A SELF ORGANIZED CLASSIFIER BASED ON MAXIMUM INFORMATION INDEX AND ITS DEVELOPMENT USING VHDL

Janusz Starzyk *Yongtao Guo*
School Of Electrical Engineering and Computer Science
Ohio University
Athens, OH 45701, U.S.A.
{Starzyk,gyt}@bobcat.ent.ohiou.edu

ABSTRACT

An entropy-based self organized learning algorithm is presented for classifier design. The method, which is based on ideas from information theory, maximizes the information index during feed forward learning in ANN. The neuron in the architecture divides the subspace based on the division of previous neurons which are connected to the current neuron by some probability distribution. Utilizing the entropy-based evaluator (EBE), the neuron further divides current subspace via optimal selection of the subspace control neuron, feeding features and fitting functions. Our method is simulated on standard benchmark examples and achieves performance improvement over many traditional classifier algorithm. The hardware development in behavioral level simulation using VHDL gains satisfactory results and provides significant reference to FPGA synthesis in structural level.

KEYWORDS: Self Organized, ANN, Entropy, VHDL and FPGA

1 INTRODUCTION

Artificial neural networks (ANNs) derived from the field of neuroscience display interesting features such as parallelism, adaptation and classification. Data processing with ANN mimics the parallel, adaptive, neural structures of the brain for reproducing some of its capabilities and has wide research and application values. Classifier design is one of the important research and application field in ANN. To implement classifier, there are many methods including learning machines, neural network and statistical methods. Our proposed ANN classifier method based on information theory demonstrates to perform well in comparison with many conventional classifier method.

The proposed self organized learning algorithm is derived from both neural networks and information theory. In information theory, entropy maximization problem is seen to be equivalent to a free energy minimization in a closed system, motivating our entropy based approach to minimize the misclassification cost. Based on our previously proposed method [1][2], in this paper, we will future explore and explain our method from our current development.

When we design and simulate the algorithm, we already consider the implementation of the algorithm. Custom hardware requires significant non-recurring engineering (NRE) cost which makes ANN difficult to implement. FPGAs are readily available, reconfigurable and have smaller NRE costs – all important advantages for ANNs applications. VHDL provides a convenient construct for the implementation of ANN into FPGA. This paper also presents our development approach for hardware implementation of the ANN classifier. Currently the EBE module has been developed and presented at RTL level [2]. In this paper, we will introduce the whole system development using VHDL at behavioral level .

In section 2, the self organized learning algorithm development is introduced. Section 3 deals with the algorithm simulation using both MATLAB and VHDL. Finally, a conclusion is given in Section 4.

2 ALGORITHM

Let $S = \{(\boldsymbol{n}, c)\}$ be a training set of N vectors, where $v \in \Re^n$ is a feature vector and $c \in Z$ is its class label from an index set Z. A classifier is a mapping $C : \Re^n \to Z$, which assigns a class label in Z to each vector in $\Re^n$. A training pair $(\boldsymbol{n}, c) \in S$ is misclassified if $C(\boldsymbol{n}) \neq c$. The performance measure of the classifier is the probability of error, i.e. the fraction of the training set that it misclassifies. Our goal is to minimize this cost and it is achieved through simple threshold searching based on maximum information index, which is calculated from estimated subspace probability and local class probabilities. In the latter section of the paper, we will list the comparison of the probability of error between a number of existing classifier algorithm with our self organized approach. Here we estimate the probability distribution of

the training data and calculate the system entropy. Then we use entropy based information index to built the neural network structure in the learning process. The entropy calculation is performed in parallel in different neurons to speed up the learning process. Several EBE units are used to accomplish this learning and self-organizing task. The number of EBE depends on the design area requirement and the number of neurons per a single layer. The learning idea is derived from the notion of a "fitness" functional, suitably chosen to represent the clustering problem. Mathematically the fitness functional class has the form f(S; C, T) where S is a coving of training set T and C is the class distribution of T. Since f must correlate to how well a function can emerge from the network (the resulting clustering function), f depends on the netwo rk organization and the goal is thus to optimize f for all possible coverings S that can be represented by a given organization. For any covering S represented by the given organization (i.e., an optimal classification), the function f must be chosen such that its value can be computed with little overhead (such as in polynomial time with respect to input size), this way we can verify in a tractable manner whether this value is within an acceptable range of values that correspond to "good" solutions, normalized with respect to the given organization. Since it is impracticable to examine all possible coverings, the approach taken is that each computing fitness function for each computing element is chosen in the statistical sense: f(S';C',T') = H(C':S';T')/ H(C';T') where H(C':S';T') is the mutual entropy of the local distribution of classes C' among the chosen set T' (subset of T) of samples of local features and the distribution of clusters S' among these sample features; H(C';T') is the entropy of the distribution of classes among the chosen samples. The clustering is thus solved cooperatively by all the resulting active neurons and f(S;C,T) reflects the cooperative nature of the algorithm. So the algorithm is self organized dynamic learning process.

In the algorithm implementation of the self-organizing learning model, the ANN is a feed forward structure. It has a prewired organization that contains a number of identical processing blocks (called neurons), which are pseudo-randomly connected to the input nodes and other neurons. In addition, EBEs are utilized to select a proper operation representing the different fitness function and input selection for each neuron. In the learning process, the set of training signals is searched sequentially class by class to establish the optimum point (threshold) which best separates the signals of the various training classes. The quality of the partition is measured by the entropy based information index defined as follows:

$$I = 1 - \Delta E / E_{\max} \quad \text{where}$$

$$\Delta E = -\sum_{s}\sum_{c} P_{sc} \log(P_{sc}) + \sum_{s} P_{s} \log(P_{s})$$

and $E_{\max} = -\sum_{c} P_{c} \log(P_{c})$

here P_c, P_s, P_{sc} represent the probabilities of each class, attribute probability and joint probability respectively. The summation is performed with respect to all classes and subspaces. The information index should be maximized to provide an optimum separation of the input training data. The competition among neurons is realized by maximizing information content in neurons. When the limit value of the information index equals to 1, the problem at hand is solved completely, i.e. the training data is correctly classify by the ANN. The training objective is to find a threshold value which maximizes normalized information index I in the separate neurons. The threshold value searching is also subject to current confidence value. If acceptable confidence value is achieved and the current neuron reaches some information index threshold, the neuron will be counted as voting neuron. Several voting neurons are weighted together to solve a given problem.

3 SIMULATION

A Algorithm Simulation

The EBE algorithm is first verified by Matlab simulation in both behavioral and structural level. The simulation result obtained in structural levels is shown in Fig. 1.

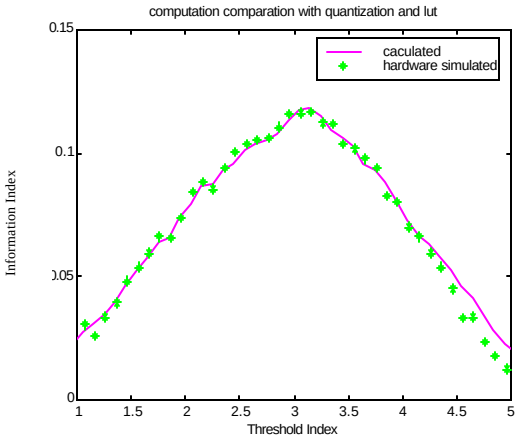

Fig.1 Structural Simulation

In this example, two one thousand, one-dimensional and normally-distributed points with different mean values and variances are taken as training data. In the behavioral simulation, we use 8bit widths to represent the input analog data and set threshold searching step to be maximum quantification noise. In order to simplify hardware used for EBE, an effect of round off errors on the

accuracy of the resulting information index is considered. The first test performed is to determine the dependence of the information index error on the number of bits used in hardware implementation. For simple verification, we use two classes training data with three dimensions per class. In every dimension, there are one-thousand normally distributed random values with different mean value and variance.

Using the above EBE module, our system simulation which implement our self organized learning algorithm is based on a typical credit card problem which has two classes to recognize. Several classification algorithm is tested on this benchmark [3] including some traditional learning machines, neural network and statistical methods. Our method is in the last two item in the table 1 [4].

Method	Miss Detection Probability	Method	Miss Detection Probablity
CAL5	.131	Naivebay	.151
DIPOL92	.141	CASTLE	.148
Logdisc	.141	ALLOC80	.201
SMART	.158	CART	.145
C4.5	.155	NewID	.181
IndCART	.152	CN2	.204
Bprop	.154	LVQ	.197
Discrim	.141	Kohenen	-
RBF	.145	Quadisc	.207
Baytree	.171	Default	.440
ITule	.137		
AC2	.181	SOLAR* (single)	.183
k-NN	.181	SOLAR* (ensemble)	.135

* SOLAR is our Self-Organizing Learning Array System[4]

Table 1 Miss Rate for Algorithm

B. VHDL Simulation

The use of VHDL has a number of advantages over conventional hardware design techniques like schematic capture. Its high level syntax is not very different from conventional imperative programming languages, thus the design effort is not significantly different from writing a software simulation of a ANN. And VHDL supports extensive optimizations. We use VHDL to describe a digital system at the behavioral level so that we can simulate the system to check out the algorithm used and to make sure that the sequences of operations are correct. Fig.2 is the EBE module RTL level simulation, the simulation match the Matlab simulation results. We add the PCI bus interface modules into the design and organize them as a hierarchical structure. The first level in hierarchy is the PCI interface which includes DMA, FIFO,etc. The *Plog(P)* function is implemented by the ROM LUT with 5-bit width address. Other units adopt 8-bit data flow. The outer modules like PCI interface, FIFO and so on are linked to EBE module by 32-bit data bus. In the process of simulation, we use the lowest 10 bits as the data channel *I* from Class *I* and the upper 10 bits as the data channel *II* from Class *II*. From the simulation results, we can see that the data are transferred and controlled by the signals -Request, Start, Done, OE and current state. The interface and control signals are transferred between EBE interface and PCI and between Control unit and other modules. After the current threshold reaches the maximum threshold set by the generic parameter, the Done signal will be set to High and the simulation process will be over. The last output threshold will be the classification threshold for the current dimension of the classes. In the EBE calculating process, the input data will be resended each time the threshold is updated by the generic threshold step parameter while the signal Request to the interface model will be set to low. The Request, Done, Start, OE are also used as the handshake signal groups for the synchronized work of the whole hardware module. The simulation results are obtained by using Aldec-HDL [5] simulator. Most of units including PCI bus interface are synthesized using logic synthesis tool, Leonardo yielding the gate level structure of the full EBE model.

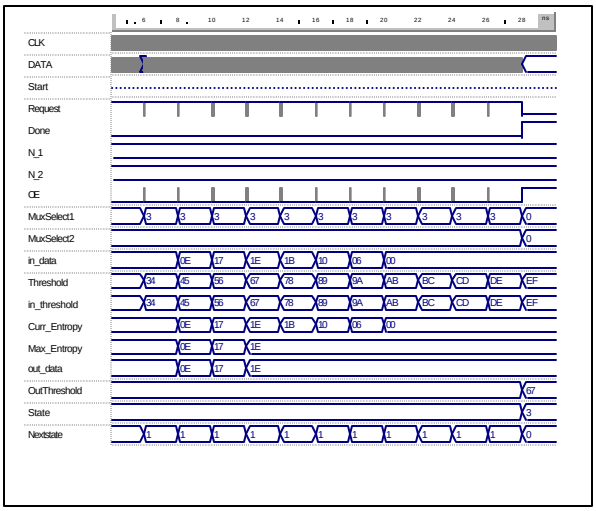

Fig. 2 EBE VHDL Simulation at RTL

The whole system simulation using VHDL is at behavioral level to verify the system adaptability in FPGA hardware. The VHDL system simulation hierarchy architecture in behavioral level is shown in Fig.3.

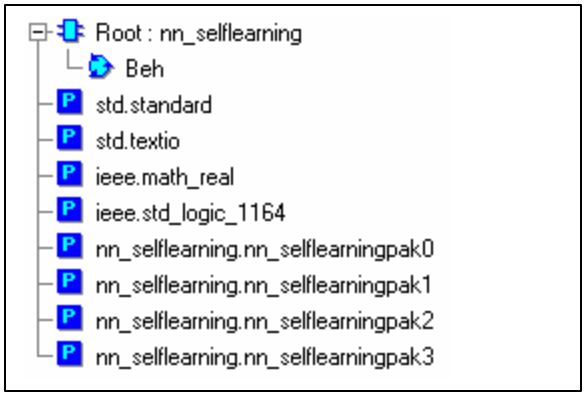

Fig.3 System Hierarchy Architecture

The topmost design is the description for system input and output, the initialization and the update of the memory element in the network including reading the training data $S = \{(\boldsymbol{n}, c)\}$ and writing the learning results. The initialization is used to build up the initial 2D neural network mesh architecture including neurons' connections and fitting functions. The fitting functions are treated as low-level, canonical neural functions. Ideally they should satisfy closure and completeness. Completeness guarantees that the set is sufficient to implement any function; Closure guarantees there's no outlier function that cannot be uniformly approximately by a series expansion based on the kernel set. Fig.4 gives a learn result of one neuron in the mesh using the VHDL simulated data including the original training data. For the neuron ID 5, the selected input subspace is the entire space for the raw input data, the neuron ID 5 uses the composite fitting function to evolve the learning space shown as the below part in Fig.4.

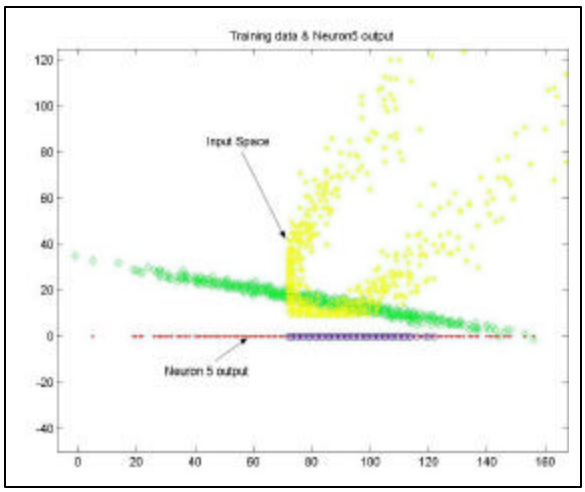

Fig.4 Training data & Neuron ID5 Output Subspace

The set of fitting functions on the original data feed into the current neuron and the neuron output indicates that the neuron has gained enough knowledge to resolve the classification problem in the given subspace. In the 2D lattice, we use Manhatan distance as a distance measure for any given pair of neurons. There is only one nearest neighbor from pervious layer, at most three available next nearest neighbors and all neurons with distance greater than two are considered remote. The learning process is a supervised learning procedure assuming that the class information is given. Subspace learning is based on local optimization of mutual entropy between the class and the subspace, generally the subspace learning is to explore the best local transformation which is restricted to a given set of kernel functions, their linear combinations and composites. So the classes are optimally separated into subspaces. Once certain neuron's information index reaches maximum, the related information is updated in neuron's memory. At the same time, if the calculated information index reaches some generic maximum value, not much information can be gained for further dividing the selected subspace and the neuron will be labeled as "voting" neuron to cooperate with other voting neurons in testing stage when the learning is over. All these functions are given in hierarchical organization. Package 0 gives the necessary initiation function, kernel function and other calculation function. Package 1 implements EBE calculation based on current threshold. Package 2 calls EBE calculation to seek optimum threshold to reach maximum information index. Package 3 carried out the self organized learning to update the neurons' information memory. Fig.5 gives the final learned self organized architecture based on VHDL system simulated data using small number of neurons.

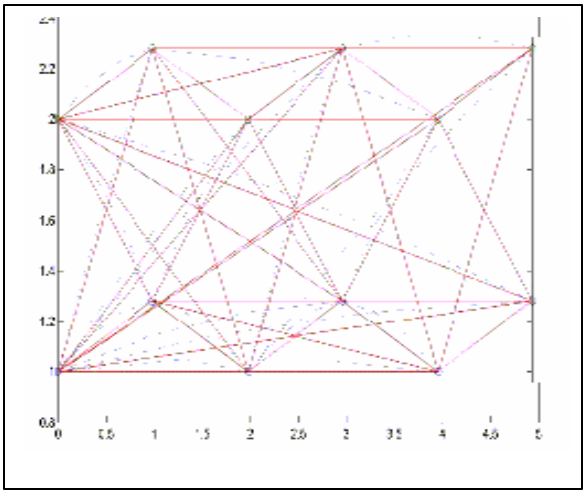

Fig.5 Learned Architecture

4 CONCLUSION

In the paper, we presented a self organized ANN classifier design based on information entropy. The algorithm based on ideas from information theory maximizes the information index during feed forward learning in ANN. So the neuron networks will be self organized connected and the processing function is searched to achieve optimal division of the current subspace. Once the information index reaches some certain threshold, the neuron will be labeled as voting neuron. The voting neurons vote to classify the input data. The developed models have been verified by both Matlab and VHDL simulation results. We used the VHDL simulation to verify the hardware organization and operation at structural level and whether the selected bit widths for internal and external signals are sufficient for achieving a required computation precision under certain confidence values. In classification area, the necessary calculation accuracy varies by application. The low precision can simplify the hardware implementation complexity and speedup the performance. The EBE module, consisting of calculating unit, a memory unit and a few digital components, has been modeled and simulated in VHDL at structural level. Experimental results show that the obtained classification of the training data by system behavioral VHDL simulation matches closely with that anticipated from the analysis results. Our next objective is to construct a parallel self organized neural networks classifier using Virtex FPGA [6] from Xilinx Corp, verify hardware training phase and test the approach in the real world applications.

In our proposed method, although it achieves good performance over many traditional classifier algorithm, we still need consider other aspects including both information theory and neural networks itself. For instance, one of the possible consideration is more efficient cooperative and competitive information control which can mediate between cooperation and competition during neuron's learning. On the other hand, the hardware implementation on FPGA is limited by the reconfigurable hardware resource, especially the connection between neurons. Similar to the currently implemented ANN chips, our ANN architecture inevitably has high interconnection requirement including long line to connect to remote control neurons which is very limited resources inside FPGA. So except the consideration of our algorithm, the hardware architecture is also our future work.

5 REFERENCES

[1] J. A. Starzyk and J.Pang, "Evolvable binary artificial neural network for data classification.", The 2000 Int. Conf. on Parallel and Distributed Processing Techniques and Applications, (Las Vegas, NV, June 2000).

[2] J.A Starzyk and Y. Guo, "Entropy-Based Self-Organized ANN Design Using Virtex FPGA", the Int. Conf. on Engineering of Reconfigurable Systems and Agorithms, (Las Vegas, NV, June 2001).

[3] D.Michie, D.J.Spiegelhalter, and C.C.Taylor, "Machine learning, Neural and Statistical Classification" London, U.K. Ellis Horwood Ltd.1994.

[4] J. A. Starzyk and Zhen Zhu, "Software Simulation of a Self-Organizing Learning Array System". Presented to Int. Conf.,(Canada,2002).

[5] Aldec, "Aldec-HDL™ Series User Guide Version 4.1", August 2000.

[6] Xilinx, "The Programmable Logic Data Book", 1993.