Vol. 40 No. 1 Transactions of Information Processing Society of Japan Jan. 1999

Technical Note

An Associative Memory System with Fuzzy Numbers

YUKINORI SUZUKI,† NORIO KONNO†† and JUNJI MAEDA†

We propose an associative memory system (AMS) with fuzzy numbers. The AMS uses a memory matrix and a defuzzification procedure. The memory matrix associates fuzzy response vectors which are transformed into non-fuzzy response vectors by the defuzzification procedure. The performance of the AMS was evaluated by computational experiments. The results of the experiments indicated that the memory capacity of the AMS was larger than that of the standard linear associative memories. In addition, the noise tolerance of the AMS was practically the same as that of the linear associative memory using the generalized inverse matrix method.

1. Introduction

An associative memory (AM) recalls contents of stored information from all or a portion of the stored information. In AM, the information is distributed over the whole memory system unlike the standard computer memory systems in which each piece of information is stored at a specific address [1),2)]. The distributive information storage of the AM makes the overall memory function highly resistant to malfunction due to faults in memory and/or noisy input. These characteristics of the AM resemble those of the human brain which is speculated to be working as an AM. AM should be the key to the correct and quick retrieval of necessary and related data from a large database [3),4)].

Many types of AMs have been proposed. The major difference among the AMs is in the memory construction procedure used to form a memory matrix. In a linear AM framework, the stimulus and response vectors are formed column by column into stimulus and response matrices $\mathbf{X}$ and $\mathbf{Y}$, respectively. From these matrices, a memory matrix $\mathbf{M}$ is generated by various memory construction procedures. Unlike the standard computer memories, the memory construction phase of the AM is a "write" operation, and the association phase corresponds to a "read" operation. In the recall phase of the AM, the memory matrix is multiplied by a stimulus vector $\mathbf{x}$ to produce the associative response vector $\mathbf{y}$ as

$$\mathbf{y}=\mathbf{M}\mathbf{x}. \tag{1}$$

The standard methods of constructing memory matrices are the correlation matrix (CM) method and the generalized inverse matrix (GIM) method. In the CM method, response vectors are retrieved correctly under the condition that stimulus vectors are orthogonal to each other. On the other hand, the GIM method performs correct response vector retrieval under the condition that the stimulus vectors are linearly independent [5)~7)]. These conditions are too difficult to be met to make AM applicable to real-world problems such as pattern recognition. Therefore, we have developed an associative memory system (AMS) using fuzzy numbers for pattern recognition. The AMS is not constrained by the above conditions, and in addition, the AMS is superior to the standard methods in the following respects.

For pattern recognition, the performance of the AM to recognize incoming patterns is determined not only by the ability of the AM, but also by the stimulus and response vectors to be stored in the AM. If the incoming pattern is similar to the stimulus vector stored in the AM, the AM is able to associate a response vector to recognize the pattern correctly. On the other hand, if there is no stimulus vector similar to the incoming pattern, the AM cannot associate a response vector to recognize the incoming pattern correctly. However, in real-world problems, available information is imprecise, uncertain, and possibly incorrect. Since we make stimulus and response vectors stored in the AM using such information, the stored vectors may be imprecise and incorrect. The AMS that we have developed can cope with this problem. We stored the stimulus and response vectors which contain uncertainty in the AMS using fuzzy numbers. The AMS allows us to process information flexibly to solve the above problem [8),9)].

In this paper, we propose an AMS which is a memory matrix and a defuzzification procedure. The elements of the matrix are fuzzy numbers and the matrix associates a fuzzy response vector. Then, the defuzzification procedure transforms the fuzzy response vector to a non-fuzzy response vector. In addition, we evaluate the performance of the AMS with regard to memory capacity and noise

† Department of Computer Science & Systems Engineering, Muroran Institute of Technology
†† Department of Applied Mathematics, Faculty of Engineering, Yokohama National University

tolerance by computational experiments.

This paper is organized as follows. In Section 2, we will show a design method of the memory matrix using fuzzy numbers when a set of stimulus and response vectors to be stored in the AMS are given. In Section 3, we will describe the defuzzification procedure. We evaluate performance of the AMS by computational experiments in Section 4. We will conclude this paper in Section 5.

2. Design of a Memory Matrix

q pairs of vectors $(\mathbf{x}_i, \mathbf{y}_i)\,(i = 1, 2, \ldots, q)$ are given to be stored in the AMS. $\mathbf{x}_i$ and $\mathbf{y}_i$ are stimulus and response vectors, respectively. Since the stimulus vector is different from the corresponding response vector, this type of the AM is a hetero-AM. In a linear AM framework, the relationship between a stimulus and a response vectors is described as

$$\mathbf{y}_i = \mathbf{M}\mathbf{x}_i \quad (i = 1, 2, \ldots, q) \tag{2}$$

where the dimensions of the vectors $\mathbf{x}_i$ and $\mathbf{y}_i$ are n and p, respectively. Therefore, the size of memory matrix $\mathbf{M}$ is (p, n). q stimulus vectors are formed column by column into a stimulus matrix $\mathbf{X}$. In the same manner, response vectors are formed column by column into response a matrix $\mathbf{Y}$. Usually, $\mathbf{Y}$ is an identity matrix in hetero-AMs. Therefore, we employ an identity matrix $\mathbf{Y}$. We represent the elements of $\mathbf{x}_i$ and $\mathbf{y}_i$ as $x_{ki}\,(k = 1, 2, \ldots, n)$ and $y_{ji}\,(j = 1, 2, \ldots, p)$, respectively. The elements of $\mathbf{M}$ are represented as m_{jk}. In the present AMS, m_{jk} is a fuzzy number.

A fuzzy number is a convex and a normal fuzzy set defined in the real line, and its membership function is piecewise smooth. The fuzzy number represents the imprecisely specified quantity "approximate m". Dubois and Prade introduced a L-R fuzzy number to improve arithmetic efficiency between fuzzy numbers[10]. We employed a L-L fuzzy number to design the memory matrix $\mathbf{M}$. It is a special case of the L-R fuzzy number in which the left side spread of the membership function is the same as the right side. A membership function of the L-L fuzzy number is given as

$$\mu(x) = \left\{ \begin{array}{ll} L\left(\dfrac{a-x}{c}\right) & x \le a, c > 0 \\ L\left(\dfrac{x-a}{c}\right) & x > a, c > 0 \end{array} \right\} \tag{3}$$

where a and c are the mean and the spread, respectively. L is the shape function of the membership function. For the memory matrix design, we use a triangle shape membership function as shown in **Fig. 1**, so that the element of the fuzzy matrix m_{jk} is determined by its mean a_{jk} and spread c_{jk}.

To define the equality between two fuzzy numbers, Dubois and Prade introduced a possibility measure which defines a grade of possibility of equality between two fuzzy numbers[10]. The following equation (4) expresses that the grade of possibility of the equality between two fuzzy numbers f_i and f_j is greater than α.

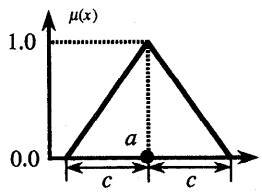

Fig. 1 A L-L fuzzy number. $\mu(x)$ is a membership function of the fuzzy number, and a and c are the mean and the spread of the membership function, respectively.

$$Pos(f_i = f_j) > \alpha \tag{4}$$

By the definition, Eq. (4) is equivalent to $f_i^{\alpha R} \ge f_j^{\alpha L}$ and $f_i^{\alpha L} \le f_j^{\alpha R}$, where $f_i^{\alpha L}$ and $f_i^{\alpha R}$ are left and right sides of an α-level set of the fuzzy number f_i, respectively. $f_j^{\alpha L}$ and $f_j^{\alpha R}$ are also left and right sides of α-level set of the fuzzy number f_j, respectively.

For Eq. (2) to hold, y_{ji} has to be equal to $m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}$ for all $(i = 1, 2, \ldots, q;\ j = 1, 2, \ldots, p)$. A set of coefficients $(m_{j1}, m_{j2}, \ldots, m_{jn})$ is determined to hold the equality between the real number y_{ji} and the fuzzy number $m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}$. This is a fuzzy regression analysis[11],[12]. Using a possibility measure, the grade of possibility of equality between y_{ji} and $m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}$ is greater than α is

$$Pos(y_{ji} = m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}) \ge \alpha. \tag{5}$$

An α-level set of $m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}$ satisfies the equations

$$L\Big(\big((a_{j1}x_{1i} + \cdots + a_{jn}) - M_j^{\alpha L}\big) \Big/ \big(c_{j1}|x_{1i}| + \cdots + c_{jn}|x_{ni}|\big)\Big) = \alpha \tag{6}$$

and

$$L\Big(\big(M_j^{\alpha R} - (a_{j1}x_{1i} + \cdots + a_{jn}\big) \Big/ \big(c_{j1}|x_{1i}| + \cdots + c_{jn}|x_{ni}|\big)\Big) = \alpha \tag{7}$$

where $M_j^{\alpha L}$ and $M_j^{\alpha R}$ are left and right sides of an α-level set of $m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}$, respectively. Since a real number is a special case of a L-L fuzzy number whose spreads are zero, Eq. (5) is equivalent to $y_{ij} \ge M_{ij}^{\alpha L}$ and $y_{ij} \le M_{ij}^{\alpha L}$. From Eqs. (6) and (7), $y_{ij} \ge M_{ij}^{\alpha L}$ and $y_{ij} \le M_{ij}^{\alpha L}$ become

$$-y_{ji} + a_{j1}x_{1i} + \cdots + a_{jn}x_{ni} \le L^{-1}(\alpha)\,\{c_{j1}|x_{1i}| + \cdots + c_{jn}|x_{ni}|\} \tag{8}$$

and

$$y_{ji} - a_{j1}x_{1i} - \cdots - a_{jn}x_{ni} \le L^{-1}(\alpha)\,\{c_{j1}|x_{1i}| + \cdots + c_{jn}|x_{ni}|\} \tag{9}$$

for all $(i = 1, 2, \ldots, q;\ j = 1, 2, \ldots, p;\ k = 1, 2, \ldots, n)$, respectively.

As shown in Eqs. (8) and (9), multiplying a L-L fuzzy number by a real number makes the spread of a L-L fuzzy number wider. Many multiplications by real numbers causes a very large spread. Since a very large spread of the fuzzy number is trivial, we compute the fuzzy numbers so as to minimize their spreads as much as possible under the constraints (8) and (9). To perform this, we set an objective function as

$$J(c)=\sum_{i=1}^{q}\sum_{j=1}^{p}\{c_{j1}|x_{1i}|+\cdots+c_{jn}|x_{ni}|\}. \quad (10)$$

Therefore, means and spreads of the elements of the memory matrix are computed by linear programming which minimizes the objective function (10) under the constraints (8) and (9).

3. Defuzzification

As illustrated in Section 2, the memory matrix **M** is computed by both the given stimulus matrix **X** and response matrix **Y**. For association, since **MX** operation produces a fuzzy matrix $\tilde{\mathbf{Y}}$ whose elements are L-L fuzzy numbers, defuzzification is necessary to transform $\tilde{\mathbf{Y}}$ into a crisp matrix **D**. **D** is an associated matrix with **X**. Elements of $\tilde{\mathbf{Y}}$ and **D** are represented by $\tilde{y}_{ji}$ and d_{ji}, respectively. If **D** is identical to **Y**, the AMS performs correct association.

The defuzzification is carried out based on the following idea. The memory matrix is computed by linear programming. To solve a linear programming problem, since a simplex method is the most popular, the defuzzification procedure is developed under the assumption that the simplex method is used to solve a linear programming problem. In the simplex method, constrains form the feasible region where the solutions exist. The feasible region is bounded by hyperplanes which form a geometrical simplex. The optimal solution of linear programming by the simplex method will be at the vertices of the simplex made by the hyperplanes. At those vertices, $Pos(y_{ji} = m_{j1}x_{1i} + \cdots + m_{jn}x_{ni}) \geq \alpha$ is close to α but greater than α. When the grade of membership function of $\tilde{y}_{ji}$ at one is closer to α than the grade of membership function of $\tilde{y}_{ji}$ at zero, the defuzzification procedure generates one as element d_{ji} of **D**. The converse is also true. The defuzzification procedure is therefore concluded as follows. 0_α and 1_α are the grade of membership function of y_{ji} at one and zero, respectively.

(1) If $0_\alpha \geq \alpha$ and $1_\alpha < \alpha$, then $\tilde{y}_{ji}$ is transformed into a zero.

(2) If $0_\alpha < \alpha$ and $1_\alpha \geq \alpha$, then $\tilde{y}_{ji}$ is transformed into a one.

(3) when $0_\alpha \geq \alpha$ and $1_\alpha \geq \alpha$,
if $|\alpha - 0_\alpha| < |\alpha - 1_\alpha|$, then $\tilde{y}_{ji}$ is transformed into a zero.
if $|\alpha - 0_\alpha| > |\alpha - 1_\alpha|$, then $\tilde{y}_{ji}$ is transformed into a one.
if $|\alpha - 0_\alpha| = |\alpha - 1_\alpha|$ and mean of $\tilde{y}_{ji}$ is less than or equal to 0.5, then $\tilde{y}_{ji}$ is transformed into a zero.
if $|\alpha - 0_\alpha| = |\alpha - 1_\alpha|$ and mean of $\tilde{y}_{ji}$ is greater than 0.5, then $\tilde{y}_{ji}$ is transformed into a one.

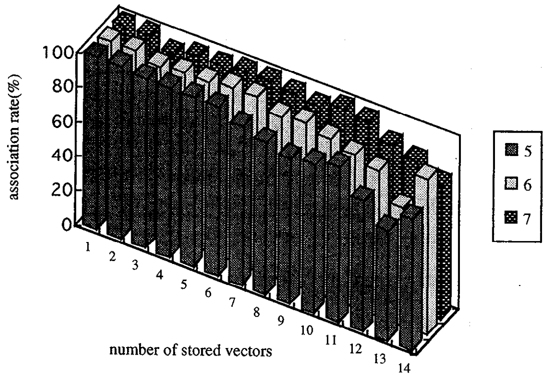

Fig. 2 The performance with regard to the memory capacity. In the experiments, the dimension of the stimulus vector is varied to be 5, 6, or 7. The number of the stored vectors in the AMS is varied between 1 to 14.

4. Performance of the AMS

The performance of the AMS with regard to both the memory capacity and noise tolerance is evaluated by computational experiments. In the experiments, stimulus matrix **X** is generated by random variables uniformly distributed in the interval [0, 1]. 50 **X**'s for a pair of matrix size (n, p) are made and 50 **M**'s corresponding to each **X** and **Y** are computed through the procedure described in Section 2. We computed a linear programming problem by two-phase revised simple method[13]. In the experiments, the dimensions of stimulus vector (n) are varied to be 5, 6, or 7. A number of stored vectors (q) are varied between 1 and 14, so that the experiments were carried out in the case that the number of stored vectors is larger than the dimensions of the stimulus vector. α is given as 0.8 in the experiments. We consider that the AMS carried out correct association when 90% of the elements of **D** is the same as the corresponding elements of **Y**. A correct association rate is evaluated by the number of correct association for 50 **Y**'s.

Figure 2 shows the results of the performance in terms of memory capacity. When the number of stored vectors is smaller than or equal to the dimension of the stimulus vector, the AMS indicated more than 95% correct association rate at ($n = 6$, $q = 3$) and 94% at ($n = 7$, $q = 3$). In the case of $n = 7$, the correct association rate of the AMS was more than 90% up to $q = 11$. More than 90% cor-

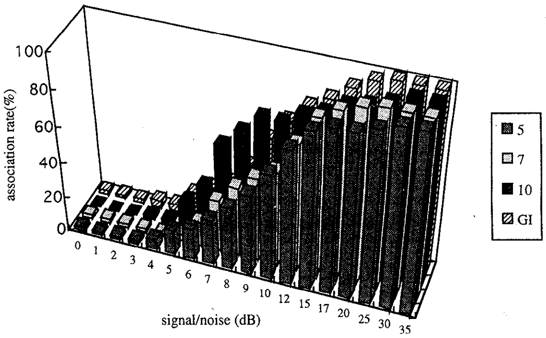

Fig. 3 The performance with regard to the noise tolerance. This is the result of the experiments in the case that $n = 7$. The number of stored vectors varied to be 5, 7, 10. The noise tolerance of the GIM method in the case of ($n = 7$, $q = 7$) is also presented for the purpose of comparison.

rect association rate was achieved up to $q = 10$ in the case of $n = 6$. Moreover, in the case of $n = 5$, more than 92% association rate was demonstrated up to $q = 7$. Therefore, the experiment showed that the AMS can store stimulus vectors up to the dimensions of stimulus vectors. However, the AMS is free from orthogonality and linear independence problems unlike the linear AMs.

The noise tolerance of the AMS is evaluated by making noise which is added to the stimulus vectors. The same **X**'s and **M**'s used in the experiments of the memory capacity were employed. The experiments were conducted by changing both a pair of (n, q) and the signal-to-noise ratio. The noise tolerance is evaluated for three cases: $q < n$, $q = n$, and $q > n$. We also compared the noise tolerance of the AMS with the noise tolerance of the AM using GIM method. **Figure 3** shows the result of the experiments in the case that $n = 7$. In this case, the number of the stored vectors (q) was varied to be 5, 7, and 10. The noise tolerance of the GIM method in the case of ($n = 7$, $q = 7$) is also presented for the purpose of comparison. The results show that the noise performance of the AMS is practically the same as that of the AM using the GIM method in all cases $q < n$, $q = n$, and $q > n$.

5. Conclusion

In this paper, we proposed an AMS with fuzzy numbers. Information which contains uncertainty is stored in the AMS using fuzzy number. The computational experiments found that the AMS is able to store stimulus vectors up to the dimensions of the stimulus vectors, and that the noise tolerance is practically the same as AM using the GIM method. The AMS is the first step to realize the AM which is adaptive to the real-world problems.

References

1) Baum, E.B.: Building an Associative Memory Vastly Larger than Brain, *Science*, Vol.268, pp.583–585 (1995).
2) Kohonen, T.: *Self-Organization and Associative Memory*, Springer-Verlag, Berlin (1987).
3) Tsunoda, T.: Present Status and Tendency of Associative Memory, *J. IPS Japan*, Vol.36, No.1, pp.89–96 (1995).
4) Nakano, K. (Ed.): *An Introduction to Neurocomputing*, Corona, Tokyo (1990).
5) Kohonen, T.: Correlation Matrix Memories, *IEEE Trans. Comput.*, Vol.C-21, pp.353–359 (1972).
6) Kohonen, T. and Ruohonen, M.: Representation of Associated Data by Matrix Operators, *IEEE Trans. Comput.*, Vol.C-22, pp.701–702 (1973).
7) Micheal, A.M. and Farrel, J.A.: Associative Memories via Artificial Neural Networks, *IEEE Control Systems Magazine*, April, pp.6–17 (1990).
8) Ogawa, H.: *New Development of Pattern Recognition*, Sanmi Publisher, Tokyo (1993).
9) Zadeh, L.A.: What is soft computing? *Soft Computing*, Vol.1/1, pp.1–2 (1997).
10) Sakakazu, M.: *Fundamentals and Practice on Fuzzy Theory*, Morikita Publisher, Tokyo (1989).
11) Konno, N. and Suzuki, Y.: A Formulation for Fuzzy Matrices by Fuzzy Linear Regression Analysis, *Trans. IEICE* (D-II), Vol.J74-A, No.3, pp.574–575 (1991).
12) Tanaka, H., Uejima, S. and Asai, K.: Linear Regression Analysis with Fuzzy Model, *IEEE Trans. Syst., Man, Cybern.*, Vol.SMC-12, No.6, pp.903–907 (1982).
13) Ibaragi, T. and Fukushima, M.: *Fortran 77 Optimized Programming*, Iwanami, Tokyo (1991).

(Received May 11, 1998)
(Accepted October 2, 1998)