Int. J. Advance. Soft Comput. Appl., Vol. 2, No. 2, July 2010
ISSN 2074-8523; Copyright © ICSRS Publication, 2010
www.i-csrs.org

Hybrid Learning Algorithm in Neural Network System for Enzyme Classification

Mohd Haniff Osman, Choong-Yeun Liong, and Ishak Hashim

Faculty of Engineering and Built Environment,
Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia
Email: haniff68@ukm.my

School of Mathematical Sciences, Faculty of Science and Technology,
Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia,
Email:lg@ukm.my,

School of Mathematical Sciences, Faculty of Science and Technology,
Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia,
Email:ishak_h@ukm.my

Abstract

Nucleic acid and protein sequences store a wealth of information which ultimately determines their functions and characteristics. Protein sequences classification deals with the assignment of sequences to known categories based on homology detection properties. In this paper, we developed a hybrid learning algorithm in neural network system called Neural Network Enzyme Classification (NNEC) to classify an enzyme found in Protein Data Bank (PDB) to a given family of enzymes. NNEC was developed based on Multilayer Perceptron with hybrid learning algorithm combining the genetic algorithm (GA) and Backpropagation (BP), where one of them acts as an operator in the other. Here, BP is used as a mutation-like-operator of the general GA search template. The proposed hybrid model was tested with different topologies of network architecture, especially in determining the number of hidden nodes. The precision results are quite promising in classifying the enzyme accordingly.

Keywords: *enzyme; protein classification; neural networks; hybrid GA-BP*

1. Introduction

Protein sequence classification is one of the challenging and crucial problems of computational biology. The problem is to determine whether or not an unknown protein sequence belongs to a known set of class or family. If a new protein sequence belongs to a given class, it is presumed that it shares similar functions and structural characteristics. In several studies, protein classification problem has been examined at various levels, according to a top hierarchy in molecular taxonomy, consisting of superfamilies, families and subfamilies [1].

There are several approaches have been developed for solving protein classification. Most of them are based on appropriately modeling proteins families, either directly or indirectly [2]. Direct modeling technique means that by using a set of sequences namely training pattern, a model that characterizes the family of interest is built. The tool named HMMer using Hidden Markov models (HMM) employs a machine learning algorithm based on probabilistic graphical models to describe time-series and sequence data [3]. HMM is a generalization of the position-specific scoring matrix to include insertion and deletion states. HMM aligns an unknown sequence, Q to a given family if the scoring point is more significant than a cut-off value. Indirectly the techniques use a preprocessing tool to extract significant feature from sequences. In this way, sequences of variable length are transformed into fixed-length input vectors that are subsequently used for training discriminative models, such as neural networks [3].

Enzymes are proteins that catalyze (i.e. accelerate) chemical reactions [14]. There are generally globular proteins which play a big role in determining the next steps that will occur in metabolic pathway inside living organisms. Without enzymes, metabolism would neither progress through the same steps, nor be fast enough to serve the needs of the cell.

Artificial neural networks (ANNs) are a neurobiological inspired paradigm that emulates the functioning of the brain, based on the way we believe that neurons work, because they are recognized as the cellular elements responsible for the brain information processing [4]. In more practical terms, neural networks are non-linear statistical data modeling tools. ANNs are particularly suitable to solve problem in such applications like time series prediction, pattern and sequence recognition, etc. The ANNs have shown to be a powerful tool in many wide areas, but they have a big problem: their reasoning process cannot be explained, that is no clear relationship between inputs presented to the network and the outputs it returns [5].

Genetic algorithm (GA) is a parallel stochastic search technique used in computing to find an exact or approximate solution to an optimization and search problems. Formally introduced by John Holland at University of Michigan in 1970, GA works very well on mixed (continuous and discrete), combinatorial problems. GA creates a population that contains a number of possible solutions to a given problem called individual and applies genetic operator such as mutation and crossover to evolve the solutions in order to find the best one.

In this work, we develop an enzyme classification system based on a hybrid genetic algorithm and BP methodology in neural networks. We shall call the system NNEC which stands for Neural Network Enzyme Classification. Some testings were undertaken to determine the performance of the system.

2. Sequence encoding

There are many protein databases available in open source websites, such as Protein Information Resources (PIR), SCOP, Swiss Prot, Protein Data Bank (PDB). In our work, the training-testing datasets are collected from PDB (http://www.rcsb.org/pdb). The contents of this database come from X-ray crystallography and nuclear magnetic resonance imaging. Figure 1 represents enzyme coded by 77 amino acids named DNA TOPOISOMERASE I which belongs to Isomerases family.

MRALFYKDGKLFTDNNFLNPVSDDNPAYEVLQHVKIPTHLTDVVVYEQTWEEALTRLIFVGSDSKGRRQYFYGKMHV

Fig. 1 Primary structure of the enzyme DNA TOPOISOMERASE I

For protein classification, two types of extracting feature from sequence are conducted: one is related to the global structure, and the other is related to the local similarity of sequence. Global feature is usually made by using 2-gram encoding scheme that count occurrence of two consecutive amino acids in protein sequence [3]. A more recent approach [6], is a scheme of globally encoding the sequence, where each amino acids character is initially represented as a unique binary number with n bits ($n = 5$ for 20 amino acids) and then each sequence is mapped into a position inside the n-dimensional hypercube.

Enzymes are composed by a variable number of amino acids. In this section, we focus on encoding directly primary structure of protein, in string of letters forms into a numerical vector that appropriate for NNs. The main idea of encoding procedure is by using Kyte and Doolittle hydrophobicity scale as seen in Table 1 to convert a string of amino acids symbol into real-valued vector. For instance, suppose that we have a sequence in form of ‘MRALF’. After applying the scale, we got a vector of size of 1 x 5 containing the value of [1.9 -4.0 1.8 3.8 2.8].

3. Neural network system

In this section, we defined the step involved in the construction of the neural network system motivated by Weinert et al. [13]. Typically, users only apply one single network when solving problem using ANN approach. But, in this work, we have a set of network to produce the results. We consider that there can be z different classes or targets, and each target can be r patterns (number of enzymes). The length of a given sequence $Q_{i,j}$ is defined as $\ell_{i,j}$

$(i = 1 \ldots z, j = 1 \ldots r)$. The reality is that the length of the sequences for training set is not similar. The number of network in system is defined by p, where the value of p is determined as:

$$p = \frac{max_{i=1\ldots z, j=1\ldots r}(t_{i,j})}{t} \quad , \tag{1}$$

where t is number of amino acids into which the sequences will be broken into. The value t also represents the number of nodes in the input layer of network. The result of equation (1) is rounded down to the nearest integer. The system ignores any sequences whose length is smaller than t. In order to choose the value for t, no specific method was stated. If users choose small number of t, then a number of networks to be trained will be large; hence causes network's complexity in this work. But if the chosen value of t is too big, then the network will face the curse of dimensionality scenario. Curse of dimensionality [15] refers to the exponential growth of hyper volume as a function of dimensionality. The curse of dimensionality causes networks with lots of irrelevant inputs to be behave relatively badly; the dimension of the input space is high, and the network uses almost all its resources to represent redundant portions of the space.

Table 1. Kyte and Doolittle (K & D) hydrophobicity scale

Amino Acid		K & D scale	Type
name	symbol		
Isoleucine	I	+4.5	Hydrophobic
Valine	V	+4.2	Hydrophobic
Leucine	L	+3.8	Hydrophobic
Phenylalanine	F	+2.8	Hydrophobic
Cysteine	C	+2.5	Hydrophobic
Methionine	M	+1.9	Hydrophobic
Alanine	A	+1.8	Hydrophobic
Glycine	G	-0.4	Natural
Threonine	T	-0.7	Natural
Serine	S	-0.8	Natural
Tryptophan	W	-0.9	Natural
Tyrosine	Y	-1.3	Natural
Proline	P	-1.6	Natural
Histidine	H	-3.2	Hydrophilic
Giutamine	Q	-3.5	Hydrophilic
Asparagine	N	-3.5	Hydrophilic
Glumatic acid	E	-3.5	Hydrophilic
Aspartic acid	D	-3.5	Hydrophilic
Lysine	K	-3.9	Hydrophilic
Arginine	R	-4.0	Hydrophilic

So, based on distribution of sequence's length in our dataset, the good number for t are 40. Every ANN will have different number of patterns and each of them will process data from the corresponding partition. So the first block of the

sequence will be fed into the first network, the second block by the second network, and so on. The architecture of the ANN is fixed and defined a priori as three layer back propagation multilayer perceptron (MLP). The topology of the network is represented by the ratio of the three variables $I{:}H{:}D$. The first variable is the number of nodes in the input layer, here, t. The second variable is number of nodes in hidden layer and the last value is number of nodes in output layer, which represents the number of enzyme's classes.

When applying BP neural network, determining the optimal number of hidden nodes has always been a question. Determining the number of hidden nodes to be used resembles the main bottle neck of computation leading to better performance. In most application, the numbers of hidden nodes are determined on intuitive bases. For this study, we will consider two approaches. Wanas et al. [7] proposed that optimal number of hidden nodes is $\log_2 T$ where T is number training patterns and we denoted the training procedure as profile 1. For profile 2, we apply approach by Baum and Haussler [8], where if net expected to classify $(1 - e/2)$ of the training pattern correctly and classify $(1 - e)$ of testing pattern correctly, enough training pattern is determined by the condition

$$\frac{W}{T} = e,$$

where T is the number of training patterns, W is the number of weights in network and e is the accuracy of classification expected. So, the number of hidden nodes, H can be describe as

$$W = (H * I) + (D * H),$$

$$H = \frac{W}{(I + D)},$$

where I is nodes in the input layer and D is nodes in the output layer. Different numbers of training patterns participate in each network contribute differently to the final classification. A higher number of training patterns in ANN will merit more confidence to the system. Therefore, a classification weight (CW) is defined as a function of the number of training cases in each network. The CW for each network is simplified as

$$CW_{(i)} = \frac{T_{(i)}}{T_{(1)}},$$

where $i = 1 \ldots p$. For example, let say we have two classes of targets $(z = 2)$, each class with 4 enzymes $(r = 4)$. Let say $Q_{2,2}$ is the longest sequence with 260 amino acids. By using t =40, equation (1) gives the number of NN to be used of 6. Therefore, the neural system for this example is composed of 6 ANNs. As shown in

Figure 2, the first ANN $(p = 1)$ would be trained with 8 protein segments; the second NN with 7; the third NN with 6 and so on. In the last column, the classification weights are printed.

		Class 1				Class 2				Total	CW
		Sequence ($m = 4$)				Sequence ($m = 4$)					
		1	2	3	4	1	2	3	4		
NETWORK	1	■	■	■	■	■	■	■	■	8	1.0
	2	■	■	■	■		■	■	■	7	0.875
	3	■	■		■		■	■	■	6	0.75
	4		■		■		■		■	4	0.5
	5		■		■		■			3	0.375
	6						■			1	0.125

Fig. 2 Example of how training set for each network and classification weights, CW are obtained

Suppose that the sequence $Q_{(i,j)}$ is belonged to class 1, it should have the output vector (1, 0) for each NN involved. Conversely if the sequence belongs to other class, the output should be (0, 1). Output vector also can be in bipolar form, (1,-1) or (-1, 1).

4. The Classifier Module

For every testing sequence, they are codified in the same way as the training sequence. Then, the sequence was separated into block of t amino acids. The first block was submitted to the first ANN, the second block to second ANN, and so on. If, for a given sequence, the length of the sequence was longer than $p.t\left(Q_{i,j} > p.t\right)$, then all amino acids after $p.t$-th position were ignored. The output vector for all networks was the input to the classifier module. The set of all network's output was considered to be a single, $z\ x\ p$ matrix, called $\boldsymbol{N}$, where column i represented the output vector of i-th network, and row j refers the network target. And the set of classification weight were in form $p\ x\ 1$ vector. The final classification performed by the classifier module is given by equation (2), and is the maximum value of the product between $\boldsymbol{N}$ and $\boldsymbol{CW}$;

$$\boldsymbol{N.CW} = \begin{pmatrix} n_{1,1} & n_{1,2} & \cdots & n_{1,p} \\ n_{2,1} & n_{2,2} & \cdots & n_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ n_{z,1} & n_{z,2} & \cdots & n_{z,p} \end{pmatrix} \begin{pmatrix} cw_1 \\ cw_2 \\ \vdots \\ cw_p \end{pmatrix}.$$

$$class = max(\boldsymbol{N.CW}) \qquad (2)$$

5. Hybrid GA-BP

Genetic algorithm is algorithm for optimization based on the principles of biological evolution [9]. GA has been applied in variety of work such as engineering, business, bioinformatics, etc. The GA adopts a population units of analysis and each individual of the population corresponds to an encoding (binary, real, etc) of a potential solution to the problem of interest [10]. It proceeds in interactive manner and consists of stochastic operators such as selection, crossover and mutation by generating new population of individuals from the old ones. The pseudo code of the GA can be seen in Figure 3. Main merit of GA is that it is expected to avoid local optima frequently by promoting exploration of search space. However, the price one pays for implementing GA is its slowness, cause by crucial exploration mechanism employed.

```
Set epoch {Set maximum iteration}
iter = 0 {Start with initial iteration}
Initialize random population, P(iter)
aluate fitness for all individual in population, f(P(iter))
while iter < epoch
        iter = iter + 1
        Select P(iter) from P(iter  1)
        Crossover P(iter)
        Mutation P(iter)
        Evaluate f(P(iter))
end while
```

Fig. 3 Pseudo code of the Genetic algorithm

Multilayer perceptron with back propagation (BP) algorithm is often used to solve a great variety of real world problems [11]. Applications using BP algorithm as training method can be found in virtually every fields that uses ANN for problems that involves mapping a given set of inputs to a specified set of target outputs. It is simply an iterative gradient descent local search procedure to adapt network weights so the cost function, i.e. mean square error (MSE) defined by the difference between actual output and desired output reaches the minimum computed by the network. Advantage of BP is that the adjustment of weights is always toward the descending direction of the cost function and contributes to faster convergence speed around the optimum.

Despite the apparent dissimilarities between GA and BP methodologies, they can usefully complement each other since search feature of GA is population driven while BP is trajectory driven. Here, hybridization refers to the inclusion of problem-dependent knowledge in a general search template [12]. The hybrid algorithms that we use in this work are combination of two algorithms; where back-

propagation is used as mutation-like operation of the general search template GA. Figure 4 describes a complete pseudo code for hybrid GA-BP algorithm:

6. Result and Discussion

In this study, our dataset consists of 6 enzymes superfamilies extracted from PDB. Here we used 3200 enzymes in total as the training and testing samples. The number of these samples constituted the training set is 1200. Other proteins were allocated to the testing set (2000). Table 2 summarized the data used for the training and the testing processes. We used a same number of enzymes in training for each class while for testing; number of samples for each class was according to percentage of total enzymes in PDB database, respectively. For all samples, length of sequence is in the range of 40 to 600 amino acids. A total of 13 networks participated in the network system. All networks have the same topology; one input layer with 40 input nodes, one hidden layer and one output layer with 6 output nodes where each node represents one class of enzymes. The exception is in determining the number of hidden nodes, in which we applied two different approaches, i.e. using those proposed by Wanas et al. [13] and, Baum and Haussler [8]. For more details and the proof, readers can refer to their papers as given in the reference section. Table 3 shows the distributions of the training patterns and the number of hidden nodes in each of the network used in NNEC.

Set $epoch$ {Set maximum iteration}
$iter = 0$ {Start with initial iteration}
Set number of individuals, Id
Initialize random population of individuals (set of weights for network), $P(iter)$
while $iter < epoch$
Evaluate fitness for all individual in population, $F(P(iter))$
    for $i = 1{:}Id/2$ { Breeding new individuals }
        Select two individuals from $P(iter)$ using available *selection method*.
        Do *crossover* between selected individuals to form two offspring (new individuals).
        Apply *mutation* for both offspring.
        i. Let offspring become initial weight to train network using backpropagation algorithm.
        ii. Run for small epoch, i.e. 10 epochs.
        Both of offspring become new individuals in next population, $P(iter + 1)$
    end
end while

Fig. 4 Pseudo code of the hybrid GA-BP

Next, each network was trained using hybrid GA-BP algorithm with maximum epochs is set to 300. This process was repeated for 10 times to find the best set of weights. All parameters used in NNEC's training session are described in Figure 5. The NNEC was implemented using MATLAB software version R2006b with own-written code.

To evaluate performance for our network system, we measured the precision in simulation for the testing set. Precision is defined as followed:

$$Precision = \left(\frac{No.\ of\ test\ sequences\ predicted\ correctly}{Total\ no.of\ sequences\ participated}\right) * 100\%$$

Table 2: Size of the various training set and testing set in NNEC

Class	Family	Training set	Testing Set	Percentage (%)	Total
1	Oxidoreductases	200	354	17.7	554
2	Transferases	200	422	21.1	622
3	Hydrolases	200	918	45.9	1118
4	Lyases	200	164	8.1	364
5	Isomerases	200	92	4.6	292
6	Ligases	200	50	2.6	250
	Total	1200	2000	100	3200

Table 3: Distribution of the number of training patterns and number of hidden nodes used in NNEC

Network	No. of enzymes						Total	No. of hidden nodes	
	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6		Profile 1	Profile 2 ($\varepsilon = 0.2$)
1	200	200	200	200	200	200	1200	10	5
2	200	200	200	200	200	200	1200	10	5
3	199	195	188	199	184	192	1157	10	5
4	194	186	159	185	178	178	1080	10	5
5	185	172	135	176	145	162	975	10	4
6	175	147	106	157	131	151	867	10	4
7	161	126	72	114	98	140	711	9	3
8	128	79	55	105	88	119	574	9	2
9	94	61	40	82	77	98	452	9	2
10	71	43	33	62	48	88	345	8	2
11	54	29	25	44	34	63	249	8	1
12	40	26	18	32	25	35	176	7	1
13	24	20	11	25	23	24	127	7	1

We summarize the classification results for the 6 enzymes superfamilies for testing sets in Table 4. Looking at Table 4, which compare the performance of NNEC using two sets of hidden nodes, it is clear that no vast difference in average precision rate between them. Both profiles were consistent, displaying good performance (around average) at classifying testing sequences for first 5 enzymes superfamilies. However, there a little bit decreased in precision rate for Ligases enzyme superfamily. With average rate at 72.94%, neural system with hybrid GA-BP algorithm with profile 1 is the best system for NNEC. From now onwards, NNEC with profile 1 will officially become our enzyme classification system.

1)	Cost function, f	f = MSE $f = \frac{1}{2z}(T - Y)^2$, T = Target matrix, Y = Output matrix
2)	Initialize network architecture	
i.	No. of input nodes	40
ii.	No. of output nodes	6
iii.	Activation function:	
	hidden layer	Tan sigmoid
	output layer	Log sigmoid
3)	Initialize GA parameters.	
i.	No. of individuals	100
ii.	Encoding scheme	Binary
iii.	Mode of selection	Tournament selection
iv.	Prob. of crossover	0.9
v.	Prob. of mutation	0.01
vi.	Stopping criteria	Epoch : 300

Figure 5. Parameters for the system

Table 4: Precision of the testing result

	Precision	
Superfamily	Profile 1	Profile 2
Oxidoreductases	74.29	72.29
Transferases	70.59	70.12
Hydrolases	73.91	71.20
Lyases	78.18	72.73
Isomerases	76.09	68.48
Ligases	64.58	58.33
Average	**72.94**	**68.86**

7. Conclusion

In this paper, we adopted a new approach in encoding primary structure into real-valued vector that feeds into neural network using Kyte and Doolittle hydrophobicity scale [13]. Additionally, we used a set of neural network called NNEC with hybrid GA-BP algorithm in the weighting system. The results achieved showed that combination between GA and BP algorithm can be employed to optimize network structure. By using the method proposed by Wanas et al. [13] in finding the optimal number of hidden nodes in neural network, we successfully obtained good precision performance values for the testing datasets.

The accuracy rate for testing dataset with 6 classes of enzymes is reasonable (72.94% on average). To lead to better classification, it is possible to increase the number of training set since the total dataset studied is large. For future work, we will include more classes as network's target by using other datasets available such

as the PIR. In order for NNEC to reach the standard, we will make comparison with other existing classification algorithms.

ACKNOWLEDGEMENT

The authors acknowledge the financial support from Grants No. 01-01-02-SF0251, and the support received from the Bioinformatics Research Centre, Institute of Systems Biology (INBIOSIS), Universiti Kebangsaan Malaysia.

References

[1] M.O. Dayoff, R.M. Schwartz, and B.B. Orcutt, *A Model of evolutionary change in proteins*, Atlas of protein sequence and structure, National Biomedical Resource Found. 5(1978), pp. 345-358.

[2] K. Blekas, D.I. Fotiadis, and A. Likas, *Motif-based protein sequence classification using neural networks*, Journal of Computational Biology. 12(2005), pp. 64-82.

[3] J.T.L Wang, Q. Ma, D. Shasha, and C.H. Wu, *New techniques for extracting feature from protein sequences*, IBM: System Journal. 40(2001), pp. 426-441.

[4] E.D. Martin and A. Araque, *Astrocytes and the biological neural networks*, in *Artificial Neural Networks in Real Life Applications*, J.R. Rabunal and J. Dorrado, eds., IGI Global, USA, 2006, pp. 22-45.

[5] J.R. Rabunal and J. Puertas, *Hybrid system with artificial neural networks and evolutionary computation in civil engineering*, in *Artificial Neural Networks in Real Life Applications*, J.R. Rabunal and J. Dorrado, eds., IGI Global, USA, 2006, pp. 166-187.

[6] J.S. Almeida and S.Vinga, *Universal sequence map (USM) of arbitrary discrete sequences*, BMC Bioinformatics. 3(2002), pp. 3-6.

[7] N. Wanas, G. Auda, M.S. Kamel, and F. Karray, *On the optimal number of hidden nodes in a neural network*, IEEE Canadian Conference on Electrical and Computer Engineering, 1998.

[8] E.B. Baum and D. Haussler, *What size net gives valid generalization?*. Neural Computation. 1(1989), pp. 151-160.

[9] J.H. Holland, *Adaption in natural and artificial systems*, University of Michigan Press, Cambridge, 1975.

[10] R. Perkins and A. Brabazon, *Predicting credit ratings with a GA-MLP hybrid*, in *Artificial Neural Networks in Real Life Applications*, J.R. Rabunal and J. Dorrado, eds., IGI Global, USA, 2006, pp. 220-237.

[11] U. Seiffert, *Multiple layer perceptron training using genetic algorithm*, Proceedings of European Symposium on Artificial Neural Network, ESANN, 2001.

[12] E. Alba and J.F. Chicano, *Training neural networks with GA hybrid algorithms*, Lecture Notes in Computer Science, Springer Berlin/Heidelberg. 3102(2004), pp. 852 -863.
[13] W.R. Weinert and H.S. Lopes, *Neural networks for protein classification*, Applied Bioinformatics. 3(2004), pp. 41-48.
[14] A.D. Smith, *Oxford dictionary of biochemistry and molecular biology*, Oxford University Press, Oxford, 1997.
[15] *R. Bellman, Adaptive control processes: A guided tour*, Princeton University Press, New Jersey, 1961.