N-dimensional vector neuron

Tohru Nitta
Neuroscience Research Institute
National Institute of Advanced Industrial Science and Technology (AIST)
tohru-nitta@aist.go.jp

Abstract

We describe a new neuron model, N-dimensional vector neuron, which can deal with N signals as one cluster, by extending the 3-dimensional vector neuron to N dimensions naturally. The N-bit parity problem which cannot be solved with a single usual real-valued neuron, can be solved with a single N-dimensional vector neuron with the orthogonal decision boundary, which reveals the potent computational power of N-dimensional vector neurons. Rumelhart, Hinton and Williams showed that increasing the number of layers made the computational power of neural networks high [Rumelhart *et al.*, 1986a; Rumelhart *et al.*, 1986b]. Here we show that extending the dimensionality of neural networks to N dimensions originates the similar effect on neural networks.

1 Introduction

In order to provide a high computational power, there have been many attempts to design neural networks, taking account of task domains. For example, complex-valued neural networks have been researching since the 1970s [Aizenberg *et al.*, 1971; Widrow *et al.*, 1975]. Complex-valued neural networks whose parameters (weights and threshold values) are all complex numbers, are suitable for the fields dealing with complex numbers such as telecommunications, speech recognition and image processing with the Fourier transformation. Actually, we can find some applications of the complex-valued neural networks to various fields such as telecommunications and image processing in the literature [Hirose *et al.*, 2002; Nitta, 2001; Kuroe *et al.*, 2002]. For example, the fading equalization problem has been successfully solved with a single complex-valued neuron with the highest generalization ability [Nitta, 2003], using the property that the decision boundary for the real part of an output of a single complex-valued neuron and that for the imaginary part intersect orthogonally [Nitta, 2000]. The exclusive-or (XOR) problem and the detection of symmetry problem which cannot be solved with a single real-valued neuron [Minsky and Papert, 1969], can be solved with a single complex-valued neuron with the orthogonal decision boundaries [Nitta, 2003].

A three-dimensional vector neuron is a natural extension of the complex-valued neuron to three dimensions [Nitta and Garis, 1992], which can deal with three signals as one cluster: the input signals, thresholds and output signals are all 3D real-valued vectors, and the weights are all 3D orthogonal matrices. The activity $\boldsymbol{a}$ of neuron is defined to be :

$$\boldsymbol{a} = \sum_k \boldsymbol{V}_k \boldsymbol{s}_k + \boldsymbol{t}, \tag{1}$$

where $\boldsymbol{s}_k$ is the k-th 3D real-valued vector input signal, $\boldsymbol{V}_k$ is the 3D orthogonal weight matrix for the k-th input signal $\boldsymbol{s}_k$ (that is, an element of the 3-dimensional orthogonal group $O_3(\boldsymbol{R})$: $\boldsymbol{V}_k \cdot {}^t\boldsymbol{V}_k = {}^t\boldsymbol{V}_k \cdot \boldsymbol{V}_k = \boldsymbol{I}_3$ where $\boldsymbol{R}$ denotes the set of real numbers, and $\boldsymbol{I}_3$ the 3-dimensional identity matrix), and $\boldsymbol{t}$ is the 3D real-valued vector threshold value. The output signal $F(\boldsymbol{a})$ is defined to be :

$$F(\boldsymbol{a}) = \begin{bmatrix} f(a_1) \\ f(a_2) \\ f(a_3) \end{bmatrix}, \quad \text{where} \quad \boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad \text{and} \quad f(a_i) = \frac{1}{1+\exp(-a_i)}. \tag{2}$$

In the above formulation, various restrictions can be imposed on the 3D matrix, e.g. it can be regular, symmetric or orthogonal etc., which will influence the behavioral characteristics of the neuron. The weights are assumed to be orthogonal matrices because this assumption is a natural extension of the weights of the complex-valued neuron. We demonstrate this natural extension as follows. Consider a n-input complex-valued neuron with weights $w_k = w_k^r + iw_k^i \in \boldsymbol{C}$ $(1 \leq k \leq n)$ ($\boldsymbol{C}$ denotes the set of complex numbers, $i = \sqrt{-1}$) and a threshold value $\theta = \theta^r + i\theta^i \in \boldsymbol{C}$. Given input signals $x_k + iy_k \in \boldsymbol{C}$ $(1 \leq k \leq n)$, the neuron generates a complex-valued output value $X + iY$, where

$$\begin{aligned} X + iY &= f_C\left(\sum_{k=1}^{n}(w_k^r + iw_k^i)(x_k + iy_k) + (\theta^r + i\theta^i)\right) \\ &= f\left(\sum_{k=1}^{n}(w_k^r x_k - w_k^i y_k) + \theta^r\right) \\ &\quad + if\left(\sum_{k=1}^{n}(w_k^i x_k + w_k^r y_k) + \theta^i\right), \end{aligned} \tag{3}$$

and $f_C(z) = f(x) + if(y), \quad z = x + iy$. Furthermore,

$$\begin{bmatrix} X \\ Y \end{bmatrix} = F_C\left(\sum_{k=1}^{n} \begin{bmatrix} w_k^r & -w_k^i \\ w_k^i & w_k^r \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} \theta^r \\ \theta^i \end{bmatrix}\right), \tag{4}$$

where $F_C\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} f(x) \\ f(y) \end{bmatrix}$. In eqn (4), $\begin{bmatrix} w_k^r & -w_k^i \\ w_k^i & w_k^r \end{bmatrix}$ is an element of the two-dimensional orthogonal group $O_2(\boldsymbol{R})$. Considering eqn (4), the formulation of a neuron as given in eqns (1) and (2) above is natural.

2 N-Dimensional Vector Neuron

It is a matter of common occurrence that a vector is used in the real world, which represents a cluster of something, for example, a 4-dimensional vector consisting of height, width, breadth and time, and a N-dimensional vector consisting of N particles and so on. Then, we formulate a model neuron that can deal with N signals as one cluster, called ***N*-*dimensional vector neuron***, by extending the 3-dimensional vector neuron described above to N dimensions naturally.

2.1 N-Dimensional Vector Neuron Model

We will consider the following N-dimensional vector neuron with M inputs. The input signals, thresholds and output signals are all N-dimensional real-valued vectors, and the weights are all N-dimensional orthogonal matrices. The net input $\boldsymbol{u}$ to a N-dimensional vector neuron is defined as:

$$\boldsymbol{u} = \sum_{k=1}^{M} \boldsymbol{W}_k \boldsymbol{x}_k + \boldsymbol{\theta}, \tag{5}$$

where $\boldsymbol{x}_k$ is the k-th N-dimensional real-valued vector input signal, $\boldsymbol{W}_k$ is the N-dimensional orthogonal weight matrix for the k-th input signal $\boldsymbol{x}_k$ (that is, an element of the N-dimensional orthogonal group $O_N(\boldsymbol{R})$), and $\boldsymbol{\theta}$ is the N-dimensional real-valued vector threshold value. The N-dimensional real-valued vector output signal is defined to be:

$$1_N(\boldsymbol{u}) = \begin{bmatrix} 1_R(u^{(1)}) \\ \vdots \\ 1_R(u^{(N)}) \end{bmatrix}, \quad \text{where } \boldsymbol{u} = \begin{bmatrix} u^{(1)} \\ \vdots \\ u^{(N)} \end{bmatrix}$$
$$\text{and } 1_R(u) = \begin{cases} 1 & \text{if } u \geq 0 \\ 0 & \text{if } u < 0 \end{cases}. \tag{6}$$

2.2 Decision Boundary

We can find that a decision boundary of a N-dimensional vector neuron consists of N hyperplanes which intersect orthogonally each other, and divides a decision region into N equal sections, as that of a complex-valued neuron case. The net input $\boldsymbol{u}$ (eqn (5)) to a N-dimensional neuron with M inputs can be rewritten as:

$$\boldsymbol{u} = \sum_{k=1}^{M} \boldsymbol{W}_k \boldsymbol{x}_k + \boldsymbol{\theta}$$
$$= \begin{bmatrix} [\boldsymbol{w}_1^{(1)} \cdots \boldsymbol{w}_M^{(1)}] \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_M \end{bmatrix} + \theta^{(1)} \\ \vdots \\ [\boldsymbol{w}_1^{(N)} \cdots \boldsymbol{w}_M^{(N)}] \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_M \end{bmatrix} + \theta^{(N)} \end{bmatrix}, \tag{7}$$

where $\boldsymbol{w}_k^{(i)}$ is the i-th row vector of $\boldsymbol{W}_k$ $(i = 1, \cdots, N; k = 1, \cdots, M)$, and $\boldsymbol{\theta} = {}^t[\theta^{(1)} \cdots \theta^{(N)}]$. Thus,

$$u^{(i)}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M) \stackrel{\text{def}}{=} [\boldsymbol{w}_1^{(i)} \cdots \boldsymbol{w}_M^{(i)}] \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_M \end{bmatrix} + \theta^{(i)}$$
$$= 0 \tag{8}$$

is the decision boundary for the i-th component of an output of the N-dimensional vector neuron with M inputs ($i = 1, \cdots, N$). That is, input signals ${}^t[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M] \in \boldsymbol{R}^{MN}$ are classified into two decision regions $\{{}^t[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M] \in \boldsymbol{R}^{MN} \mid u^{(i)}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M) \geq 0\}$ and $\{{}^t[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M] \in \boldsymbol{R}^{MN} \mid u^{(i)}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M) < 0\}$ by the hyperplane given by eqn (8) ($i = 1, \cdots, N$). The normal vector $\boldsymbol{q}^{(i)}$ of the decision boundary for i-th component (eqn (8)) is given by ${}^t[{}^t\boldsymbol{w}_1^{(i)} \cdots {}^t\boldsymbol{w}_M^{(i)}]$ $(i = 1, \cdots, N)$, and it follows from the orthogonal property of the weight matrix $\boldsymbol{W}_k$ (i.e., $\boldsymbol{W}_k \cdot {}^t\boldsymbol{W}_k = {}^t\boldsymbol{W}_k \cdot \boldsymbol{W}_k = \boldsymbol{I}_N$($N$-dimensional identity matrix)) that the inner product of the normal vectors of the decision boundaries for any two distinct components is zero: for any $1 \leq i, j \leq N$ such that $i \neq j$,

$${}^t\boldsymbol{q}^{(i)} \cdot \boldsymbol{q}^{(j)} = [\boldsymbol{w}_1^{(i)} \cdots \boldsymbol{w}_M^{(i)}] \cdot \begin{bmatrix} {}^t\boldsymbol{w}_1^{(j)} \\ \vdots \\ {}^t\boldsymbol{w}_M^{(j)} \end{bmatrix} = 0. \tag{9}$$

Thus, the decision boundary of a N-dimensional vector neuron consists of N hyperplanes which intersect orthogonally each other.

2.3 N-Bit Parity Problem

We will find a solution to the N-bit parity problem, using a single N-dimensional vector neuron with the orthogonal decision boundary with the highest generalization ability. Minsky and Papert clarified the limitations of a single real-valued neuron: in a large number of interesting cases, a single real-valued neuron is incapable of solving the problems [Minsky and Papert, 1969]. The most difficult problem among them is the parity problem, in which the output required is 1 if the input pattern contains an odd number of 1s and 0 otherwise.

Rumelhart, Hinton and Williams showed that the *3-layered* real-valued neural network (i.e., with one hidden layer) can solve the parity problem [Rumelhart *et al.*, 1986b]. As described above, the parity problem cannot be solved with a single real-valued neuron. Then, it will be proved that the parity problem can be solved by a single N-dimensional vector neuron with the orthogonal decision boundary. Rumelhart, Hinton and Williams showed that increasing the number of layers made the computational power of neural networks high. We will show that extending the dimensionality of neural networks to N dimensions originates the similar effect on neural networks.

2.4 A Solution

The input-output mapping in the N-bit parity problem is shown in Table 1(a). In order to solve the N-bit parity problem with N-dimensional vector neurons, the input-output mapping is encoded as shown in Table 1(b) where the outputs ${}^t[\,0\ \ 0\ \ 0 \cdots\ 0\ \ 0\ \ 0]$, ${}^t[\,0\ \ 0\ \ 0 \cdots\ 0\ \ 1\ \ 1]$, ${}^t[\,0\ \ 0\ \ 0 \cdots\ 1\ \ 0\ \ 1]$, ${}^t[\,0\ \ 0\ \ 0 \cdots\ 1\ \ 1\ \ 0]$, ... are interpreted to be 0, and ${}^t[\,0\ \ 0\ \ 0 \cdots\ 0\ \ 0\ \ 1]$, ${}^t[\,0\ \ 0\ \ 0 \cdots\ 0\ \ 1\ \ 0]$, ${}^t[\,0\ \ 0\ \ 0 \cdots\ 1\ \ 0\ \ 0]$, ... are interpreted to be 1 of the original N-bit parity problem (Table 1(a)), respectively. We use a single N-dimensional vector neuron with only one input with a weight

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & \cdots & w_{1N} \\ \vdots & & \vdots \\ w_{N1} & \cdots & w_{NN} \end{bmatrix} \in O_N(\boldsymbol{R}) \qquad (10)$$

(we assume that it has no threshold parameters). The decision boundary of the N-dimensional vector neuron described above consists of the following N hyperplanes which intersect orthogonally each other:

$$\begin{aligned} [w_{11} \ \cdots \ w_{1N}] \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} &= 0, \\ &\vdots \\ [w_{N1} \ \cdots \ w_{NN}] \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} &= 0 \end{aligned} \qquad (11)$$

for any input signal $\boldsymbol{x} = {}^t[x_1 \ \cdots \ x_N] \in \boldsymbol{R}^N$. The N equations of eqn (11) are the N decision boundaries for the N components of the output of the N-dimensional vector neuron, respectively. Fig. 1 shows an example of the decision boundary of the N-dimensional vector neuron ($N = 2$ for the sake of simplicity).

Letting $w_{ii} = 1$ $(i = 1, \cdots, N)$ and $w_{ij} = 0$ $(i \neq j)$ (i.e., the weight $\boldsymbol{W}$ is the N-dimensional identity matrix), we can find that the N-dimensional vector neuron implements the input-output mapping shown in Table 1(b), the decision boundary of which consists of the orthogonal N hyperplanes

$$\begin{aligned} x_1 &= 0, \\ &\vdots \\ x_N &= 0 \end{aligned} \qquad (12)$$

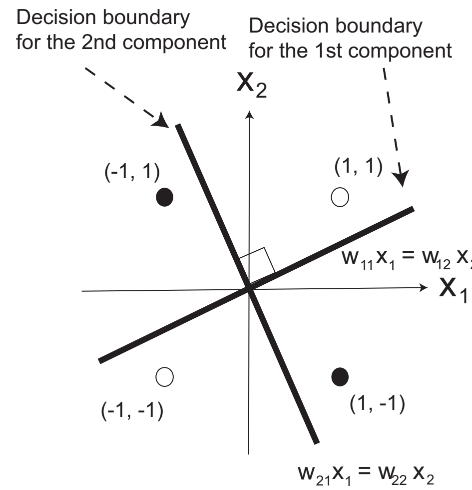

Figure 1: An example of the decision boundary in the input space of the 2-dimensional vector neuron (i.e., $N = 2$). The black circle means that the output in the parity problem is 1, and the white one 0.

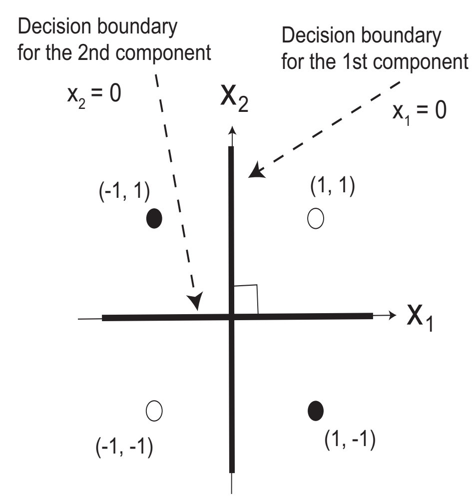

Figure 2: The decision boundary in the input space of the 2-dimensional vector neuron that solves the 2-parity problem (i.e., $N = 2$). The black circle means that the output in the parity problem is 1, and the white one 0.

and divides the input space (the decision region) into 2^N equal sections, and has the highest generalization ability for the N-bit parity problem. Fig. 2 shows an example of the decision boundary for the 2-bit parity case.

3 Discussion

There exist some neural network models that can solve the N-bit parity problem. The comparison between our result and the previous works is shown in Table 2. The number of neurons of the N-dimensional vector neuron and Aizenberg's model is 1 constantly whereas those of the other models increase as N increases on the order N. The number of parameters of the Lavretsky's model is the least, but the number of layers of which is $N-1$ which increases as N increases. The number of parameters of the Stork and Allen's model is on the order N, but the activation function of the hidden neurons of which is considerably complicated:

$$f(x) = \frac{1}{N}\left(x - \frac{\cos(\pi x)}{\alpha x}\right), \tag{13}$$

where α is a constant greater than 1.0. The number of parameters of the N-dimensional vector neuron is the least among the models on the order N^2. The number of layers of the N-dimensional vector neuron and Aizenberg's model is only 2, which is the least. As described above, the Aizenberg's model seems to be the best totally, but its activation function is somewhat special. Thus, we can conclude that the N-dimensional vector neuron proposed in this paper is the best totally among the models with the traditional activation functions such as a step function. It should be emphasized here that the number of neurons needed is only one (i.e., a single neuron).

4 Conclusions

We cannot point out for the present that the N-dimensional vector neuron is a plausible model of brains. However, a solution to the N-bit parity problem with a single N-dimensional vector neuron suggests that making the dimensionality of neural networks high (for example, complex numbers and quaternions [Nitta,1995] is a new directionality for enhancing the ability of neural networks, and that it is worth researching the neural networks with high dimensional parameters, keeping the association with brains in mind.

References

[Rumelhart *et al.*, 1986a] Rumelhart, D. E., Hinton, G. E. & Williams, R. J. Learning representations by back-propagating erroes. *Nature*, **323**, pp.533-536, 1986.

[Rumelhart *et al.*, 1986b] Rumelhart, D. E., Hinton, G. E. & Williams, R. J. *Parallel Distributed Processing*, **1** (MIT Press, Cambridge, 1986).

[Aizenberg *et al.*, 1971] Aizenberg, N. N., Ivaskiv, Yu. L., Pospelov, D. A. & Hudiakov, G. F. Multiple-valued threshold functions, boolean complex-threshold functions and their generalization. *Kibernetika*(*Cybernetics*) **4**, pp.44-51 (1971) (in Russian).

[Widrow *et al.*, 1975] Widrow, B., McCool, J. & Ball, M. The complex LMS algorithm. *Proc. the IEEE* **63**, pp.719-720, 1975.

[Hirose *et al.*, 2002] Hirose, A. et al. 6 special session papers. in *Proc. International Conference on Neural Information Processing*, **3**, Singapore, pp.1074-1103, 2002.

[Nitta, 2001] Nitta, T. in *Knowledge-based Intelligent Information Engineering Systems & Allied Technologies* (eds. Baba, N., Jain, L. C. & Howlett, R. J.), Part I, pp.550-580 (IOS Press, Tokyo, 2001).

[Kuroe *et al.*, 2002] Kuroe, Y. et al. in *Knowledge-based Intelligent Information Engineering Systems & Allied Technologies* (eds. Damiani, E., Howlett, R. J., Jain, L. C. & Ichalkaranje, N.), Part I, pp.623-647 (IOS Press, Amsterdam, 2002).

[Nitta,1997] Nitta, T. An extension of the back-propagation algorithm to complex numbers. *Neural Networks*, **10**, pp.1392-1415, 1997.

[Nitta, 2003] Nitta, T. Solving the XOR problem and the detection of symmetry using a single complex-valued neuron. *Neural Networks*, **16**, pp.1101-1105, 2003.

[Nitta, 2000] Nitta, T. An analysis on fundamental structure of complex-valued neuron. *Neural Processing Letters*, **12**, pp.239-246, 2000.

[Minsky and Papert, 1969] Minsky, M. L. & Papert, S. A. *Perceptrons* (MIT Press, Cambridge, 1969).

[Nitta and Garis, 1992] Nitta, T. & Garis, H. D. A 3D vector version of the back-propagation algorithm. *Proc. IEEE/INNS International Joint Conference on Neural Networks*, **2**, Beijing, pp.511-516, 1992.

[Setiono, 1997] Setiono, R. On the solution of the parity problem by a single hidden layer feedforward neural network. *Neurocomputing*, **16**, pp.225-235,1997.

[Stork and Allen, 1992] Stork, D. G. & Allen, J. D. How to solve the N-bit parity problem with two hidden units. *Neural Networks*, **5**, pp.923-926,1992.

[Minor, 1993] Minor, J. M. Parity with two layer feedforward nets. *Neural Networks*, **6**, pp.705-707, 1993.

[Lavretsky, 2000] Lavretsky, E. On the exact solution of the parity-N problem using ordered neural networks. *Neural Networks*, **13**, pp.643-649, 2000.

[Liu *et al.*, 2002] Liu, D., Hohil, M. E. & Smith, S. H. N-bit parity neural networks: new solutions based on linear programming. *Neurocomputing*, **48**, pp.477-488, 2002.

[Nitta,1995] Nitta, T. A quaternary version of the back-propagation algorithm. *Proc. IEEE International Conference on Neural Networks*, **5**, pp.2753-2756, 1995.

[Aizenberg *et al.*, 1996] Aizenberg, N. N., Aizenberg, I. N. & Krivosheev, G. A. CNN based on universal binary neurons: learning algorithm with error-correction and application to impulsive-noise filtering on gray-scale images, *Proc. IEEE Int. Workshop on Cellular Neural Networks and their Applications*, Seville, Spain, pp.309-314, 1996.

Input							Output
x_1	x_2	x_3	$\cdots$	x_{N-2}	x_{N-1}	x_N	y
0	0	0	$\cdots$	0	0	0	0
0	0	0	$\cdots$	0	0	1	1
0	0	0	$\cdots$	0	1	0	1
0	0	0	$\cdots$	1	0	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
0	0	1	$\cdots$	0	0	0	1
0	1	0	$\cdots$	0	0	0	1
1	0	0	$\cdots$	0	0	0	1
0	0	0	$\cdots$	0	1	1	0
0	0	0	$\cdots$	1	0	1	0
0	0	0	$\cdots$	1	1	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
0	1	1	$\cdots$	0	0	0	0
1	0	1	$\cdots$	0	0	0	0
1	1	0	$\cdots$	0	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	1	1	$\cdots$	1	1	1	$\begin{cases} 1 & \text{(if } N \text{ is odd)} \\ 0 & \text{(if } N \text{ is even)} \end{cases}$

Table 1(a) The N-bit parity problem

Input							Output
x_1	x_2	x_3	$\cdots$	x_{N-2}	x_{N-1}	x_N	y
-1	-1	-1	$\cdots$	-1	-1	-1	${}^t[\,0\ 0\ 0 \cdots 0\ 0\ 0\,]$
-1	-1	-1	$\cdots$	-1	-1	1	${}^t[\,0\ 0\ 0 \cdots 0\ 0\ 1\,]$
-1	-1	-1	$\cdots$	-1	1	-1	${}^t[\,0\ 0\ 0 \cdots 0\ 1\ 0\,]$
-1	-1	-1	$\cdots$	1	-1	-1	${}^t[\,0\ 0\ 0 \cdots 1\ 0\ 0\,]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
-1	-1	1	$\cdots$	-1	-1	-1	${}^t[\,0\ 0\ 1 \cdots 0\ 0\ 0\,]$
-1	1	-1	$\cdots$	-1	-1	-1	${}^t[\,0\ 1\ 0 \cdots 0\ 0\ 0\,]$
1	-1	-1	$\cdots$	-1	-1	-1	${}^t[\,1\ 0\ 0 \cdots 0\ 0\ 0\,]$
-1	-1	-1	$\cdots$	-1	1	1	${}^t[\,0\ 0\ 0 \cdots 0\ 1\ 1\,]$
-1	-1	-1	$\cdots$	1	-1	1	${}^t[\,0\ 0\ 0 \cdots 1\ 0\ 1\,]$
-1	-1	-1	$\cdots$	1	1	-1	${}^t[\,0\ 0\ 0 \cdots 1\ 1\ 0\,]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
-1	1	1	$\cdots$	-1	-1	-1	${}^t[\,0\ 1\ 1 \cdots 0\ 0\ 0\,]$
1	-1	1	$\cdots$	-1	-1	-1	${}^t[\,1\ 0\ 1 \cdots 0\ 0\ 0\,]$
1	1	-1	$\cdots$	-1	-1	-1	${}^t[\,1\ 1\ 0 \cdots 0\ 0\ 0\,]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	1	1	$\cdots$	1	1	1	${}^t[\,1\ 1\ 1 \cdots 1\ 1\ 1\,]$

Table 1(b) An encoded N-bit parity problem

	The number of neurons	The number of parameters	The number of layers	Direct link	Activation function
Ours	1	$\frac{N}{2}(N-1)$	2	No	Step function
Aizenberg [Aizenberg *et al.*, 1996]	1	$2N+2$	2	No	Somewhat special
Setiono [Setiono, 1997]	$\frac{3}{2}(N+1)$ or more	$\frac{N}{2}(N-1)+2(N+1)$ or more	3	No	Sigmoidal function
Stork and Allen [Stork and Allen, 1992]	$N+3$	$2N+4$	3	No	Considerably complicated
Minor [Minor, 1993]	$\frac{1}{2}(3N+1)$ or more	$\frac{N}{2}(N-1)+(N-1)$	3	Yes	Sigmoidal function
Lavretsky [Lavretsky, 2000]	$N-1$	$2N-1$	$N-1$	Yes	Sigmoidal function
Liu et al. [Liu *et al.*, 2002]	$\frac{1}{2}(3N+1)$ or more	$\frac{N}{2}(N+3)$ or more	3	Yes	Step function

Table 2 The comparison between our result and the previous works. *The number of layers* includes an input layer; it is 3 if the network has one hidden layer. *Direct link* means that there are at least one direct link between the input layer and the output layer in the neural network with at least one hidden layer. Note that the number of parameters in Aizenberg's work in the table is the estimated one by the author because Aizenberg et al. solved only the 3, 8 and 9-bit parity problems with a single complex-valued neuron.