166166

NEURAL CORTEX

G00002

Field of Invention

The invention relates to an index-based neural network and to a method of processing information by pattern recognition using a neural network. It relates particularly but not exclusively to a neural network computer system which has an index-based weightless neural network with a columnar topography, and to a method whereby an input pattern is divided into a plurality of components and each component is processed according to a single pattern index to identify a recognised output pattern corresponding to the input pattern.

Background to the Invention

An artificial neural network is a structure composed of a number of interconnected units typically referred to as artificial neurons. Each unit has an input/output characteristic and implements a local computation or function. The output of any unit can be determined by its input/output characteristic and its interconnection with other units. Typically the unit input/output characteristics are relatively simple.

There are three major problems associated with artificial neural networks, namely: (a) scaling and hardware size practical limits; (b) network topology; and (c) training. The scaling and hardware size problem arises because there is a relationship between application complexity and artificial neural network size, such that scaling to accommodate a high resolution image may require hardware resources which exceed practical limits.

The network topology problem arises due to the fact that, although the overall function or functionality achieved is determined by the network topology, there are no clear rules or design guidelines for arbitrary application.

The training problem arises because training is difficult to accomplish.

The n-Tuple Classifier has been proposed in an attempt to address these problems. This classifier was the first suggested RAM-based neural network concept. The first hardware implementation of the n-Tuple Concept was the WISARD system developed at Brunel University around 1979 (see "Computer Vision Systems for Industry: Comparisons", appearing as Chapter 10 in "Computer Vision Systems for Industry", I Alexander, T Stonham and B Wilkie, 1982). The WISARD system belongs to the class of RAM-based weightless neural networks. This style of neural network addresses the problem of massive computations-based training by writing the data into a RAM-network and the problem of topology by suggesting a universal RAM-based network structure. However, the network topology of WISARD-type universal structure does not simulate the higher levels of neuronal organization found in biological neural networks. This leads to inefficient use of memory with the consequence that the problem of scaling still remains acute within RAM-based neural networks, and the application range of the WISARD-technology is limited.

Another example of neural networks that overcomes the problem of training by a simple memorization task is Sparse Distributed Memory (P Kanerva, 1998, "Sparse Distributed Memory", Cambridge, MA: NIT Press). However, a problem with the Sparse Distributed Memory, as with the WISARD system, is a large memory size. Another disadvantage of the Sparse Distributed Memory is its computational complexity. This is because for this type of memory, an input word must be compared to all memory locations.

N-Tuple classification systems use a method of recognition whereby an input to the neural network is divided into a number of components (n-Tuples) with each component compared to a series of component look-up tables. Normally there is an individual look-up table for each component. The network then processes each component in light of a large number of look-up tables to determine whether there has been a match. Where a match occurs for a component then that indicates that the component has been recognised. Recognition of each of the components of an input leads to recognition of the input.

The presence of a number of look-up tables results in a potentially large memory size. The memory size required is proportional to the number of components which the network may identify. This can result in a substantial increase in memory where the pattern size increases. For example, an artificial neural network might be designed for an image processing application, initially using an $n \times n$ image, where $n = 128$. This is a relatively low-resolution image by today's standards. Where the image to be processed increases from $n = 128$ to $n = 2048$ the number of neurons, the size of the network, increases by a factor of 256. This increase in memory results in the requirement for network expansion potentially requiring additional hardware modular blocks. Where the resolution of the image increases a point is quickly reached where the scaling to accommodate a high resolution image is beyond a practically achievable memory limit.

An object of the present invention is to address, overcome or alleviate some or all of the disadvantages present in the prior art.

Summary of the Invention

According to a first aspect of the invention, there is provided a neural network system including:

(a) a random access memory (RAM); and

(b) an index-based weightless neural network with a columnar topography;

wherein patterns of binary connections and values of output nodes' activities are stored in the RAM.

Preferably, the neural network system comprises a computer hardware component.

In a preferred form the neural network system has potential for scaling. Scaling may be achieved in any suitable manner. It is preferred that systematic expansion is achieved by increasing the size of the RAM.

The neural network may be trained in any suitable manner. It is preferred that the neural network is trained by writing of data into the RAM, and network topology emerges during the training.

It is preferred that performance of the neural network is adjustable by changing decomposition style of input data, and thereby changing dynamic range of input components.

It is preferred that input components to the neural network address a single common index.

According to a second aspect of the invention, there is provided a method of processing information by pattern recognition using a neural network including the steps of –

(a) storing a plurality of output patterns to be recognised in a pattern index;

(b) accepting an input pattern and dividing the input pattern into a plurality of components;

(c) processing each component according to the pattern index to identify a recognised output pattern corresponding to the input pattern.

Preferably each output pattern is divided into a plurality of recognised components with each recognised component being stored in the pattern index for recognition. The index preferably consists of columns with each column corresponding to one or more recognised components. Preferably the index is divided into a number of columns which is equal to or less than the number of recognised components. More preferably, the index is divided into a number of columns which is equal to the number of recognised components.

The method may further include the steps of each input component being compared to the corresponding recognised component column, and a score being allocated to one or more recognised components. Preferably the score for each recognised component of a pattern is added and the recognised pattern with the highest score is identified as the output pattern.

Brief Description of the Drawings

The invention will now be described in further detail by reference to the attached drawings which show example forms of the invention. It is to be understood that the specificity of the following description does not limit the generality of the foregoing disclosure.

Figure 1 is an index table illustrating processing of an input according to one embodiment of the invention.

Figure 2 is a schematic block diagram illustrating processing of an input according to an embodiment of the invention.

Figure 3 is a schematic block diagram illustrating processing of an output according to an embodiment of the invention.

Detailed Description

The invention can be implemented through the use of a neural card built with the use of standard digital chips. The invention is an index-based weightless neural network with a columnar topology that stores in RAM the patterns of binary connections and the values of the activities of output nodes. The network offers:

(a) Scaling potential: Systematic expansion of the neural network can be achieved not by adding extra modular building blocks as in previous artificial neural networks, but by increasing the RAM size to include additional columns or by increasing the height of the index. For example, 16 million connections can be implemented with a 64 MB RAM.

(b) The required memory size is reduced by a factor of N, when compared with previous n-Tuple systems such as the WISARD system, with N being the number of input components (n-Tuples). This is because the n-Tuple Classifier requires N look-up tables, whereas the present invention requires only one common storage.

(c) The network topology emerges automatically during the training.

(d) Training is reduced to writing of data into RAM.

(e) The performance can easily be adjusted by changing the dynamic range of input components, which can be achieved by changing the decomposition style of input data.

A device made according to the present invention is hereinafter referred to as a Neural Cortex. Both traditional artificial neural networks and traditional RAM-based artificial neural networks are networks of neuron-like computing units. However, the computing units of the human brain are multi-neuron cortical columns. A general view of the single common index on which the present invention is based can best be described as a collection of vertical columns, wherein the signals propagate in a bottom-to-top fashion.

Unlike traditional RAM-based neural networks, the Neural Cortex operates not by memorizing the names of classes in component look-up tables but by creating and memorizing an index (a linked data representation) of input components. This index contains the names of classes (class reference numbers) and is created on training.

On retrieval, the Neural Cortex, like the n-Tuple Classifier, sums up the names activated by input components. The summing operation provides the generalizing ability typical of neural networks. However, unlike the n-Tuple Classifier, where a "winner-takes-all" strategy is employed, the Neural Cortex employs a "winners-take-all" strategy. This is not a matter of preference but a necessity brought about by using a single common storage. In case of the n-Tuple Classifier, each input component (n-tuple) addresses its own look-up table. In case of the Neural Cortex, all input components address a single common index. This brings about a dramatic decrease in memory size. The absence of a single common index in both the n-Tuple Classifier and the Sparse Distributed Memory systems explains why previous RAM-based neural networks had difficulties in terms of memory requirements whose large size significantly limited the application range.

Further, a single common index is an efficient solution to the neural network expansion problem. As has been indicated above, both traditional artificial neural networks and traditional RAM-based artificial neural networks have scaling difficulties when the application size grows. For instance, if the image size grows from 128x128 pixels to 2048x2048 than a traditional artificial neural networks will need a 256-fold increase in memory because the number of n-tuples increases by the factor of 256 = 2048*2048/128*128. However paradoxically in the same situation, the Neural Cortex size according to the present invention may remain unchanged because still only one common index is used.

The present invention creates a single pattern index of input components. The index contains the output components and is created by storing the output pattern and training the neurons to recognise the pattern stored within the pattern index.

An output pattern S is decomposed into N number of components S_1, S_2, ..., S_N such that each component S_i is interpreted as the address of a column from the index. Each column stores the reference number of those patterns which have the value S_i in one or more of their components; each column does not contain more than one sample of each reference number. When an input I is received this is divided into a number of components I_1, I_2, ... ,I_X. Each input component I_1 to I_X is processed by the network by comparing the component with the pattern index. Where a component of the input I_i matches a component of the output S_i then each reference number listed in the column of S_i has a score of one added to its total. This process is repeated for each of the input components. The scores are then added to determine the winner. The winner is the reference number with the greatest score. The reference number, corresponding to a recognised output pattern, is recognised by the network.

An example of the pattern index is illustrated in Figure 1. This figure illustrates where the index has been trained or programmed to recognise three words "background", "variable" and "mouse". In this figure the words are assigned the reference numbers 1, 2 and 3 respectively. The output patterns are letters from

"a" to "z" with these included as columns within the index. When an input is received by the network each of the components of the input is processed by reference to this single pattern index. In this example the input is in the form of the word "mouse". This input is subsequently broken down into five letters. Each letter is processed in the network by using the index. The simultaneous nature of processing undertaken in the network can ensure that processing of each component is undertaken virtually simultaneously. The following processing is undertaken –

(a) the component of the input "m" is processed and in this case one point is added to the score attributable to variable 3;

(b) the component input "o" is processed and one point is added to variable 1 and 3;

(c) the component input "u" is processed and one point is attributable to variable 1 and 3;

(d) the component input "s" is processed and one point is attributable to variable 3;

(e) the component input "e" is processed and one point is attributable to variable 2 and 3.

The network then sums up the points attributable to each variable. In this instance variable 1 has a score of 2, variable 2 a score of 1 and variable 3 a score of 5. The variable with the highest score is determined to be the winner and hence identified. The variable 3 which has a score of 5, corresponding to the word "mouse", is therefore considered to be identified.

In case of standard RAM, two different address words always point towards two different memory locations. This is no longer true in case of the Neural Cortex. For example, if the input pattern has three components (a, b, c) and the component dynamic range is 1 byte then the patterns (a,c,b), (b,a,c), (b,c,a), (c,a,b), (c,b,a) will produce the same score equal to 3 because the Neural Cortex is invariant with respect to permutations. The invariance is caused by the fact that all components (n-tuples) address a single common storage. The common storage collapses the N-dimensional space into a one-dimensional space thus creating the permutational invariance, which is the price to be paid

for dramatic reduction in memory size as compared to traditional RAM-based neural networks. This invariance is the key to the Neural Cortex. At the same time, it is the beauty of the approach because this invariance becomes practically harmless when the component dynamic range is increased. For the above example, by using the 2 bytes dynamic range, where the pattern (a,b,c) is converted into the 2 component pattern (ab, bc), the following scores will be obtained: (a,b,c) =>2, (a,c,b)=>0, (b,a,c)=>0, (b,c,a)=>1, (c,a,b)=>1, (c,b,a)=>0, so that the pattern (a,b,c) will be identified correctly.

In general case, the conversion of the n-component input pattern (s_1, s_2, ... , s_N) into a new pattern (c_1, c_2, ... , c_M) whose components have a greater dynamic range and M < N is preferably done by the software driver of the Neural Cortex card.

This conversion can be referred to as the C(haos)-transform, if it converts the sequence of all input patterns into a one-dimensional chaotic iterated map. The sufficient condition for the absence of identification ambiguity is that the sequence of all C-transformed input patterns is a chaotic iterated map. This is true because in this case all pattern components will be different, which leaves no room for identification ambiguity. In fact, this condition is too strong because it is sufficient that any two patterns differ in one component, at least. For practical purposes a good approximation of the C-transform can be achieved by increasing components' dynamic range to 2 bytes, 3 bytes, etc. when 2, 3 or more components are joined together, e.g., (a,b,c) is converted into the 2 component pattern (ab, bc).

A Neural Cortex read-cycle block-diagram is shown in Figure 2. The blocks 'Roots', 'Links', 'Names', 'Score' are RAM-devices. Σ is a summer. T-logic is a terminating logical device.

1. Each pattern component (A-Word) is passed to the address bus of the 'Roots' RAM.
2. The output value R of the 'Roots' RAM is passed to the address bus of the 'Links' RAM.

3. The output value L of the 'Links' RAM is passed to the address bus of the 'Names' RAM.

4. And, finally, the output value N of the 'Names' RAM is passed to the address bus of the 'Score' RAM.

If L is 0 then the T-logic terminates the process. Otherwise, the 'Score' RAM content found at address N that is determined by the output of the 'Name' RAM is incremented by the value of 1. Next, the 'Links' RAM output is fed back to the 'Links' RAM address bus. The process repeats itself from point 3.

A Neural Cortex write-cycle block-diagram is shown in Figure 3. The blocks 'Roots', 'Links', 'Names', are RAM-devices. CU is the control unit.

1. Each pattern component A is passed to the address bus of the 'Roots' RAM.

2. The output value R of the 'Roots' RAM is passed to the address-bus of the 'Links' RAM.

3. The output value L of the 'Links' RAM is passed to the address-bus of the 'Names' RAM. The output value of the 'Names' RAM is denoted by N, and the current pattern name by P.

4. The values R, L, N and P are passed to the control unit, which utilizes the following logic. If L is 0 then the control unit makes a decision (point 5) on updating 'Roots', 'Links' and 'Names' RAM. Otherwise, L is fed back to the 'Links' RAM address bus. The process repeats itself from point 3.

5. Decision Logic:

a) if N = P, terminate the process;

if R = 0, increment the counter value C by 1,

write C to 'Roots' RAM at address A,

write C to 'Links' RAM at address R,

write P to 'Names' RAM at address L,

if R > 0 & L=0, increment the counter value C by 1,

write C to 'Links' RAM at address R,

write P to 'Names' RAM at address L,

b) terminate the process.

Performance of the Neural Cortex can be adjusted in terms of memory size and read/write times. Normally, storage and recall times increase when the number of classes grows as the training continues. Additional classes increase the amount of reference numbers that are stored in index columns and, therefore, the amount of index cells that have to be accessed. As a remedy, one can increase the dynamic range ***D*** of input pattern components. This increases the number of index columns because the index address space is equal to ***D***. As a result, the same amount of reference numbers will be spread upon the greater area, which, in turn, decreases the average index height ***H***.

The processing time on storage and recall is proportional to the number of accessed memory cells, which is proportional to ***HN***. Here, ***N*** is the number of the pattern components. As ***D*** increases, the processing time approaches ***O(N)***. This follows from the fact that ***H*** is inverse proportional to ***D***.

The memory size is proportional to ***HD***. However, ***H*** grows/decreases faster than ***D***. Hence, adjusting the dynamic range ***D*** can efficiently control the memory size. In the worst case, the Neural Cortex size does not exceed ***CD***, where ***C*** is the number of pattern classes, which is because the Neural Cortex has only one "look-up-table". On the other hand, the memory size of a traditional RAM-based artificial neural network is ***CDN*** because for this type of artificial neural network the number of input look-up-tables is equal to the number ***N*** of input pattern components.

It is to be understood that various modifications, alterations and/or additions may be made to the parts previously described without departing from the ambit of the present invention.

The claims defining the invention are as follows :

1. A neural network system including:

(a) a random access memory (RAM); and

(b) an index-based weightless neural network with a columnar topography;

wherein patterns of binary connections and values of output nodes' activities are stored in the RAM.

2. A neural network system according to claim 1 wherein the system comprises a computer hardware component.

3. A neural network system according to claim 1 or claim 2 wherein systematic expansion is achieved by increasing the size of the RAM.

4. A neural network system according to any one of claims 1 to 3 wherein the neural network is trained by writing of data into the RAM, and network topology emerges during the training.

5. A neural network system according to any one of claims 1 to 4 wherein performance is adjustable by changing decomposition style of input data, and thereby changing dynamic range of input components.

6. A neural network system according to any one of claims 1 to 5 wherein input components address a single common index.

7. A method of processing information by pattern recognition using a neural network including the steps of –

(a) storing a plurality of output patterns to be recognised in a pattern index;

(b) accepting an input pattern and dividing the input pattern into a plurality of components;

(c) processing each component according to the pattern index to identify a recognised output pattern corresponding to the input pattern.

8. A method according to claim 7 wherein each output pattern is divided into a plurality of recognised components with each recognised component being stored in the pattern index for recognition.

9. A method according to claim 8 wherein the index consists of columns with each column corresponding to one or more recognised components.

10. A method according to claim 9 wherein the index is divided into a number of columns which is equal to or less than the number of recognised components.

11. A method according to claim 9 wherein the index is divided into a number of columns which is equal to the number of recognised components.

12. A method according to any one of claims 8 to 10 wherein each input component is compared to the corresponding recognised component column and a score is allocated to one or more recognised components.

13. A method according to claim 12 wherein the score for each recognised component of a pattern is added and the recognised pattern with the highest score is identified as the output pattern.