West Indian Journal of Engineering Vol. 25, No. 1, (July 2002) Technical Paper (Kong) 52 - 62

Real Sparse Distributed Memory

A. Kong*

A neural network model is described which uses the principles of Kanerva's sparse distributed memory (SDM) [1], but which functions on data sets consisting of real number vectors as opposed to binary vectors. This limitation to binary patterns is considered one of the main drawbacks of Kanerva's SDM [2]. The model involves the development of a linear threshold function for neurons that allow them to act as address decoders of data sets of real number vectors. Although developed independently of any investigation into the physiological behaviour of natural neurons, subsequent investigation revealed a possible mechanism for this model for which a linear approximation can be made. This involves a study of the integration of excitory postsynaptic potentials (EPSP) and inhibitory postsynaptic potential (IPSP) of a postsynaptic neuron which make multiiple synapses with the terminal fibres of a single axon of another presynaptic neuron.

1. Introduction

A main drawback of Kanerva's *Sparse Distributed Memory* theory [1] is its predication on the premise of data sets consisting of *binary numbers* or *Boolean vectors*. This limitation has perhaps prevented this theory from entering the mainstream of neural network research, where popular networks, e.g., *Multilayer Perceptrons* with backpropogated learning routinely work with *real number data sets*. The familiarity with and therefore preference for the use of real numbers in computation is indisputable. The criticisms of Kanerva [1] of networks which require adjustable input weights (such as Multilayer Perceptron networks, for example), are also compelling. The essence of which is that these networks require a mechanism just as complex as the network itself for training. Weight adjustment (using a complicated algorithm) in these models is performed in training mode and produces the desired linear separation of data sets. However, the above drawback is far more serious given the preference for working with real numbers mentioned.

This paper describes a model (**Section 2**) which uses Kanerva's [1] principles of *sparse distributed memory* (**Section 4**), but which functions on data sets consisting *of real number vectors* instead of *Boolean vectors* [1]. The basis of the model is that there is a specific setting of real number weights and real number threshold of a neuron (called its *tuning* by the author). This setting is such that, it *fires* (or responds) to only one real number vector input, out of the almost infinite number of possible inputs that may be presented to the neuron. At this setting, the neuron is said to behave as an *address decoder* (**Section 3**) and to be tuned to a specific address (the neuron may be called an *address decoder neuron*). Subsequent relaxation (i.e., reduction) of the *tuning* (or threshold) of a tuned *address decoder neuron*, would admit responses from inputs other than the one to which it was tuned, but only if, such inputs do not deviate too much from that input. The amount of deviation permitted that solicits a response from the neuron is dependent on the amount of relaxation of the *tuning*.

Kanerva's [1] methods remain relevant to the function of the new neural network model as a *sparse distributed memory* model (**Section 4**). More recent work in this area is described in [3]. The essence of which is that for some input, there exists a number of neurons out of a population of neurons, which would respond by firing, because they are tuned to this input or to inputs close by in address space with their tuning sufficiently relaxed. Some data with which one would like the input associated, can be stored redundantly at those neurons which fire (one hypothesis of Kanerva is that such storage takes place at synapses along the axons of the neurons). If the same input is presented to the network subsequent to data storage, then the same set of neurons would fire and the associated data recalled. The stochastic nature of the process ensures

Pertinent discussion will be published in January, 2003 West Indian Journal of Engineering if received by November, 2002.

that recall of stored data is possible even with inexact cues, by a recursive reading mechanism. Kanerva's processes for writing to and reading from memory can be applied almost without modification (in essence just the type of data to be stored changes from integers to real numbers). The new model (**Section 2**) although functionally consistent with its requirements for construction of a SDM appeared so different from other models, that it obligated a search for a physiological basis.

In the search for such a basis, the synaptic connection of artificial neurons and natural neurons were examined and compared. Very often, for artificial neural network systems used in problem-solving in the literature, the output of a single unit (or neuron) in an *input layer* is observed to be presented to a single input of each of the units of a *hidden layer* or layers. The path from one neuron to another, with which a real numerical weight is invariably associated, is modelled after a synapse in natural neurons. It is rare, if at all, that the outputs of a single input unit is observed to be presented to many different inputs of the *same unit* in the *hidden layer*. That is, in addition to be presented to many other units of the *hidden layer* (where it can again make multiple synaptic connections with each unit).

Although rare in artificial neural networks described in the literature, this configuration is more common in the literature on *natural neural networks*. In this literature, the end fibres of an axon of one neuron is obseved to make many synapses (for example, through dendrites) with another *postsynaptic neuron* instead of just one synapse [4]. This is depicted in **Figure 3(b)** (also see **Section 5**). For proper functionality of the model described in this paper, this sort of connection is the rule rather than the exception. That is, terminal fibres of an axon of a *presynaptic neuron* in an input layer is not allowed to make less than two synapses with each *postsynaptic neuron* in a hidden layer. The behaviour of the multiple synapses can be integrated into one function and one input into the neuron (or unit) of the hidden layer.

2. Artificial Neuron Model

A *hidden layer neuron* is approximated by the block diagram shown in **Figure 1** where $w_i \in \left\{ r \in R \,\middle|\, 0 \le r \le \max_{w_i} \right\}$ are fixed for each neuron (Kanerva's model differs from almost all other models which allow tuning of input weights, in that, weights are fixed). Similarly, $x_i \in \left\{ r \in R \,\middle|\, 0 \le r \le \max_{x_i} \right\}$ are variable (inputs range from no input, $x_i = 0$, to some maximum intensity, $x_i = \max_{x_i}$). The constant $c \in R$ is called its threshold. Its output is the *Boolean function,*

$$f = \begin{cases} 1 & if \sum_{i=1}^{m} S_i \ge c \\ 0 & otherwise \end{cases} \qquad (1)$$

Where,

$$S_i = \begin{cases} x_i & if\ x_i \le w_i \\ 2w_i - x_i & otherwise \end{cases} \qquad (2)$$

For an output of 1, the neuron is said to *fire*. The function S_i is shown in **Figure 2** for some $w_i > 0(a)$ and $w_i = 0(b)$. For $w_i = 0$ and any input x_i, and from Equation (1), S_i tends to *inhibit firing* of the neuron since $S_i < 0(b)$, whereas, for some $w_i > 0$, S_i tends to *encourage firing* (excitory) for inputs $x_i < 2w_i$ (i.e., $S_i > 0$), and to *inhibit firing* for $x_i > 2w_i (S_i < 0)$. Thus S_i can be described as either *excitory* or *inhibitory* depending on w_i and x_i. For fixed weights, w_i, S_i is a function of x_i.

This model can be compared to the model used by Kanerva (Equation {3}) for which weights $w_i \in \{1, -1\}$ and inputs $x_i \in \{0,1\}$ (i.e., a binary vector). Note that both models, Equations (1) and (3) are idealised models with no time delays between presentation of inputs and firing.

$$f = \begin{cases} 1 & if \sum_{i=1}^{m} w_i x_i \ge c \\ 0 & otherwise \end{cases} \qquad (3)$$

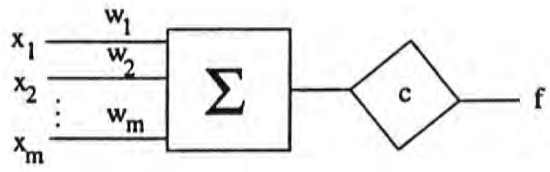


FIGURE 1: *Artificial Neuron*

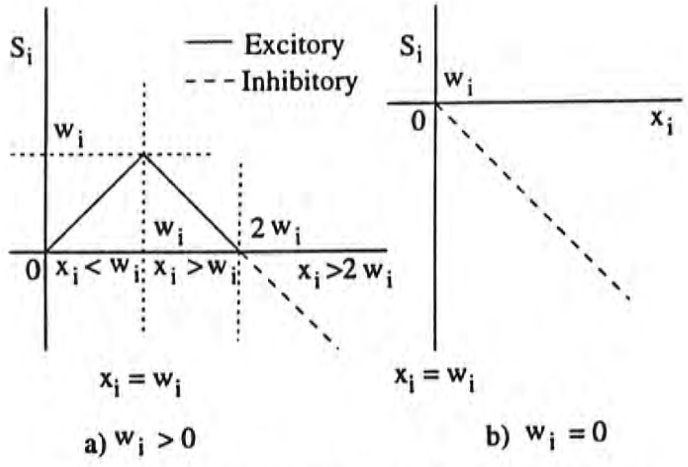


FIGURE 2: *S_i as a Function of x_i for Different Settings of w_i*

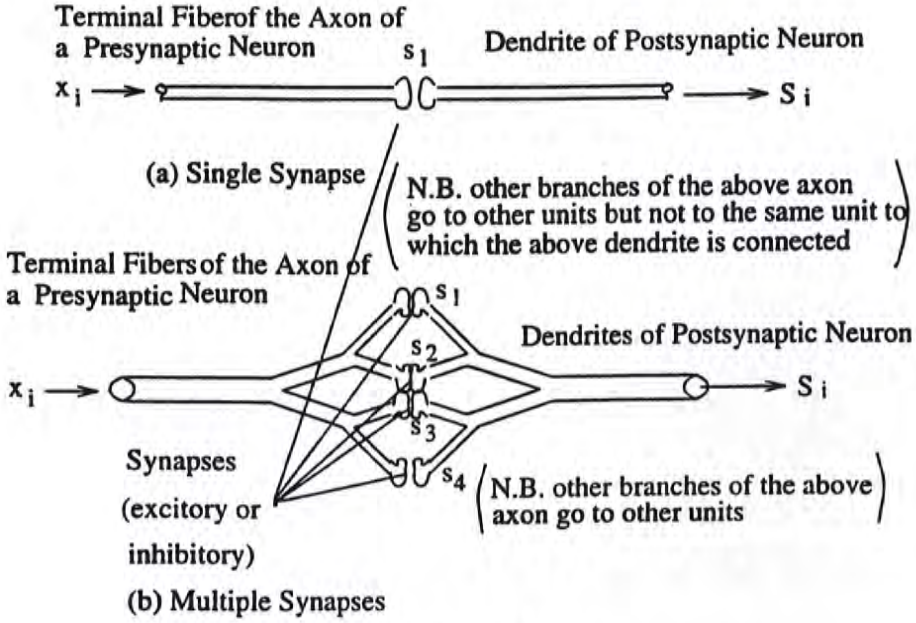


FIGURE 3: *Synaptic Interface*

The neuron separates all possible inputs into two sets, P, the set of inputs for which the neuron fires (the firing set) and Q, the set of inputs for which the neuron does not fire (the non-firing set), said to be linearly separable by the neuron.

3. Address Decoder Function

The artificial neuron model may function as an address decoder since from Equation (1), for threshold, $c = \sum_{i=1}^{m} w_i$, the cardinality of set P is one (i.e., *card* P=1), in which case the neuron fires for only one unique input $[x_i \in R, i = 1,\ldots,m \mid x_i = w_i]$ and no other. That is for $c = \sum_{i=1}^{m} w_i$ the address of the neuron is given by $[w_1,\ldots,w_m]$ and the neuron fires for one input, $[x_1,\ldots,x_m]$ such that, $x_i = w_i$, which corresponds to its address.

The size of the address space, N (the notation N will actually be used to refer to both the space itself as well as its size according to context), is given by $\prod_{i=1}^{m}(\max_{x_i} / \Delta x_i)$, where Δx_i is the size of the smallest increments with which x_i may change. Thus provided $\exists \max_{x_i} \cdot \max_{x_i} > 0$, then $N \to \infty$ if $\Delta x_i \to 0$.

The difference between points **x** and **y** of address space

$N, x - y = [\,|x_1 - y_1|, |x_2 - y_2|, \ldots, |x_m - y_m|\,]$

(note the difference is commutative, i.e., **x** - **y** = **y** - **x**). The *Hamming distance* between points **x** and **y**, d(**x**,**y**), is then

$$|x - y| = |x_1 - y_1| + |x_2 - y_2| + \cdots + |x_m - y_m|$$

If an artificial neuron with tuning $c = \sum_{i=1}^{m} w_i$, whereby it responds to only one input $x = [x_i, i = 1,\ldots,m \mid x_i = w_i]$ were to relax its threshold, i.e., $c < \sum_{i=1}^{m} w_i$, then the cardinality of P (the firing set) increases with increased relaxation of the threshold. If the threshold is reduced by an amount r, i.e., $c = \sum_{i=1}^{m} w_i - r$, then P would consist of all addresses $y \in P$, such that d (**x**,**y**) $\le r$. That is P would consist of all addresses within the circle centre **x** of radius r of address space N.

Its ability to decode (i.e., respond to) a unique input can be seen from **Figure 2**. In **Figure 2**, a neuron with address $[w_1,\ldots,w_m]$ and tuning $c = \sum_{i=1}^{m} w_i$ responds to input $[x_1,\ldots,x_m]$, with x_i corresponding to the peak of the curve shown in **Figure 2(a)** or the start of the curve (**Figure 2(b)**), depending on $w_i (i.e., x_i = w_i)$. Any deviation from w_i, whether x_i increases or decreases, results in a reduction in S_i, and therefore inhibition of firing of the neuron which is already at its threshold. If however, the threshold is reduced by r, then a deviation in input from its address $y = [(x_i + \Delta x_i) \in R \mid x_i = w_i]$, may be tolerated (i.e., still result in firing of the neuron), provided the reduction of $\sum_{i=1}^{m} S_i$ as a result, which is $\sum_{i=1}^{m} |\Delta x_i|$, is less than or equal to r. Note that $\sum_{i=1}^{m} |\Delta x_i|$ is the *Hamming distance d (x,y)*.

Similarly, an increase in threshold above $c = \sum_{i=1}^{m} w_i$ (i.e., $c > \sum_{i=1}^{m} w_i$) gives *card* $P = 0$ (i.e., P={}), that is, no input can be found for which the neuron fires. A neuron can thus be tuned to respond to P between and inclusive of extremes of none of the address space, one address, a set of addresses, or the entire address space, by choice of its threshold.

4. Sparse Distributed Memory

Neurons which may fire for inputs corresponding to any member of a circle of addresses in address space, defined by the choice of weights and the setting of thresholds, is the basis behind *sparse distributed memory*. This means that for the set of neurons, a single input presented simultaneously to all neurons would result in several neurons firing. That is provided, weights are chosen such that their addresses are uniformly distributed throughout address space, and thresholds chosen such that a sufficient portion of the space is covered by each neuron (i.e., for which the neuron would respond). When a neuron fires, then some data may be associated with the neuron. If that data is the same as the input, then this is called *autoassociation*. If however, this data is different from the input, then this is called *heteroassociation*. Both

autoassociative and *heteroassociative* data may be connected to each neuron depending on the application.

Although *heteroassociative data* may be calculated using *regression algorithms* making their application domains very similar to other networks, for example, *Radial Basis Function* networks they differ in several important areas. Most important of which is the differentiation of network *addressing* and *function approximation* as a result of their *weightlessness*, which are not distinguished in almost all other network types.

Since the addresses of neurons are few and far between in address space (which can be of almost infinite size, then memory is said to be *sparse*. Memory is also *distributed*, since invariably many neurons (which are simulated as processing elements in computer systems) take part in a single operation whether writing to or reading from memory. The mechanisms for writing to and reading from memory remain valid. An application is discussed in [5] that applies the principle to *optimal control* of the *non-linear dynamics* of a *3-joint robotic manipulator.*

Further work involve implementing the network using *high dimensional sparse asymmetric adaptive component cell arrays* which have the potential for real time solutions to large problems without the need for specialised computer architectures. Large problems (high dimensional, i.e., network with much input) suffer from what is called the *combinatorial explosion problem* and the *SDM concept* is instrumental in overcoming this problem.

5. Synaptic Integration

Note that the intensity of input is coded as a sequence of action potentials of a frequency that varies with intensity. This sequence produces a *presynaptic input* of a potential that varies with intensity due to *temporal summation* of action potentials. In [6], the behaviour of *excitory post synaptic potential* (EPSP) in response to *presynaptic inputs* of varying potential were discussed, where it was indicated that the function describing this behaviour was exponential (or logarithmic) and had a clear threshold. It was also indicated that this threshold varied from one synapse to another. Since this behaviour is related to the quanta of chemical transmitter released per synapse, then this implies that the same behaviour holds for *inhibitory post synaptic potentials* (IPSP), which can be said to also have clear thresholds varying from one synapse to another. This exponential function can be approximated by the *logistic activation function* (or sigmoid function) defined by Equations (4) and (5) in the **Appendix*** and depicted in **Figure 4(a)** and **(b)** for excitory and inhibitory synapses respectively.

A common configuration is one where there is only one synapse between the output of one unit (through a terminal fibre of its axon) and an input of another unit (for example, through a dendrite) as shown in **Figure 3(a)**. For this, the *postsynaptic potential* whether *excitory* or *inhibitory* produced for some *presynaptic input* would resemble curves shown in **Figure 4(a)** and **(b)**, respectively. This model of the interface between one unit and another, is inconsistent with the model desired as described in Equation (1) and depicted in **Figure 2** which requires, that the output S_i rises to a maximum at $x_i = w_i$. In this model (Equations {4} and {5}), S_i rises (or falls for an inhibitory synapse) rapidly beyond a certain threshold of x_i. If x_i continues to increase beyond this threshold, then S_i also continues to increase (or decrease for an inhibitory synapse) until saturation, but never falls again.

Another configuration is one where a *postsynaptic neuron* is considered to make *multiple synapses* with each axon from other *presynaptic neurons* (e.g., receptor cells) from which it receives its input, instead of just one, as shown in **Figure 3(b)**, some of which are *excitory* and some *inhibitory*. In this case, the *postsynaptic potential* as a result of these multiple synapses can be approximated (linear summation is assumed [7]) by the integration or summation of the postsynaptic potentials of each synapse. This is possible since, as pointed out in [8,9,10], although the transmitter released at each synapse of a presynaptic neuron is the same, the nature of the receptors in the postsynaptic neurons can be *pharmacologically distinct* and control different ionic channels. As a result, a presynaptic cell can mediate *opposite* (excitory and inhibitory) synaptic actions to a single postsynaptic cell.

It may be shown that the integration of excitory and inhibitory postsynaptic potential resulting from multiple synapses with a single axon can result in a similar but non-linear characteristic to the one

* See **Appendix** on page 61.

described by the function S_i. Let signal S_i be the result of integration (summation) of the postsynaptic potentials of the synapses, some of which may be excitory and some inhibitory. By a hypothetical example, it would be shown that S_i exhibits a maximum for some input x_i.

Consider the case where there are only two synapses in the interface (note the interface of **Figure 3** has four), one excitory (s_1), and the other inhibitory (s_2). Both can be modelled by logistic activation functions with diferent thresholds, as shown in **Figure 4(a)** and **(b)**. The result of synaptic integration is shown in **Figure 4(c)**. This curve exhibits a maximum at $x_i \approx 5$, as can be seen from **Figure 4(c)**, and although non-linear, is similar to the one shown in **Figure 2(a)**, in the existence of input x_i, for which the output S_i is a maximum.

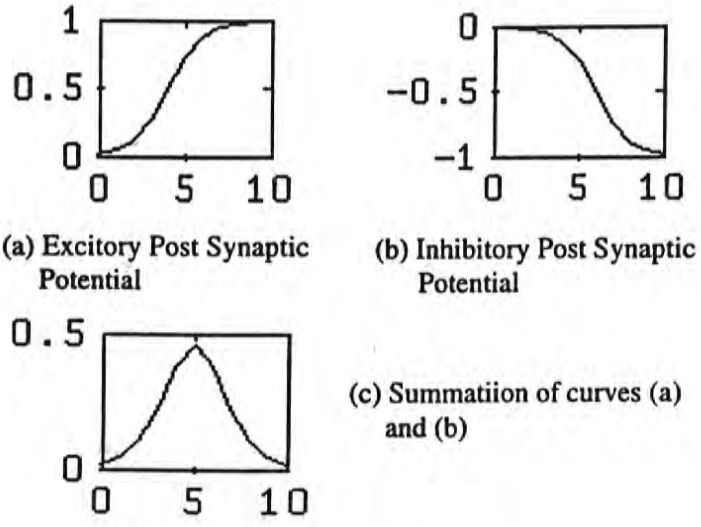
The value x_i for which S_i is a maximum can be changed, by changing the settings of the thresholds and gains of individual synaptic activation functions, and can be determined analytically given the thresholds and gains for interfaces with two synapses. From Equation (14), a value x_i for which S_i is a maximum exists, and is a real number provided $\theta_2 > \theta_1$. For example, the location of the maximum for $\theta_1 = 4$, $\theta_2 = 6$, and $k_1 = k_2 = 1$, is calculated as x_i=5.0727 (see **Figure 4(c)**). The location of the maximum can be raised or lowered by choice of θ_1 and $\theta_2 (\theta_2 > \theta_1)$, for example, $\theta_1 = 1$ and $\theta_2 = 3 (k_1 = k_2 = 1)$ gives a maximum at $x_i = 2.0715$, similarly, $\theta_1 = 7$ and $\theta_2 = 9 (k_1 = k_2 = 1)$ gives a maximum at $x_i = 8.0727$.

The above analysis can be extended to the case of interfaces with more than two synapses as shown in **Figure 3(b)**, which has four synapses. Analytic solution of the type given in Equation (14) becomes more difficult, as the number of synapses increases. However, it can be shown empirically, that S_i, for multiple synapses also exhibit some maximum point, provided that the sum of the thresholds of inhibitory synapses exceeds the sum of the thresholds of excitory synapses. This can be shown in **Figure 5**, where six curves are displayed representing the behaviour of six synaptic interfaces, for inputs $0 < x_i < 10$, each with 20 synapses, $\{s_1, \ldots, s_{20}\}$. Of these, there are equal numbers of excitory and inhibitory synapses and thresholds chosen randomly, in pairs, for excitory and inhibitory synapses, such that, the thresholds for inhibitory synapses exceed those of excitory synapses, and the gains $\{k_1, \ldots, k_{20}\}$ also chosen randomly. Such a scenario of more than two synapses per interface makes the S_i function more flexible (where flexibility increases with the increasing numbers of synapses n). That is, in addition to the location of the maximum, the steepness of ascent and descent from the maximum point can be selected by choice of thresholds $\{\theta_1, \ldots, \theta_n\}$ and gains $\{k_1, \ldots, k_n\}$.

If a linear approximation is used for the behaviour of S_i, and the gradient of ascent to and descent from the maximum is considered the same for each S_i, which is equal to 1 and -1 respectively, then the approximate model described is Equation (1) is obtained. Although these simplifications maintain the functionality required of an SDM, while facilitating ease of simulation on a digital computer, some flexibility is lost. For example, the ability to determine selectively the shapes of the S_i curves for each input, means that, the sensitivity of the neuron to changes in a particular input (such as, $x_i = w_i \pm \Delta x_i$), need not be the same for all inputs as it is in the approximate model. Thus, a neuron can be made more sensitive to changes in one direction (e.g., $x_i = w_i \pm \Delta x_i$) than another (e.g., $x_i = w_i - \Delta x_i$). This can be seen for the curve with the lowest maximum point in **Figure 5** (a dashed curve), where clearly, the slopes away from the turning point are steeper on one side than the other. This flexibility is lost in the approximations made in the model described by Equation (1).

6. Resource Requirements

The model described in Equation (1) can be used in construction of a sparse distributed memory which stores patterns *as real number vectors* instead of *Boolean vectors*, as in Kanerva's original model. The cost of the switch from working with data sets of binary numbers, to data sets of real numbers is relatively *inexpensive* when compared to the benefits



(a) Excitory Post Synaptic Potential

(b) Inhibitory Post Synaptic Potential

(c) Summatiion of curves (a) and (b)

FIGURE 4: *Synaptic Integration*

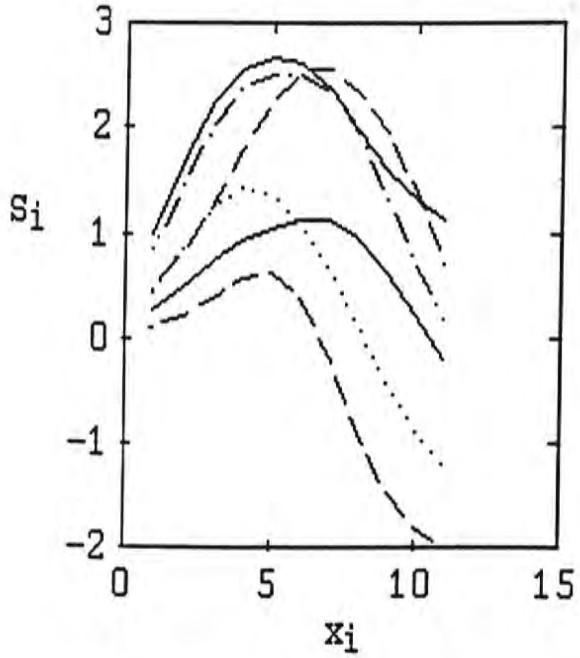


FIGURE 5: *Multiple Synapses*

that working with real numbers provide. When Equation (1) and (3) are compared, it is easy to see and just as easy to demonstrate using a computational tool which counts floating point operations (FLOPs), e.g., MATLAB, that the number of FLOPs required to evaluate Equation (3) is equal to $2m$. The variable m is the number of network inputs (one FLOP for each multiplication of w_i and x_i, and another for the addition of the product to the sum, for each input). The number of FLOPs required for simulation of Equation (1) however, depends on the input. This ranges from a maximum of $3m$ to a minimum of

7. Conclusion

A new artificial neural network model is described. This model can be used to construct a *sparse distributed memory*, similar to the one described by Kanerva, but which can function on data sets consisting of *real number vectors*, as opposed to *binary number vectors*. It was shown that this model is not inconsistent with the theoretical behaviour and synaptic connections of *natural neurons*, in their fundamental ability to *integrate postsynaptic potentials*, both excitory and inhibitory, due to *multiple dual action synapses* with terminal fibres of axons from other presynaptic neurons, thereby allowing tuning. There was however, no firm evidence that natural neurons actually function *as address decoders*, although, Lansner [11] noted that individual neurons often respond selectively in a narrow magnitude interval or to specific forms of stimuli. Recent work at MIT however confirms that neurons are actually tuned to concepts [12]. The extension of Kanerva's theory to allow an SDM that is capable of learning patterns consisting of vectors of real numbers broadens its application base. In such applications, each real number represents the intensity of a particular feature of the pattern, and facilitates its application to a number of problems requiring *autonomous learning* and *pattern analysis*, where data is encoded using real numbers. The switch from *Boolean vectors* to *real number vector patterns* can be performed at a small cost in terms of additional resources.

References

[1] Kanerva, P. (1988). *Sparse Distributed Memory*. The MIT Press, Cambridge, Massachusetts.

[2] Fernandez, B., Tsai, W.K., Parlos, A. (1990). *ASDM - A Novel Neural Network Model Based on Sparse Distributed Memory*. International Joint Conference on Neural Networks, Washington.

[3] Hely, T.A., Willshaw, D.J. and Hayes, G.M. (1997). A New Approach to Kanerva's Sparse Distributed Memory. IEEE Trans. on Neural Networks 8(3).

[4] Stuart, G., Spruston, N. and Häusser, M. (1999). (eds) *Dendrites*. Oxford University Press.

[5] Kong, A. (2001). *Intelligent Optimal Non-Linear Control*. Proc. Int. Conf. on Computational Intelligence Modelling and Control (CIMCA 2001), pp. 472-482, Las Vegas, Nevada.

[6] Kandel, E.R. (1976). *Cellular Basis of Behaviour*. pp. 179-184, W.H. Freeman and Co., San Francisco.

[7] Kandel, E.R. and Schwartz, J. (1986). *Principles of Neural Science*. pp. 119, Elsevier, Amsterdam.

[8] Shepherd, G.M. (1983). *Neurobiology*. pp. 150, Oxford University Press, UK.

[9] Kandel, E.R. (1976). *Cellular Basis of Behaviour*. pp. 304-310, W.H. Freeman and Co., San Francisco.

[10] Miles, F.A. (1969). *Excitable Cells*. pp. 89, Heinermann.

[11] Lasner, A. (1986). *Associative Processing in Brain Theory and Artificial Intelligence*. In Palm, G. and Aertser, A. (eds), Brain Theory, pp. 198, Springer-Verlag, Berlin.

[12] Freedman, D.J., Riesenhuber, M., Poggio, T. and Miller, E.K. (2001). *Categorical Representation of Visual Stimuli in the Primate Prefrontal Cortex*. Science, 291, pp. 312-316.

A Maximum S_i

For two synapses s_1 (excitory) and s_2 (inhibitory):

$$s_1 = \frac{k_1}{1 + e^{-(x_i - \theta_1)}} \tag{4}$$

where, θ_1 is the threshold of s_1, and k_1 is its gain and,

$$s_2 = \frac{-k_2}{1 + e^{-(x_i - \theta_2)}} \tag{5}$$

where, θ_2 is the threshold of s_2, and k_2 its gain.

The result of synaptic integration is given by,

$$S_i = s_1 + s_2 = \frac{k_1}{1 + e^{-(x_i - \theta_1)}} - \frac{k_2}{1 + e^{-(x_i - \theta_2)}} \tag{6}$$

At the maximum $dS_i/dx_i = 0$, i.e.,

$$\frac{dS_i}{dx_i} = \frac{k_1 e^{-(x_i - \theta_1)}}{\left[1 + e^{-(x_i - \theta_1)}\right]^2} - \frac{k_2 e^{-(x_i - \theta_2)}}{\left[1 + e^{-(x_i - \theta_2)}\right]^2} = 0 \tag{7}$$

$$\Rightarrow k_1 e^{-(x_i - \theta_1)} \left[1 + 2e^{-(x_i - \theta_2)} + e^{-2(x_i - \theta_2)}\right] - k_2 e^{-(x_i - \theta_2)} \left[1 + 2e^{-(x_i - \theta_1)} + e^{-2(x_i - \theta_1)}\right] = 0 \tag{8}$$

$$\Rightarrow k_1 e^{-(x_i - \theta_1)} + k_1 e^{-(x_i - \theta_1) - 2(x_i - \theta_2)} - k_2 e^{-(x_i - \theta_2)} - k_2 e^{-(x_i - \theta_2) - 2(x_i - \theta_1)} = 0 \tag{9}$$

$$\Rightarrow k_1 e^{\theta_1} e^{-x_i} + k_1 e^{\theta_1+2\theta_2} e^{-3x_i} - k_2 e^{\theta_2} e^{-x_i} - k_2 e^{\theta_2+2\theta_1} e^{-3x_i} = 0 \qquad (10)$$

$$\Rightarrow k_1 e^{\theta_1} + k_1 e^{\theta_1+2\theta_2} e^{-2x_i} - k_2 e^{\theta_2} - k_2 e^{\theta_2+2\theta_1} e^{-2x_i} = 0 \qquad (11)$$

$$\Rightarrow \left(k_1 e^{\theta_1+2\theta_2} - k_2 e^{\theta_2+2\theta_1}\right) e^{-2x_i} = k_2 e^{\theta_2} - k_1 e^{\theta_1} \qquad (12)$$

$$\Rightarrow e^{-2x_i} = \frac{k_2 e^{\theta_2} - k_1 e^{\theta_1}}{\left(k_1 e^{\theta_1+2\theta_2} - k_2 e^{\theta_2+2\theta_1}\right)} \qquad (13)$$

$$\Rightarrow e^{-x_i} = \sqrt{\frac{k_2 e^{\theta_2} - k_1 e^{\theta_1}}{\left(k_1 e^{\theta_1+2\theta_2} - k_2 e^{\theta_2+2\theta_1}\right)}} \qquad (14)$$

$$\Rightarrow x_i = -ln\sqrt{\frac{k_2 e^{\theta_2} - k_1 e^{\theta_1}}{\left(k_1 e^{\theta_1+2\theta_2} - k_2 e^{\theta_2+2\theta_1}\right)}} \qquad (15)$$