassification or false matches that the system should produce, and then one would determine an appropriate database size.

When using the RCE network on databases of up to 1000 people, the results indicate that the system can achieve a correct classification rate of over 99%, with a rejection rate of about 5%, and a false acceptance rate of less than 1%.

In future work, we will investigate the performance of this associative memory model when other types of features [4] and distance measures [9, 16] are used. In addition, we will explore the possibility of using a soft output (real number between 0 and 1) for each hidden unit instead of just 0 or 1.

Acknowledgements

The authors would like thank to the anonymous reviewers for their helpful comments and suggestions.

References

1. Artiklar, M., Hassoun, M. and Watta, P.: Application of a postprocessing algorithm for improved human face recognition, *Proceedings of the IEEE International Conference on Neural Networks*, IJCNN-1999, July 10–16, Washington, DC., Paper #JCNN 2166, 1999.
2. Belhumeur, P., Hespanha, J. and Kreigman, D.: Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* **19**(7) (1997), 711–720.
3. Duc, B. and Fischer, S.: Face authentication with Gabor information on deformable graphs, *IEEE Transactions on Image Processing*, **8**(4) (1999).

4. Duda, R., Hart, P. and Stork, D.: *Pattern Classification*, John Wiley & Sons, New York, Second edition, 2000.
5. Hasegawa, A., Shibata, K., Itoh, K., Ichioka, Y. and Inamura, K.: An adaptive neural network: application to character recognition on x-ray films, *Neural Networks*, **9**(1) (1996), 121–172.
6. Hassoun, M. H. (ed.): *Associative Neural Memories: Theory and Implementation*. Oxford University Press, New York, 1993.
7. Hassoun, M. H.: *Fundamentals of Artificial Neural Networks*, MIT Press, Cambridge, Mass, 1995.
8. Ikeda, N., Watta, P., Artiklar, M. and Hassoun, M.: Generalizations of the hamming net for high performance associate memory, *Neural Networks*, **14**(9) (2001), 1189–1200.
9. Mosorov, V.: A main stem concept for image matching, *Pattern Recognition Letters*, **26**(8) (2005), 1105–1117.
10. Mu, X.: Automated Face Recognition: A weighted voting method, *Ph.D. Dissertation*, Wayne State University, Dept. ECE, Detroit, MI, (2004) 48202.
11. Phillips, P., Moon, H., Rizvi, S. and Rauss, P.: The FERET evaluation methodology for face-recognition algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(10) (2000), 1090–1104.
12. Reilly, D. L., Cooper, L. N. and Erlbaum, C.: A neural model for category learning, *Biological Cybernetics*, **45** (1982), 35–41.
13. Turk, M. and Pentland, A.: Eigenfaces for recognition, *Journal of Cognitive Neuroscience*, **3**(1) (1991), 71–86.
14. Watta, P., Artiklar, M., Masadeh, A. and Hassoun M. H.: Construction and analysis of a database of face images which requires minimal preprocessing, *Proceedings of the IASTED Conference on Modeling and Simulation,* pp. 465–469, MS-2000, May 15–17, Pittsburgh Pennsylvania, (2000).
15. Watta, P. and Hassoun, M. H.: Generalizations of the hamming associative memory, *Neural Processing Letters*, **13**(2) (2001), 183–194.
16. Wilson, D. and Martinez, T.: Improved heterogeneous distance functions, *Journal of Artificial Intelligence Research*, **6** (1997), 1–34.
17. Wiskott, L., Fellous, J., Kruger, N. and von der Malsburg, C.: Face recognition by elastic bunch graph matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19** (1997), 775–779.
18. Wiskott, L.: The role of topographical constraints in the face recognition, *Pattern Recognition Letters*, **20** (1999), 89–96.