in the field of neural networks have brought out certain good temporal models and methods to meet this demand. One major method is to incorporate feedback into static or mapping neural networks, making them recurrent. This leads to the so-called recurrent neural network [13]. In fact, the recurrent neural network contains many types of neural network, such as Hopfield network [2] as well as its generalized version [4], simplex memory neural network [5], continuous-time recurrent neural network [8] and discrete-time or real-time recurrent neural network [10,15]. Moreover, the recurrent neural network has been widely applied to temporal information processing, such as speech recognition, prediction, and system identification and control (e.g., [7,9–11,14]).

However, little progress has been made on theoretical study of the capacity of the recurrent neural network to learn and memorize spatio-temporal sequences. In literature, we can only find a theoretic result given by Amari [1] that the memory capacity of the so-called autoassociative memory model with the sum-of-outer product scheme on a simple spatio-temporal sequence, is about $0{:}27n$, where n is the number of the processing neurons. Actually, the autoassociative memory model is just a special form

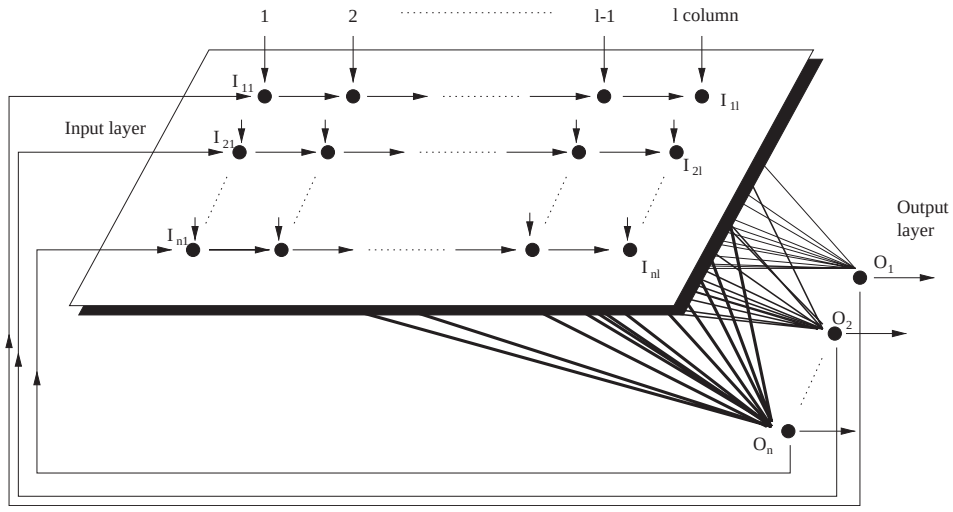
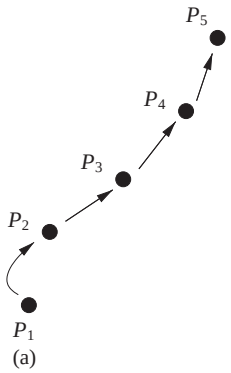


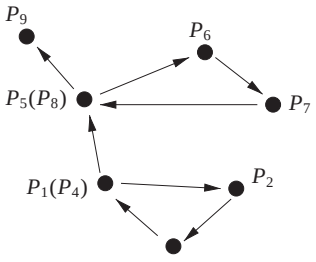
Fig. 1. The structure sketch of the time-delay recurrent neural network (TDRNN) with l-step delay feedback.

be fed back to $l_{i;1}$. All the input neurons are connected to each processing neuron in the output layer.

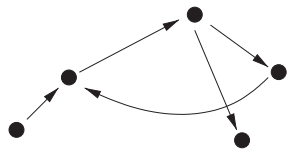
The TDRNN operates in the following way. It starts with a set of initial states (i.e., bipolar patterns) $X(0); X(1); \ldots; X(l-1)$. We assume that the state of the array of input neurons at the time t is $[X(t); X(t-1); \ldots; X(t-l+1)]$. Then, the output of the TDRNN at the next time, i.e., $X(t+1) = [x_1(t+1); xp$



(a)



(b)


(c)

Proof. By the function of a TDRNN, N can only produce a spatio-temporal sequence with its order equal to or less than l, from any set of initial patterns. Thus via any learning algorithm, N is only possible to memorize a spatio-temporal sequence whose order is equal to or less than l. Therefore, the condition $l \geqslant k$ is necessary. The proof is completed. □

Essentially, Theorem 1 gives a match law between a TDRNN and a spatio-temporal sequence. That is, if the order of a spatio-temporal sequence is equal to or less than the number of steps of time-delay feedback in a TDRNN, it is possible for the TDRNN to learn and memorize this spatio-temporal sequence. Otherwise, the TDRNN cannot learn and memorize this spatio-temporal sequence by any learning algorithm. It is also clear that the condition $l \geqslant k$ is not sufficient for memorizing any $\mathscr{S}$ by the TDRNN via a general learning algorithm. However, we will prove that this condition is really sufficient in certain case in the next section. For simplicity, we will only analyze the memory capacity of the TDRNN on learning and memorizing bipolar spatio-temporal sequences in the remainder of this paper. It is certain that the length of a non-periodic bipolar spatio-temporal sequence $\mathscr{S}$, i.e., m, is certainly finite if its order is finite. When spatio-temporal sequences become binary, the following analysis is still true if the states of the processing neurons become binary.

3. The higher order TDRNNs

We begin with the order of a TDRNN. For the storage of bipolar spatio-temporal sequences, we use perceptrons as the processing neurons in the network. Then, the input pattern for each of these perceptrons is just the state of the array of input neurons. Mathematically, a perceptron becomes of higher order if some higher order terms of the components of the input pattern are involved in the computation. It is natural to define the order of a TDRNN as the order of these perceptrons, i.e., the processing neurons of the TDRNN. For the sake of clarity, we define the higher order perceptron and its generalized perceptron learning algorithm as follows.

As well-known, a perceptron is a processing unit with a d-dimensional input pattern $X = [x_1; x_2; \ldots; x_d]^T \in R^d$ and a bipolar output $y \in \{-1; 1\}$ via a weight vector $W = [w_1; w_2; \ldots; w_d]^T$ and a threshold value θ such that for an input pattern X, the output y is computed by

$$y(X) = T(X \mid W; \theta) = \mathrm{Sgn}(H(X)) = \begin{cases} 1 & \text{if } H(X) \geqslant 0; \\ -1 & \text{otherwise;} \end{cases} \tag{3}$$

where

$$H(X) = \sum_{i=1}^{d} w_i x_i - \theta.$$

We consider bipolar input patterns and assume that A; B are two disjoint subsets of a sample set in $\{-1; 1\}^d$. Then, (A; B) becomes a learning object of the perceptron, that

is, the perceptron is expected to implement the following relationship:

$$T(X \mid W; \theta) = \begin{cases} 1 & \text{if } X \in A; \\ -1 & \text{if } X \in B: \end{cases} \tag{4}$$

Clearly, if A and B are linearly separable in the d-dimensional Euclidean space, the learning object $(A; B)$ can be realized by a perceptron via the perceptron learning algorithm [12] on the sample set $A \cup B$, i.e., W and θ are modified by the following learning rule:

$$\Delta W$$

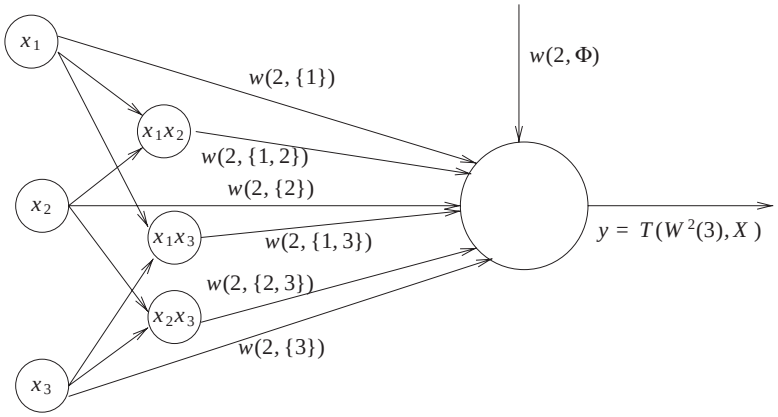


Fig. 3. The structure sketch of the second-order perceptron.

input variables $x_1; x_2; x_3$, respectively, while the other three weight vectors $w(3; \{1; 2\})$, $w(3; \{1; 3\})$, $w(3; \{2; 3\})$ are corresponding to the three second terms of the input variables, i.e., x_1x_2

(ii) $d \geqslant 2$, $b(X)$ $\{-1;1\}^d$
$(d-1)$th , , $(d-1)$th
$W^{d-1}(d)$, $X \in \{-1;1\}^d$

$$T(W^{d-1}(d);X) \neq b(X): \tag{7}$$

The proof is given in Appendix A.

We now turn to the higher order TDRNN. That is, the TDRNN consists of n higher order perceptrons as its processing neurons with the same input array in which the nl input variables are considered independently. If all these perceptrons are of the pth order ($1 \leqslant p \leqslant nl$), the TDRNN is defined to be of the pth order. In this situation, we can apply the perceptron learning algorithm to each of these pth order perceptrons to train its weights and threshold value for learning and memorizing a bipolar spatio-temporal sequence.

When the TDRNN is of the (nl)th order, that is, the order of each perceptron in the TDRNN becomes the largest one—the number of components of the input pattern, we call it the full order TDRNN. Actually, we have the following theorem on the full order TDRNN for learning and memorizing bipolar spatio-temporal sequences.

Theorem 3. l-
- k $k \leqslant l$.

Proof. By Theorem 1, it is possible for the full order TDRNN of l-step feedback to learn and memorize a bipolar spatio-temporal sequence of the order k when $k \leqslant l$. Moreover, the network is able to learn and memorize a bipolar spatio-temporal sequence $\mathscr{S}: P_1P_2\cdots P_m$ if and only if it is able to realize the recurrent function:

$$F(P_i;P_{i+1};\ldots;P_{i+l-1}) = P_{i+l}; \quad i = 1;2;\ldots;m-l: \tag{8}$$

In other words, each perceptron (i.e., processing neuron) $j(=1;2;\ldots;n)$ is able to learn and memorize the following learning object:

$$A_j(\mathscr{S}) = \{X^{nl}(i) : P_{i+l;j} = 1; \quad i = 1;2;\ldots;m-l\};$$

$$B_j(\mathscr{S}) = \{X^{nl}(i) : P_{i+l;j} = -1; \quad i = 1;2;\ldots;m-l\};$$

where

$$X^{nl}(i) = \mathrm{vec}[P_i;P_{i+1};\ldots;P_{i+l-1}] = [P_i^T;P_{i+1}^T;\ldots;P_{i+l-1}^T]^T:$$

According to Theorem 2, the (nl)th order perceptron is able to realize any learning object $(A_j(\mathscr{S});B_j(\mathscr{S}))$. Thus, each perceptron in the full order TDRNN can realize its learning object corresponding to the bipolar spatio-temporal sequence. Therefore, the full order TDRNN of l-step feedback is able to learn and memorize this bipolar spatio-temporal sequence. The proof is completed. □

By Theorem 3, we have found that the condition $l \geqslant k$ is really sufficient for learning and memorizing any bipolar spatio-temporal sequence by the full order TDRNN of l-step feedback. By Theorem 2(ii), we have also found that the condition of the full order is even necessary for the TDRNN to learn and memorize any bipolar

spatio-temporal sequence. However, it is hard to implement the full order TDRNN (or the higher order perceptron) when n (or p) is large. In fact, the first-order TDRNN of one-step feedback is already useful for many applications and its implementation is very easy. Therefore, it is still significant to study the first order TDRNN of one-step feedback. Obviously, this is Amari's autoassociative memory model. We will analyze the memory capacity of this model under the perceptron learning algorithm in the following section.

4. Memory capacity of the first-order TDRNN of one-step feedback

We first give two basic properties of the first-order TDRNN of one-step feedback on learning and memorizing a bipolar spatio-temporal sequence.

Theorem 4. $P_1, P_2, \ldots, P_{m-1}$, *fi* - - $P_1 P_2 \cdots P_m$.

Proof. According to the above definition, the first-order TDRNN of one-step feedback is uniquely defined by the weight matrix W and the threshold vector θ, where $w_{i,j}$ is the weight on the feedback line from the processing neuron j to i, θ_i is the threshold value of the processing neuron i. So, we need only to prove that there exists such a set of (W, θ) that the network can realize the function:

$$F(P_i) = P_{i+1}; \quad i = 1, 2, \ldots, m-1. \tag{9}$$

To solve for the weights and the threshold value of the ith processing neuron (a perceptron) in the network, a solution of Eq. (9) can be found from the following system of linear equations:

$$\begin{cases} p_{1,1} w_{i,1} + p_{1,2} w_{i,2} + \cdots + p_{1,n} w_{i,n} + \theta_i = p_{2,i}; \\ p_{2,1} w_{i,1} + p_{2,2} w_{i,2} + \cdots + p_{2,n} w_{i,n} + \theta_i = p_{3,i}; \\ \cdots \\ p_{m-1,1} w_{i,1} + p_{m-1,2} w_{i,2} + \cdots + p_{m-1,n} w_{i,n} + \theta_i = p_{m,i}; \end{cases} \tag{10}$$

where $w_{i,1}, w_{i,2}, \ldots, w_{i,n}$ and θ_i are the unknown numbers.

Because $P_1, P_2, \ldots, P_{m-1}$ are linearly independent, the rank of the system matrix of linear equations given by Eq. (10) is just $m-1$. Thus, the system of linear equations has solutions for $w_{i,1}, w_{i,2}, \ldots, w_{i,n}, \theta_i$. In this way for all the processing neurons, we certainly have a set of (W, θ) that enable the network to realize the function $F(P_i) = P_{i+1}$; $i = 1, 2, \ldots, m-1$. Therefore, the network is able to learn and memorize this bipolar spatio-temporal sequence. The proof is completed. □

By Theorem 4, we have a sufficient condition on bipolar spatio-temporal sequences for them to be learned and memorized by the first-order TDRNN of one-step feedback.

Theorem 5. *fi* - - ,
(i) $n \geqslant 2$ $m \leqslant 4$, -
.
(ii) $n \geqslant 3$ $m = 5$, -
.

Proof. If $n \geqslant 2$, for any simple bipolar spatio-temporal sequence $P_1P_2P_3P_4$, where $P_i \in \{-1; 1\}^n$, the rank of the matrix

$$A(P_1; P_2; P_3) = \begin{bmatrix} p_{1;1} & p_{1;2} & \cdots & p_{1;n} & 1 \\ p_{2;1} & p_{2;2} & \cdots & p_{2;n} & 1 \\ p_{3;1} & p_{3;2} & \cdots & p_{3;n} & 1 \end{bmatrix} \tag{11}$$

is certainly 3. Then, in a similar way to the proof of Theorem 4 we can prove that there exists a set of $(W; \square)$ which enable the network to realize $P_1P_2P_3P_4$. So it can be learned and memorized by the network via the perceptron learning algorithm. If m ¡ 4, all simple bipolar spatio-temporal sequences are obviously able to be stored. Thus (i) holds. As to (ii), since $n \geqslant 3$, we can easily construct a simple bipolar spatio-temporal sequence $P_1P_2 \cdots P_5$ with the following constraints:

$$P_2 = -P_1; \quad P_4 = -P_3;$$

$$p_{2;1} = 1; p_{3;1} = 1; \quad p_{4;1} = -1; p_{5;1} = -1:$$

If this spatio-temporal sequence can be learned and memorized, the first processing neuron is able to solve the XOR problem. This is contradictory to the fact that a (first-order) perceptron is unable to solve the XOR problem of the corresponding dimension. Therefore, this bipolar spatio-temporal sequence is unable to be learned and memorized by the network. The proof is completed. □

By Theorem 5, we have found that the memory capacity of the network is trivial if all possible simple bipolar spatio-temporal sequences of the same length are considered to be stored by the network. This result is just like that of Hopfield network to store a group of bipolar patterns as stable states. However, we can analyze the asymptotic memory capacity of the network which is significant when the size of the network becomes large.

We now consider that each P_i is an independent random vector and each component of P_i is an independent random variable which takes the equal probability distribution on $\{-1; 1\}$. In order to give the definition of the asymptotic memory capacity, we introduce the following probability sequence:

$$P(m; n) = P(\{\mathscr{S} = P_1P_2 \cdots P_m : \mathscr{S} \text{ is storable}\});$$

where "$\mathscr{S}$ is storable" which means that $\mathscr{S}$ can be learned and memorized by the first-order TNRNN with one-step feedback via the perceptron learning algorithm. It is easy to verify that $P(m;n)$ decreases with m.

De□nition 2. An integer function $C(n)$ is the asymptotic memory capacity of the first-order TDRNN of one-step feedback if the following two conditions hold:

(i)

$$\lim_{n\to\infty} P(m;n) = 1 \tag{12}$$

whenever $m \leqslant C(n)$;

(ii)

$$\lim_{n\to\infty} \inf P(m;n) < 1; \tag{13}$$

whenever $m > C(n)$.

We say that $C(n)$ is a lower bound of the asymptotic memory capacity if it satisfies the first condition; and that $C(n)$ is an upper bound of the asymptotic memory capacity if Eq. (13) holds whenever $m \geqslant C(n)$.

Since $P(m;n)$ decreases with m, we can easily prove that there exists a unique asymptotic memory capacity (function) of this kind of TDRNN. We further have

Theorem 6. $C(n)$ *fi* - - . $C(n) \geqslant C_1(n) = n + 1$.

Proof. We let $\mathscr{B}_m$ be the set of all bipolar spatio-temporal sequences of the length m which are storable. Then we have

$$P(n+1;n) = P(\mathscr{B}_{n+1}) = \frac{|\mathscr{B}_{n+1}|}{2^{n(n+1)}}. \tag{14}$$

We define

$$E_{n\times n} = \begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ e_{n1} & e_{n2} & \cdots & e_{nn} \end{bmatrix};$$

where $e_{ij} \in \{-1;1\}$ for $1 \leqslant i;j \leqslant n$, and let

$$\square_n = \{E_{n\times n} : \mathrm{rank}(E_{n\times n}) = n\};$$

and denote $\square_n^*$ to be the complement set of $\square_n$. We further define

$$E_n = |\square_n|; \qquad E_n^* = |\square_n^*|:$$

By Komlos theorem [3] that

$$\lim_{n\to\infty} (E_n^* \div 2^{n^2}) = 0;$$

and according to Theorem 4, we have

$$\lim_{n\to\infty} P(n+1; n) = \lim_{n\to\infty} \frac{|\mathscr{B}_{n+1}|}{2^{n(n+1)}} \geqslant \lim_{n\to\infty} \frac{E_n 2^n}{2^{n(n+1)}}$$

$$= \lim_{n\to\infty} \frac{E_n}{2^{n^2}} = 1 - \lim_{n\to\infty} \frac{E_n^*}{2^{n^2}} = 1:$$

Because $P(m; n)$ decreases with m, we have

$$\lim_{n\to\infty} P(m; n) = 1; \quad \text{for } m \leqslant n:$$

Therefore, $C_1(n) = n + 1$ is a lower bound of the asymptotic memory capacity of the first-order TDRNN of one-step feedback. The proof is completed. □

By Theorem 6, we have found that the asymptotic memory capacity of the first-order TDRNN of one-step feedback is no less than $n+1$. It is much better than [illegible]

the others by

$$d_i^* = \min_{j=i} d_H(Q_i, Q_j) = \min\{d_H(Q_i, Q_j) : j = 1, \ldots, i-1, i+1, \ldots, 10\}.$$

As is well-known in information theory, $d_1^*, d_2^*, \ldots, d_{10}^*$ essentially give the bounds of radii of attraction of these Arabic numbers in the 49-dimensional bipolar space. In fact, the reasonable radius of attraction of each Q_i should be no more than $t_i^* = [(d_i^* - 1)/2]$, where $[x]$ denotes the integer part of a real number x. For an associative memory model, only when the radius of attraction of each Q_i is just t_i^*, the error probability of the retrieval of the stored patterns or sequences reaches the minimum in a general noisy environment.

Based on the Hamming distances between these sample patterns, we have

$$(t_1^*, t_2^*, t_3^*, t_4^*, t_5^*, t_6^*, t_7^*, t_8^*, t_9^*, t_{10}^*) = (3, 6, 5, 4, 9, 4, 5, 8, 3, 4),$$

which will be used to design the object values of the radius of attraction of the Arabic numbers in the OPLA algorithm [6] for learning simple spatio-temporal sequences.

We first consider the experiments of the TDRNN for learning and storing complex numeral sequences by the perceptron learning algorithm. In each experiment, we generated a higher order numeral sequence and then used a proper TDRNN to learn it by the perceptron learning algorithm on each processing neuron independently. Typically, we selected three second-order numeral sequences as follows:

$$\mathcal{S}_1 = 036271741635428659045269584729611213738820539140224679831075664332 5;$$

$$\mathcal{S}_2 = 198836153425241670231103286491845879260435127689057382209374654714856 21;$$

$$\mathcal{S}_3 = 402582913962433681703121593427860832847269014163792206457485051987304495;$$

where the lengths of $\mathcal{S}_1, \mathcal{S}_2$ and $\mathcal{S}_3$ are 67, 71 and 72, respectively.

We began to use the first-order TDRNN of two-step feedback to learn these three numeral sequences by the perceptron learning algorithm and found that only a small part of each numeral sequence can be learned and stored in the TDRNN. For example, the longest subsequence of $\mathcal{S}_1$ that can be learned and stored by the TDRNN is 036271741635428659045, with the length 21. But when the second-order TDRNN is used, $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ became storable. That is, when we applied the generalized (second-order) perceptron learning algorithm on each processing neuron, they had been learned and stored in a second-order TDRNN of two-step feedback. However, as we successively added some numerals to each of them and maintained the second-order, it has been shown by the experiments that there exist certain expanded numeral sequences which cannot be learned and stored in a second-order TDRNN of two-step feedback. That is, they can only be learned and stored in a more higher order TDRNN of two-step feedback.

By these and the other experiments, we have found that the TDRNN can learn and store the complex numeral sequences efficiently by the perceptron learning algorithm.

Moreover, as the order of the TDRNN becomes higher, the memory capacity, i.e., the length of storable numeral sequence, increases considerably.

We then turn to the case of simple numeral sequences. The TDRNN of one-step feedback was applied to learn and store a simple numeral sequence. Since there are only ten Arabic numerals, the length of a simple numeral sequence is very limited. In this specific situation, it is easy to learn and store a simple numeral sequence by the perceptron learning algorithm. Actually, by the experiments we have found that every simple numeral sequence can be quickly learned and stored in a first-order TDRNN of one-step feedback by the perceptron learning algorithm.

In order to store a simple numeral sequence with better behavior of associative memory, we further applied the object perceptron learning algorithm (OPLA) [6] to train the TDRNN. For a simple numeral sequence $\mathscr{S} = P_1 P_2 \cdots P_m$, if it can be learned and stored in a TDRNN of one-step feedback by the perceptron learning algorithm, we only have $P_{i+1} = F(P_i)$, where $F(\cdot)$ is the function of the TDRNN. That is, P_{i+1} can be retrieved from P_i. However, in a noisy environment there may appear some errors on certain bits of P_i, i.e., it may become $\hat{P}_i$ with $\hat{P}_i \neq P_i$, but the TDRNN is still required to retrieve P_{i+1} from the noisy pattern $\hat{P}_i$. In this way, the sequence can be retrieved normally in a noisy environment. From the experiments with the perceptron learning algorithm, we have found that there generally exists a basin of attraction of P_i such that P_{i+1} can be retrieved from any numeral within it. However, the radius of attraction of P_i may be very small or trivial. To get a reasonable radius of attraction for each P_i $(i = 1; 2; \cdots; m-1)$, we applied the OPLA to train the TDRNN on each processing neuron independently, with $t_i \leqslant t_i^*$ set as the object radius of attraction of P_i.

We selected two simple numeral sequences 259471680 and 0314628, with their lengths being 9 and 7, respectively. It was found by the experiments that they can be learned and stored in a first-order TDRNN of one-step feedback by the OPLA with $t_i \geqslant t_i^* - 2$. That is, each P_i except the last one in any of these two simple numeral sequences obtains the largest and reasonable radius of attraction. However, when we expanded them by adding the other numerals, it was shown by the experiments that the OPLA cannot make their expansions storable with $t_i \geqslant t_i^* - 2$. By these and the other experiments we have found that the memory capacity of the TDRNN of one-step feedback under the OPLA reduces to a certain degree. The reason is simply that the requirement of the reasonable radius of attraction for each numeral makes it difficult to learn and store a simple numeral sequence in a TDRNN of one-step feedback.

It has been further found by the experiments that the OPLA is better than the sum-of-outer product scheme on learning and storing a simple numeral sequence in the first order TDRNN of one-step feedback. First, the OPLA can learn and store a longer simple numeral sequence than the sum-of-outer product scheme does. Actually, the sum-of-outer product scheme can only learn and store the shorter simple numeral sequences such as 274630 and 87160. It cannot learn and store the above two sequences learned and stored by the OPLA. Second, when a simple numeral sequence is stored

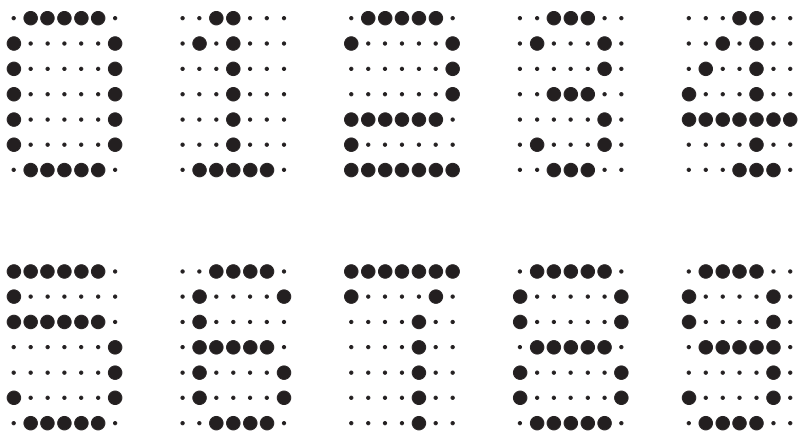


Fig. 4. The sample patterns of 10 Arabic numerals $\{0; 1; 2; 3; 4; 5; 6; 7; 8; 9\}$.

by the sum-of-outer product scheme, the radiuses of attraction of some numerals may be very small, but those of the others may be very large, which is not reasonable for associative memory. However, the OPLA can maintain reasonable radii of attraction of the numerals by setting a set of proper object values in advance.

6. Conclusion

We have investigated the capacity of time-delay recurrent neural network (TDRNN) for learning and memorizing spatio-temporal sequences. By introducing the order of a spatio-temporal sequence, we have established the match law between a TDRNN and a spatio-temporal sequence. It has been further proved that the full order TDRNN of l-step feedback is able to learn and memorize any bipolar (or binary) spatio-temporal sequence of the order k when $k \leqslant l$. Furthermore, we have obtained a lower bound of the asymptotic memory capacity of the first-order TDRNN of one-step feedback showing that such kind of TDRNN of n processing neurons can learn and memorize almost all the bipolar (or binary) random spatio-temporal sequences of the fixed length which is no more than n when n is large. Finally, the TDRNN has been demonstrated by the simulation experiments on learning and storing both the simple and complex spatio-temporal sequences of the Arabic numerals by the perceptron learning algorithm (Fig 4).

Acknowledgements

The author wishes to thank Prof. Shiyi Shen who first gave the proof of Theorem 2(i) and made some good suggestions on the paper. The author also thanks Mr. Yang Zhao for his support of the experiments.

Appendix A. Proof of Theorem 2

(i) This is certainly equivalent to the proposition that any learning object $(A^d; \overline{A^d})$ is linearly separable under $W^d(d)$, where A^d is an arbitrary subset of $\{-1;1\}^d$ and $\overline{A^d}$ is the complement set of A^d. We now prove it by induction. For the sake of clarity, we use the notation $X^d = [x_1; x_2; \ldots; x_d]^T \in \{-1;1\}^d$ in Appendix A.

When $d=1$, $\mathscr{F}_1^1 = \{\square; \{1\}\}$, $N_1^1 = 2$ and $W^1(1) = \{W(1;\square); W(1;\{1\})\}$. In this case, all the possible Boolean functions from $\{-1;1\}$ to $\{-1;1\}$ are

$$b(X) = 1; \quad b(X) = -1; \quad b(X) = X; \quad b(X) = -X; \tag{A.1}$$

corresponding to the learning objects $(\{-1;1\};\square); (\square;\{-1;1\}); (\{1\};\{-1\}); (\{-1\};\{1\})$ respectively. Certainly, these learning objects are linearly separable. In fact, $T(W^1(1); X)$ can realize these learning objects (Boolean functions) by the weight vectors $(1;0)$; $(-1;1); (0;1); (0;-1)$, respectively. Therefore it holds when $d=1$.

We assume that any learning object $(A^d; \overline{A^d})$ is linearly separable under $W^d(d)$. We now need only to prove that any learning object $(A^{d+1}; \overline{A^{d+1}})$ is linearly separable under $W^{d+1}(d+1)$ with the above inductive assumption. In order to do so, we introduce the following notations:

$$A^{d+1}(x_{d+1}) = \{X^d = [x_1; x_2; \ldots; x_d]^T : [(X^d)^T; x_{d+1}]^T \in A^{d+1}\}; \tag{A.2}$$

where $x_{d+1} = \pm 1$. Since $A^{d+1}(a) \subset \{-1;1\}^d (a = \pm 1)$, by the induction assumption, the learning object $(A^{d+1}(a); \overline{A^{d+1}(a)})$ is linearly separable. Then there exists a dth order spread weight vector $W^d(a;d) = \{w(a;d;\square) : \square \in \mathscr{F}_d^d\}$ by which we have

$$\sum_{\square \in \mathscr{F}_d^d} w(a;d;\square) U(X^d;\square) \begin{cases} \geqslant 0 & \text{if } X^d \in A^{d+1}(a); \\ < 0 & \text{if } X^d \notin A^{d+1}(a): \end{cases} \tag{A.3}$$

Now we define the $(d+1)$th order spread weight vector $W^{d+1}(d+1)$ as follows:

(1) For $\square \in \mathscr{F}_d^d \subset \mathscr{F}_{d+1}^{d+1}$,

$$w(d+1;\square) = w(1;d;\square) + w(-1;d;\square): \tag{A.4}$$

(2) For the set $\{\square; d+1\} = \square \cup \{d+1\} \in \mathscr{F}_{d+1}^{d+1}$,

$$w(d+1;\{\square; d+1\}) = w(1;d;\square) - w(-1;d;\square): \tag{A.5}$$

By $W^{d+1}(d+1)$ so defined, we have for each $X^{d+1} \in \{-1;1\}^{d+1}$

$$\begin{aligned} &\sum_{\square \in \mathscr{F}_{d+1}^{d+1}} w(d+1;\square) U(X^{d+1};\square) \\ &= \sum_{\square \in \mathscr{F}_d^d} w(d+1;\square) U(X^d;\square) + \sum_{\square \in \mathscr{F}_d^d} w(d+1;\{\square; d+1\}) U(X^d;\square) x_{d+1} \end{aligned}$$

$$= \sum_{\sigma \in \mathcal{F}_d^d} [w(1;d;\sigma) + w(-1;d;\sigma) + (w(1;d;\sigma) - w(-1;d;\sigma))x_{d+1}]U(X^d;\sigma)$$

$$= \begin{cases} 2 \sum_{\sigma \in \mathcal{F}_d^d} w(1;d;\sigma)U(X^d;\sigma) & \text{if } x_{d+1} = 1; \\ 2 \sum_{\sigma \in \mathcal{F}_d^d} w(-1;d;\sigma)U(X^d;\sigma) & \text{if } x_{d+1} = -1: \end{cases}$$

Thus, we have

$$\sum_{\sigma \in \mathcal{F}_{d+1}^{d+1}} w(d+1;\sigma)U(X^{d+1};\sigma) \begin{cases} \geqslant 0 & \text{if } X^{d+1} \in A^{d+1}; \\ < 0 & \text{if } X^{d+1} \notin A^{d+1}: \end{cases} \tag{A.6}$$

Therefore any learning object $(A^{d+1}; \overline{A^{d+1}})$ is linearly separable under $W^{d+1}(d+1)$. The proof of (i) is completed.

(ii) This is equivalent to the proposition that there exists a learning object $(A^d; \overline{A^d})$ which is not linearly separable under $W^{d-1}(d)$. We again prove it by induction and begin with $d = 2$. In this case, $W^1(2) = \{\sigma; \{1\}; \{2\}\}$. Then the perceptron is of the first order. We let $A_0^2 = \{(1;1); (-1;-1)\}$. Then the learning object $(A_0^2; \overline{A_0^2})$ is just the XOR problem and it is not linearly separable. Thus it holds when $d = 2$.

We assume that there exists a learning object $(A_0^d; \overline{A_0^d})$ which is not linearly separable under $W^{d-1}(d)$. Then we need only to prove there also exists a learning object $(A_0^{d+1}; \overline{A_0^{d+1}})$ which is not linearly separable under $W^d(d+1)$. We now prove it by contradiction. We assume that it does not hold in the case of $d+1$, that is, for any $A^{d+1} \subset \{-1;1\}^d$, $(A^{d+1}; \overline{A^{d+1}})$ is linearly separable under $W^d(d+1)$. We now let

$$A_0^{d+1} = \{[(X^d)^T; 1]^T : X^d \in A_0^d\} \cup \{[(X^d)^T; -1]^T : X^d \in \overline{A_0^d}\};$$

$$B_0^{d+1} = \{[(X^d)^T; -1]^T : X^d \in A_0^d\} \cup \{[(X^d)^T; 1]^T : X^d \in \overline{A_0^d}\}$$

$$= \overline{A_0^{d+1}}$$

Because $(A_0^{d+1}; \overline{A_0^{d+1}}) = (A_0^{d+1}; B_0^{d+1})$ is linearly separable under $W^d(d+1)$, there exists a dth order spread weight vector $W^d(d+1)$ by which we have

$$\sum_{\sigma \in \mathcal{F}_{d+1}^d} w(d+1;\sigma)U(X^{d+1};\sigma) \begin{cases} \geqslant 0 & \text{if } X^{d+1} \in A_0^{d+1}; \\ < 0 & \text{if } X^{d+1} \in B_0^{d+1}: \end{cases} \tag{A.7}$$

In a similar way as (i), we further have

$$\sum_{\sigma \in \mathcal{F}_{d+1}^d} w(d+1;\sigma)U(X^{d+1};\sigma)$$

$$= \sum_{\sigma \in \mathcal{F}_d^d} w(d+1;\sigma)U(X^d;\sigma) + \sum_{\sigma \in \mathcal{F}_d^{d-1}} w(d+1;\sigma \cup \{d+1\})U(X^d;\sigma)x_{d+1}:$$

When $X^d \in A_0^d$, since $[(X^d)^T; 1]^T \in A_0^{d+1}$ and $[(X^d)^T; -1]^T \in B_0^{d+1}$, we have the following two inequalities:

$$\sum_{\Omega \in \mathcal{F}_d^d} w(d+1; \Omega) U(X^d; \Omega) + \sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \geqslant 0; \tag{A.8}$$

$$\sum_{\Omega \in \mathcal{F}_d^d} w(d+1; \Omega) U(X^d; \Omega) - \sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \leqslant 0. \tag{A.9}$$

Subtracting Eq. (A.9) from Eq. (A.8), we have

$$2 \sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \geqslant 0. \tag{A.10}$$

Therefore, when $X^d \in A_0^d$, we have

$$\sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \geqslant 0. \tag{A.11}$$

On the other hand, when $X^d \notin A_0^d$ or $X^d \in \overline{A_0^d}$, since $[(X^d)^T; -1]^T \in A_0^{d+1}$ and $[(X^d)^T; 1]^T \in B_0^{d+1}$, we again have the following two inequalities:

$$\sum_{\Omega \in \mathcal{F}_d^d} w(d+1; \Omega) U(X^d; \Omega) + \sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \leqslant 0; \tag{A.12}$$

$$\sum_{\Omega \in \mathcal{F}_d^d} w(d+1; \Omega) U(X^d; \Omega) - \sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \geqslant 0. \tag{A.13}$$

Subtracting Eq. (A.13) from Eq. (A.12), we have

$$2 \sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \leqslant 0. \tag{A.14}$$

Therefore, when $X^d \notin A_0^d$, we have

$$\sum_{\Omega \in \mathcal{F}_d^{d-1}} w(d+1; \Omega \cup \{d+1\}) U(X^d; \Omega) x_{d+1} \geq 0. \tag{A.15}$$

Then we have $(A_0^d; \overline{A_0^d})$ is linearly separable under $W^{d-1}(d)$. This is contradictory to our inductive assumption. Thus it holds in the case of $d+1$ and the proof is completed. □

References

[1] S. Amari, Associative memory and its statistical-neurodynamical analysis, Springer Ser. Synergetics 42 (1988) 85–99.

[2] J.J. Hopfield, Neural networks and physical systems with collective computational abilities, Proc. Natl. Acad. Sci. USA 79 (1982) 2554–2558.

[3] J. Komlos, On the determinant of (0,1) matrices, Stud. Sci. Math. Hung. 2 (1967) 7–21.

[4] J. Ma, The stability of the generalized Hopfield networks in randomly asynchronous mode, Neural Networks 10 (1997) 1109–1116.

[5] J. Ma, Simplex memory neural network, Neural Networks 10 (1997) 25–29.

[6] J. Ma, The object perceptron learning algorithm on generalised Hopfield networks, Neural Comput. Appl. 8 (1999) 25–32.

[7] K. Narendra, K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Networks 1 (1990) 4–27.

[8] F.J. Pineda, Dynamics and architecture for neural computation, J. Complexity 4 (1988) 216–245.

[9] G. Puskorius, L. Feldman, Neurocontrol of dynamical systems with Kalman filter trained recurrent networks, IEEE Trans. Neural Networks 5 (1994) 279–297.

[10] A.J. Robinson, F. Fallside, Static and dynamic error propagation networks with application to speech coding, in: D.Z. Anderson (Ed.), Neural Information Processing Systems, American Institute of Physics, New York, 1988, pp. 632–641.

[11] A.J. Robinson, F. Fallside, A recurrent error propagation speech recognition system, Comput. Speech Lang. 5 (1991) 259–274.

[12] F. Rosenblatt, Principles of Neurodynamics, Spartan Books, New York, 1962.

[13] R.E. Rumelhart, G.E. Hinton, R.J. Williams, Parallel Distributed Processing (PDP): Exploration in the Microstructure of Cognition, MIT Press, Cambridge, MA, 1986.

[14] E.A. Wan, Times series prediction by using a connectionist network with internal delay lines, in: A.S. Weigend, N.A. Gershenfeld (Eds.), Time Series Prediction, Forecasting the Future and Understanding the Past, Addison-Wesley, Reading, MA, 1994, pp. 195–217.

[15] R.J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural network, Neural Comput. 1 (1989) 270–280.

Jinwen Ma received the Master of Science degree in applied mathematics from Xi'an Jiaotong University in 1988 and the Ph.D. degree in probability theory and statistics from Nankai University in 1992. From July 1992 to November 1999, he was a Lecturer or Associate professor at Department of Mathematics, Shantou University. From December 1999, he worked as a full professor at Institute of Mathematic, Shantou University. In September 2001, he was transferred to the Department of Information Science at the School of Mathematical Sciences, Peking University. During 1995 and 2003, he also visited and studied several times at Department of Computer Science and Engineering, the Chinese University of Hong Kong as a Research Associate or Fellow. He has published more than 40 academic papers on neural networks, pattern recognition, artificial intelligence, and information theory.