UNFORMATTED PREPRINT – **Applied Bioinformatics**, v. 3, n. 1, p. 41-48, 2004

Neural networks for protein classification

Wagner Rodrigo Weinert and Heitor Silvério Lopes

Laboratório de Bioinformática/CPGEI, Centro Federal de Educação Tecnológica do Paraná (CEFET-PR), Curitiba (PR), Brazil

Correspondence: Heitor Silvério Lopes, CPGEI/CEFET-PR, Av. 7 de setembro, 3165–80230–901 Curitiba (PR), Brazil; tel +55 41 310 4694; fax +55 41 310 4683; e-mail *hslopes@cpgei.cefetpr.br*

Abstract: This paper describes a biomolecular classification methodology based on Multilayer Perceptron (MLP) neural networks. The developed system is used to classify enzymes found in the Protein Data Bank. The primary goal of classification, here, is to infer the function of an (unknown) enzyme by analysing its structural similarity to a given family of enzymes. A new codification scheme was devised to convert the primary structure of enzymes into a real-valued vector. The system was tested with different number of neural networks, training set sizes and training epochs. For all experiments, the proposed system achieved a higher accuracy rate when compared with profile hidden Markov models (HMMs). Results demonstrated the robustness of this approach and the possibility of implementing fast and efficient biomolecular classification using neural networks.

Keywords: neural networks, protein classification, enzyme

Introduction

Molecular biology is a field that has experienced dramatic developments in recent years. A large number of data are constantly being generated thanks to several genome-sequencing projects throughout the world. However, little information can be readily extracted from these data and, therefore, data analysis has became a central issue in molecular biology (Hu 1998). The analysis includes methods and algorithms for pre-processing, visualisation, knowledge discovery and data-mining of genomic and proteomic data.

A vertiginous increase in the rate at which new protein structures are discovered has taken place as a by-product of ongoing sequencing projects. Therefore, more efficient methods for protein analysis, including classification, are necessary. The protein classification problem lies not in predicting the function or the secondary/tertiary structure of a new protein. Instead, researchers seek to be able to classify a new protein as belonging to a given family with previously known characteristics. From this they hope to infer its function and structural characteristics. Most proteins share similar structures (in particular, their primary structures), since many of them have a common

evolutionary origin (Murzin et al 1995). On the other hand, proteins of unrelated families can also have structures in common. This two-fold nature of structure makes protein classification difficult.

Enzymes are a subclass of proteins that are specialised in catalytic activity (Lehninger et al 1998a). They are large and complex molecules, present in all living beings, and play an essential role in biochemical reactions. They control several vital functions, including many metabolic processes that convert nutrients into energy and into other products necessary to cell functioning.

Artificial NNs (Fausett 1994) have been used to solve complex problems in several areas, such as engineering, computer science and bioinformatics (Narayanan et al 2002). Amongst other applications, NNs are particularly suitable for pattern recognition. In this work, we have developed a neural network (NN)-based system for enzyme classification using a Multilayer Perceptron (MLP).

Background

Molecular biology

Proteins are composed of amino acids connected by peptide bonds. The primary structure of proteins is regarded as the linear sequence of amino acids in a polypeptide chain. Proteins can be grouped into families and these families into superfamilies according to features – such as hydrophobicity, composition, structure, length, three-dimensional shape, and electric charge (eg isoelectric point) (see Murzin et al 1995, Lehninger et al 1998a) – with the objective of establishing the common biological functions.

The amino acid sequence of a protein ultimately determines its function. Proteins usually have segments in their sequence of amino acids known as motifs (Attwood et al 1996) that are crucial for their biological functions, and that can be used for their identification.

There are 20 different types of amino acids (see Table 1) that are combined in a linear sequence, which has the necessary information to generate a unique three-dimensional structure. Theoretically, the number of possible combinations of amino acids is infinite (Lehninger et al 1998b). Amino acids, in turn, vary in the side chain. The physical and chemical properties of the side chains of the amino acids of a protein (for instance, the fact that some of them have affinity with water) are important for the folding of the protein and its function. Like the proteins that make, amino acids can be classified in several ways, such as by electric charge, molecular weight and hydrophobicity. Kyte and Doolittle (Kyte and Doolittle 1982) proposed a hydrophobicity scale for all 20 amino acids ranging from –4.0 (most hydrophilic) to +4.5 (most hydrophobic). This scale is shown in Table 1, where it can be seen that amino acids can be also categorised as hydrophobic, neutral or hydrophilic.

Protein databases and classification

There are many molecular biology databases available in the Internet, such as Swiss-Prot, PIR, InterPro, ProDom, Protein Data Bank (PDB). In our study, we extracted the information about primary structure from PDB – an international repository of information about the three-dimensional structure of biological macromolecules. The contents of this database come from X-ray crystallography and nuclear magnetic resonance imaging. Figure 1 shows an example of data extracted from a PDB file: the primary structure of the enzyme 1CBH (C-terminal domain of cellobiohydrolase I; code: E.C.3.2.1.91) that belongs to Hydrolase family of enzymes and has only 36 amino acids.

Although, as yet, there is not consensus about protein classification, several properties can be used for this purpose, such as composition, number of side chains, 3-D folding shapes or biological function (Lehninger et al 1998a). Another classification was proposed by (Murzin et al 1995) that presents a model based on the protein domain. Many computational techniques have been used to classify proteins into families, such as structural transformations (Ohkawa et al 1996), data compression (Chiba et al 2001), genetic programming (Koza 1997) and Markov chains (Eddy 1998). They have demonstrated limited applicability and results. However, few studies published in the recent literature have applied NNs to protein classification. This approach has gained some attention in the analysis of molecular sequences. For instance, Wang et al (2000) proposed a new technique to extract data from proteins with a Bayesian NN for classification. Wu et al (1990) explored the informative segments of sequences and used a three-layer NN with a backpropagation algorithm for classification..

Artificial neural networks

An artificial NN is a computational model that mimics the functioning of the human brain by using a set of simple processing units. Every processing unit of an NN represents a neuron, which is interconnected with other neurons. A weight is assigned to each connection between neurons, and it represents the influence of one neuron on the other. Every neuron processes only local data received through its input connections and has a unique output that is fed to other neurons. The emergent intelligent behaviour of an NN comes from the interactions, between its processing units, that occur when it is presented with input data (Fausett 1994).

A processing unit, as proposed by McCulloch and Pitts in 1943 (Figure 2), can be summarised in the following way. The inputs of the processing unit, $X_1, X_2, \cdots, X_t$, are multiplied by weights $W_1, W_2, \cdots, W_t$ that indicate the influence of each input on the output (Y) of the processing unit. Then, the weighted inputs are summed, producing an activity level for the processing unit. If this activity level exceeds a given threshold, the processing unity produces a predefined output

(usually, not zero). Most NN models have a training rule, such that the weights are iteratively adjusted according to the input (training) patterns. In other words, NNs usually learn by example.

A typical NN architecture is shown in Figure 3, where a feedforward network of interconnected neurons (processing units) is organised in layers. The input layer is where input data is presented to the NN; hidden layers are where most processing effort takes place; the output layer is where the final result is available. An NN is specified by its topology (number of layers, number of processing units per layer and connectivity among them), by the transfer function of the processing units and by its training procedure.

The training procedure is critical for the NN to accomplish its purpose. Usually, training is a supervised procedure, in which a set of input-output patterns (a training set) is presented to the NN and the computed result is compared to the actual value. The difference is used to update the weights of each layer using the generalised delta rule. This training algorithm is known as 'backpropagation'. After several training epochs, when the error between the actual output and the computed output is less then a previously specified value, the NN is considered trained. The knowledge learnt by an NN is effectively represented by the set of weights, that is, the strength of the connections between neurons. Once trained, the NN can be used to process new data, classifying them according to its acquired knowledge.

Sequence encoding

The natural encoding of the primary structure of proteins is a string of letters. However, this encoding is not appropriate for NNs, since it demands numerical (and preferably, normalised) inputs. Therefore, proteins have to be encoded in a more suitable way.

Proteins – including enzymes, in general, are composed by a variable number of amino acids, from tens to thousands. The encoding process proposed here allows differently sized enzymes to be processed by a predefined, fixed-size NN, as shown in the following sections.

The core of the encoding procedure is the conversion of a string of amino acid symbols (letters) into a real-valued vector. This was accomplished using the Kyte and Doolittle hydrophobicity scale shown in Table 1. The converted values were normalised in the range 0.05 – 1.00, in steps of 0.05.

The neural network system

Once the encoding scheme is defined, the next step is the construction of the NN system for classification. In this work, we consider that there can be n different classes (numbers of families of enzymes), and in each class there can be m cases (number of enzymes). The length of a given

sequence S_{ij} is defined as λ_{ij} (i=1⋯n; j=1⋯m). For the sake of simplification, we considered that all classes in the training set (TS) have the same m. But, in the evaluation set (ES) this was not necessary, as will be shown.

The proposed system uses not one, but a set of NNs for processing the inputs. Each NN of the set is responsible for a segment of the sequence. The architecture of the NNs is fixed and defined a priori as a three-layer perceptron (MLP). The number of NNs used is as a function of the length of the sequences of the training set and the partition size (t, number of amino acids into which the sequences will be broken) chosen for the problem. The size of t will also be the number of inputs per NN and is a user-defined parameter. Each sequence of the TS is divided into k subsets of amino acids, according to equation (1).

$$k = t^{-1}.\max_{i=1..n;\, j=1..m}\left(\lambda_{ij}\right) \tag{1}$$

The result of equation (1) is rounded down to the nearest integer. That is, the fraction related to the end of the sequence is ignored. The system also ignores any sequence whose length is smaller than t.

The number of NNs of the neural system is k. Every NN will process data from the corresponding partition, so that the first partition of the sequences will be processed by the first NN, the second partition, by the second NN, and so on.

This procedure determines the number of NNs and their size. The topology, as mentioned before, is fixed, using a MLP, and is represented by the ratio of three parameters x:y:z. The first parameter is the number of neurons in the input layer, here, t. The second parameter is the number of neurons in the hidden layer, set to $2t + 1$, as usual (Fausett 1994). The last parameter is the number of neurons of the output layer, corresponding to the number of classes of the classification problem, that is, n. Figure 4 depicts the neural system proposed.

Because sequences in the training set usually have different lengths, the number of training cases (protein fragments) will be different for each NN of the neural system. Therefore, each NN will contribute differently to the final result. Accordingly, a classification weight (cw) is defined as a function of the number of training cases for each NN. Several schemes could be devised to compute cw for each NN. The easiest method is simply to count the number of protein fragments used for the training set. A higher number of instances in the training of a given NN, will merit more confidence to the classification. The set of cw will be used further in the classifier module during evaluation.

Figure 5 shows an example of the approach. Suppose that there are two classes (n=2), each class with five sequences (m=5). Suppose, also, that the longest sequence (S_{12}) has 285 amino acids. Using an arbitrary t, say 40, equation (1) gives the number of partitions (NNs, k) as 7. Therefore, the neural system for this example is composed by 7 NNs, with topology 40:81:2. If the set of

sequences of Figure 5 were used as the training set, the first NN (k=1) would be trained with 10 protein fragments (5 of each class); the second NN with 8; the third NN with 7, and so on. As mentioned before, each NN contributes differently to the final result of the classifier, therefore justifying the classification weights (*cw*) in the last line of Figure 5.

The output vector of an NN is the current activations of the neurons in the output layer, for a given input pattern. Here, all NNs were trained with a backpropagation algorithm that considered outputs as binary. That is, for the example of Figure 5, an input pattern belonging to the first class should have an output vector (***1 0***); conversely, if the input pattern belongs to the second class, the output should be (***0 1***).

The classifier module

The test of the neural system (as part of training), as well as the classification of new, unseen proteins was done as follows. First, it was assumed that the proteins are codified in the same way as the training set. Then, the sequence was partitioned into segments of t amino acids. The first segment was submitted to the first NN, the second segment to the second NN, and so on. If, for a given sequence, the length of the sequence was longer than $k \cdot t$ ($S_{ij} > k \cdot t$), then all amino acids after the $k \cdot t$-th position were ignored. The output vector of all NNs was the input to the classifier module. The set of all NNs outputs was considered to be a single, n x k matrix, called **Y**, where column i represented the output vector of the i-th NN. Also, the set of all classification weights were considered as an n x 1 vector (***cw***). The final classification performed by the classifier module is given by equation (2), and is the maximum value of the product between matrix ***Y*** and vector ***cw***:

$$\mathbf{Y} \cdot \mathbf{cw} = \begin{pmatrix} y_{11} & y_{21} & \cdots & y_{1k} \\ y_{21} & y_{22} & & y_{2k} \\ \vdots & \vdots & & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nk} \end{pmatrix} \cdot \begin{pmatrix} cw_1 \\ cw_2 \\ \vdots \\ cw_n \end{pmatrix}$$

$$class = \max(\mathbf{Y} \cdot \mathbf{cw}) \qquad (2)$$

Enzyme classification

A number of experiments were done to test the performance of the NN system in the protein classification problem. The following issues were investigated: the classification performance of the proposed system using a large dataset (of enzymes); the influence of the size of the training set in the learning process of the NNs; the number of NNs necessary for a satisfactory balance between computational time and classification performance.

For all experiments related, we used the same topology for all neural networks: a 40:81:6 MLP using backpropagation. Every NN was trained for 300 epochs with a learning rate (α) of 0.1. The NN system was implemented using MATLAB® version 5.3 (MathWorks Inc, Natick, MA, USA) and the classifier software was written using the Delphi programming language, version 5.0 (Borland Corp, Scotts Valley, CA, USA). The commented source code in MATLAB® is available in http://bioinfo.cpgei.cefetpr.br/en/softwares.htm. The source of data for the experiments was the primary structure of proteins found in PDB (http://www.rcsb.org/pdb/), release 102 (october/2002).

Classification performance

For this experiment, we used a whole enzyme superfamily extracted from PDB. A total of 8339 enzymes were used, divided into six families. Table 2 summarises the data used for the TS and the ES.

All the results reported were obtained by performing a modified five-fold cross-validation procedure (Hand 1997). First, a given number of proteins were randomly drawn from the dataset (see TS in Table 2) for each of the six families. The sum of these samples constituted the TS (600). All the other proteins were allocated to the evaluation set (7739). Then, the neural system was trained and later evaluated using this partition. The accuracy rate on the ES was computed as the ratio of the number of correctly classified proteins to the total number of proteins, as is standard in the literature. Next, a new sampling was taken from the dataset to form another TS and ES, and the training and evaluation processes were repeated. This procedure was repeated five times and the final results were reported as the averaged accuracy rate over these five runs.

The performance of our proposed neural system was compared with a classification procedure based on hidden Markov models (HMMs). For this purpose, we used the software package HMMER 2.2 (http://hmmer.wustl.edu/) that uses HMMs (Eddy 1998). HMMs are a well-known statistical modelling technique frequently applied to the analysis of time series and biological sequences (Eskin et al 2003). We stress that our neural system and the HMMER were given exactly the same training and evaluation sets in each of the five runs of the cross-validation procedure, making the comparison as fair as possible.

The use of HMMER for protein classification encompasses three steps. First, a multiple sequence alignment is done using the TS. For this purpose we used the software ClustalX, version 1.81 (http://www-igbmc.u-strasbg.fr/BioInfo/ClustalX/Top.html), for generating six files (one for each class) with the multiple alignment of the proteins. The second step is building a HMM that represents each class of the TS. This was done using as input the pre-aligned file mentioned before and the module HMMBUILD. This program creates a 'profile HMM' for the family based on the examples given. Additionally, we used HMMCALIBRATE to optimise the generated models, so as

to improve the classification performance. Finally, the third step is the classification of the ES using the profile models generated. This was done using HMMPFAM.

The results of these experiments, regarding classification accuracy, are shown in Tables 3 and 4 for our neural system and for HMMER, respectively. The last column of these tables shows the averaged accuracy rate for the fivefold cross-validation, together with the standard deviation. The number of NNs trained in the five partitions was different, as expected, owing to the different lengths of the enzymes in each partition. The number of NNs for each partition, according to the procedure explained in section on *The Neural Network system*, was (in order) 240, 145, 145, 344 and 245.

The use of accuracy rate to assess classification performance is standard in the classification literature (Hand 1997), but sometimes this measure can be misleading since it does not discriminate between positive and negative cases. That is, the accuracy rate is the sum of the correctly classified cases. Another useful way to measure a system's classification performance is using 'sensitivity' and 'specificity', two indicators commonly used in medical and life sciences. These measures are frequently used in two-class problems, but can be readily adapted for multiclass problems, as will be shown.

When using a system for classifying a protein of unknown class, depending on the class predicted by the system and on the actual class of the protein, one of the following four types of result can be observed:

- True positive (*TP*) – the system predicts that the protein belongs to a given class and the protein really does belong to that class;
- False positive (*FP*) – the system predicts that the protein belongs to a given class but, in fact, it does not belong to it;
- True negative (*TN*) – the system predicts that the protein does not belong to a given class, and indeed it does not belong to it;
- False negative (*FN*) – the system predicts that the protein does not belong to a given class but, in fact, the protein does belong to it.

Based on these parameters, sensitivity (*Se*) and the specificity (*Sp*) can be defined as follows:

$$Se = TP/(TP + FN) \tag{3}$$

$$Sp = TN/(TN + FP) \tag{4}$$

Sometimes sensitivity and specificity are called true positive rate and true negative rate, respectively. Sensitivity measures the ability of the classifier system to correctly assign a protein to its real class. In the other hand, specificity measures the ability of the system to reject a given protein as belonging to a class to which it does not belong.

Using the same data partitions and cross-validation as before, sensitivity and specificity were computed for both our NN system (Table 5) and HMMER (Table 6). According to the results, the proposed NN system can be successfully applied to the task of biomolecular data classification, showing, on average, better results when compared with HMMs, regarding predictive accuracy. The high standard deviation values for most classes using HMMs reveal that this method is strongly dependent on the training set and has a reduced generalisation capability, one of the strengths of the NNs system.

For both approaches, specificity values were always higher than sensitivity. This means that both systems are more efficient at predicting when a given protein does not belong to a class than the opposite. Again, the NN system performed better than HMMER, when the values for both sensitivity and specificity are taken into account.

Training set sizing

Since the NN system demonstrated satisfactory performance in classifying proteins with the enzyme dataset, a further study was done to analyse the influence of the size of the training set. A significant amount of effort is spent in training an NN. The main parameter that affects the training time of an NN is the number of proteins in the training set. Therefore, efforts to reducing its size are worthy, provided the accuracy performance is kept at a satisfactory level.

In the experiments of the previous section, the size of the training set for each family was 100 proteins (see Table 2). The experiments were repeated, using the enzymes dataset with the same number of partitions, but with only 50 proteins per family in the TS, extracted at random from PDB, and the rest were assigned to the evaluation set. Results of this experiment are shown in Table 7, and can be compared with those reported in Table 3.

Neural system sizing

As mentioned previously, the number of NNs in our neural system is a function of the length of the sequences being processed (equation (1)). In the previous experiments, we used from 145 to 344 NNs. Since the computational time for training the NNs is a function of the number of NNs themselves, we were interested in investigating the decrease in performance of the system when the number of NNs is arbitrarily decreased.

Using the same data and partitions shown in Table 2, we calculated the average length for each partition of the training set, as shown in Table 8. Using the average size of the proteins, we divided this value by the number of input neurons in our NNs (note that we used a fixed topology of 40:81:6). Therefore, the suggested number of NNs in the neural system was 20 (rounded up). The same experiments reported in Table 3 were repeated (with 100 proteins per class), but now using

only 20 NNs for the neural system. When the results from those experiments (presented in Table 9) were compared with those in Table 3, there was no significant difference in performance when fewer NNs were used.

Conclusions

This paper we presents a new approach for the protein classification problem using NNs and information from the primary structure of proteins. In this system, proteins are encoded into a real-valued vector using the Kyte and Doolittle hydrophobicity scale. Another key point is that we used a set of NNs and a weighting system.

The accuracy rate of the NN system was compared with HMMER for a dataset with six classes of enzymes. For all classes, the NN system outperformed HMMER (74.6 % versus 44.5 %, on average). Considering sensitivity and specificity, both systems (NN system and HMMER) were consistent, displaying better performance at identifying negative cases (proteins not belonging to the class being considered) than positive ones. Notwithstanding, the average sensitivity and specificity of the proposed NN system were better than HMMER. As expected, HMMs are very sensitive and dependent on the TS. For our experiments, they had poor generalisation capability, one of the evident advantages of the NN system.

The accuracy rate of the system decreased from 74.6 % (on average) to 63.6 % when the number of training proteins was decreased from 100 to 50. This experiment shows how sensitive the proposed system is to the size of the TS. It is possible that the use of larger TSs can lead to better results, but this was not possible with the current dataset (see Table 2).

Also, the robustness of the system was tested when a small number of NNs (20) was used, instead of the number calculated by equation (1). The accuracy rate of the system under these conditions was not significantly decreased. The main reason for the good performance was the weighting system used, which balanced the importance of the output of each NN for the final classification.

Further research will include more experiments using other datasets (drawn from SwissProt, for instance) and the study of the NN system's performance with a growing number of classes. We speculate that further use of the system here described could include determining the functional class of a recently discovered protein, without prior knowledge of its structural aspects. Also, in the emerging field of de novo protein design, a NN system like the one described herein could be useful, either alone or 'in tandem' with other methods such that proposed by Wei et al (2003).

Acknowledgements

The authors would like to thank CNPQ for the financial support (process 475049/2003-9) and the PQ grant to HS Lopes, as well as CAPES for the grant to W Weinert. Special thanks to Dr. Sue Hallas for her careful review of the final manuscript.

References

Attwood TK, Beck ME, Bleasby AJ, Degtyarenko K, Parry Smith DJ. 1996. Progress with the PRINTS protein fingerprint database. *Nucleic Acids Res*, 24:182–8.

Chiba S, Sugawara K, Watanabe T. 2001. Classification and function estimation of protein by using data compression and genetic algorithms. In Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea, may/27-30. Volume 2. Piscataway, USA: IEEE Press, p. 839–44.

Eddy SR. 1998. Profile hidden Markov models. *Bioinformatics*, 14:755–63.

Eskin E, Noble WS, Singer Y. 2003. Protein family classification using sparse Markov transducers. *J Comput Biol*, 10:187–214.

Fausett L. 1994. Fundamentals of neural networks. Upper Saddle River: Prentice-Hall, Inc.

Hand DJ. 1997. Construction and assessment of classification rules. New York: John Wiley and Sons.

Hu Y. 1998. Biopattern discovery by genetic programming. In Koza JR, ed. Proceedings of the Third Annual Genetic Programming Conference, Madison, USA, july/22-25. San Francisco: Morgan Kaufmann Publishers, p. 152–7.

Koza JR. 1997. Classifying protein segments as transmembrane domains using genetic programming and architecture-altering operations. In Bäck T, Fogel DB, Michalewicz Z, eds. Handbook of Evolutionary Computation. Volume 6. Bristol, UK: Institute of Physics Publishing, p. 1–5.

Kyte J, Doolittle R. 1982. A simple method for displaying the hydropathic character of proteins. *J Mol Biol*, 157:105–32.

Lehninger AL, Nelson DL, Cox MM. 1998a. Principles of biochemistry with an extended discussion of oxygen-binding proteins. 2 ed. New York: Worth Publishers Inc.

Lehninger AL, Nelson DL, Cox MM. 1998b. Principles of biochemistry. 2 ed. New York: Worth Publishers.

McCulloch WS, Pitts W. 1943. A logical calculus of the ideas immanent in nervous activity. *Bull Math Biol*, 5:115–33.

Murzin AG, Brenner SE, Hubbard T, Chothia C. 1995. SCOP: A structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol*, 247:536–40.

Narayanan A, Keedwell EC, Olsson B. 2002. Artificial intelligence techniques for bioinformatics. *Appl Bioinformatics*, 1.4:191–222.

Ohkawa T, Namihira D, Komoda N, Kidera, A, Nakamura, H. 1996. Protein structure classification by structural transformation. In Bourbakis NG, ed. Proceedings of IEEE International Joint Symposia on Intelligence and Systems, Rockville, USA, november/4-5. Los Alamitos, USA: IEEE Computer Society Press, p. 23-9 .

Wang JTL, Ma Q, Shasha D, Wu CH. 2000. Application of neural networks to biological data mining: a case study in protein sequence classification. In Ramakrishnan R, ed. Proceedings of 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, USA, august/20-23. New York, USA: ACM Press., p. 305–9.

Wei Y, Kim S, Fela D, Baum J, Hecht MH. 2003. Solution structure of a de novo protein from a designed combinatorial library. *Proc Natl Acad Sci USA*, 100:13270–3.

Wu CH. 1997. Artificial neural networks for molecular sequence analysis. *Comput Chem*, 21:237–56.

Wu CH, Whitson GM, Montllor GJ. 1990. PROCANS: a protein classification system using a neural network. In Proceedings of IEEE International Joint Conference on Neural Networks, San Diego, USA, june/17-21. Volume 2. Los Alamitos, USA: IEEE Computer Society Press, p. 91–6.

TQSHYGQCGGIGYSGPTVCASGTTCQVLNPYYSQCL

Figure 1 Primary structure of enzyme 1CBH, belonging to Hydrolase family, with only 36 amino acids.

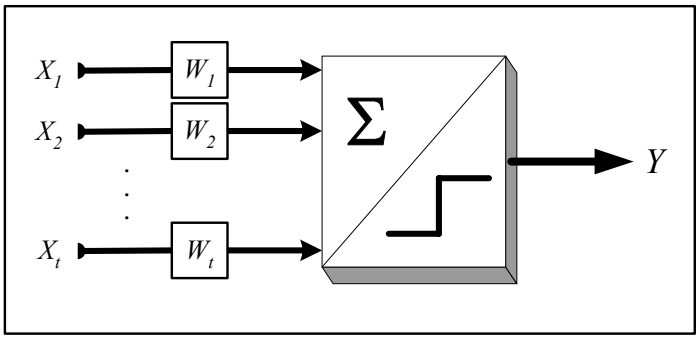

Figure 2 McCulloch and Pitts (1943) processing unit for NNs.

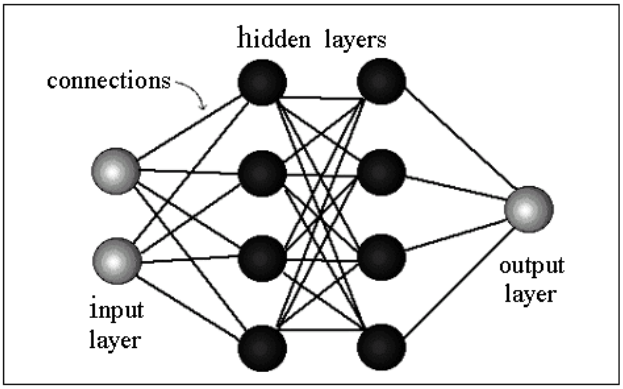

Figure 3 Feedforward NN organised in layers.

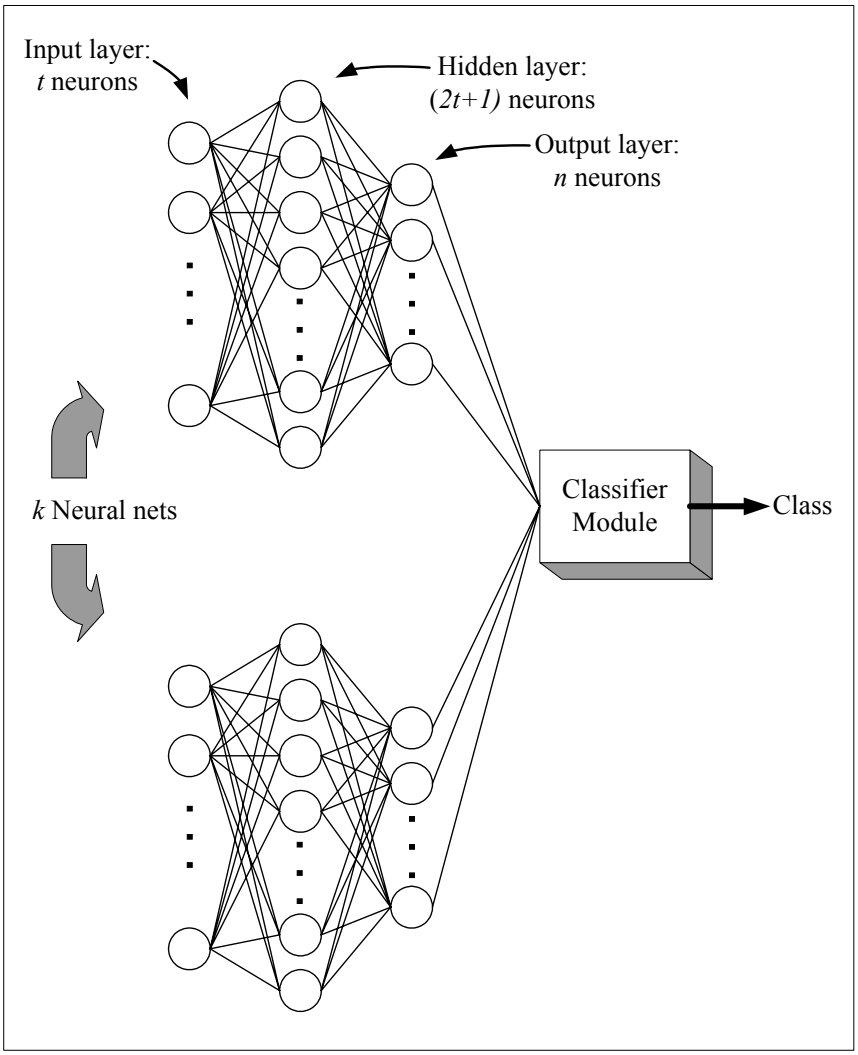

Figure 4 Generic representation of the neural system built with *k* three-layer NNs. Each NN has *t* neurons in the input layer, *n* in the output layer, and *2t+1* neurons in the hidden layer (topology *t:2t+1:n*).

		Partitions (*k*=7)						
Class (*n*=2)	**Sequence** (*m*=5)	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1	■	■	■				
	2	■	■	■	■	■	■	■
	3	■	■	■	■	■		
	4	■	■					
	5	■						
2	1	■	■	■	■	■		
	2	■	■	■	■			
	3	■						
	4	■	■	■				
	5	■	■	■	■			
	Total	10	8	7	5	3	1	1
	cw	1	0.8	0.7	0.5	0.3	0.1	0.1

Figure 5 Example of the training set, showing how the number of partitions and the classification weights (*cw*) are obtained for neural networks with an arbitrary number of inputs (*t*).

Table 1 Kyte and Doolitle (K&D) hydrophobicity scale and the normalised scale used in the neural system

Amino acid		*K&D*	*Normalised*	*Type*
name	*symbol*	*scale*	*value*	
Isoleucine	I	+4.5	0.05	Hydrophobic
Valine	V	+4.2	0.10	Hydrophobic
Leucine	L	+3.8	0.15	Hydrophobic
Phenylalanine	F	+2.8	0.20	Hydrophobic
Cysteine	C	+2.5	0.25	Hydrophobic
Methionine	M	+1.9	0.30	Hydrophobic
Alanine	A	+1.8	0.35	Hydrophobic
Glycine	G	-0.4	0.40	Neutral
Threonine	T	-0.7	0.45	Neutral
Serine	S	-0.8	0.50	Neutral
Tryptophan	W	-0.9	0.55	Neutral
Tyrosine	Y	-1.3	0.60	Neutral
Proline	P	-1.6	0.65	Neutral
Histidine	H	-3.2	0.70	Hydrophilic
Glutamine	Q	-3.5	0.75	Hydrophilic
Asparagine	N	-3.5	0.80	Hydrophilic
Glutamic acid	E	-3.5	0.85	Hydrophilic
Aspartic acid	D	-3.5	0.90	Hydrophilic
Lysine	K	-3.9	0.95	Hydrophilic
Arginine	R	-4.0	1.00	Hydrophilic

Table 2 Size of the training set (TS) and evaluation set (ES) of the enzyme families used in the experiments

Class	*Family*	*TS*	*ES*	*Total*
1	Oxidoreductases	100	1383	1483
2	Transferases	100	1666	1766
3	Hydrolases	100	3725	3825
4	Lyases	100	575	675
5	Isomerases	100	281	381
6	Ligases	100	109	209
Total		600	7739	8339

Table 3 Classification results for the NNs system

Family	*Accuracy rate (%)*					
	Partition					*Mean ± sd*
	1	*2*	*3*	*4*	*5*	
Oxidoreductases	85.7	86.7	84.3	80.8	72.4	81.9 ± 5.8
Transferases	75.8	73.1	75.0	64.4	71.7	72.0 ± 4.5
Hydrolases	74.4	73.7	77.2	76.6	64.6	73.3 ± 5.0
Lyases	75.6	77.0	81.9	62.9	71.3	73.7 ± 7.1
Isomerases	71.5	79.0	84.6	78.2	67.6	76.1 ± 6.6
Ligases	76.1	79.8	56.8	68.8	70.6	70.4 ± 8.7
Average						74.6 ± 7.02

Table 4 Classification results for HMMER

Family	*Accuracy rate (%)*					
	Partition					*Mean ± sd*
	1	*2*	*3*	*4*	*5*	
Oxidoreductases	18.0	40.4	0.8	56.9	85.9	40.4 ± 33.2
Transferases	30.4	19.9	34.4	31.9	0.78	23.5 ± 13.8
Hydrolases	46.3	51.3	64.5	62.8	31.4	51.3 ± 13.4
Lyases	63.4	40.0	27.1	32.3	29.2	38.4 ± 14.8
Isomerases	68.6	66.5	49.8	50.5	58.0	58.7 ± 8.7
Ligases	67.8	48.6	65.1	48.6	44.0	54.8 ± 10.8
Average						44.5 ± 20.1

Table 5 Sensibility and specificity for the NN system

Family	*Sensitivity*	*Specificity*
Oxidoreductases	82.0 ± 5.8	91.2 ±1.1
Transferases	72.1 ± 4.6	92.5 ±3.3
Hydrolases	73.3 ± 5.1	95.4 ±1.9
Lyases	73.8 ± 7.1	95.7 ±0.9
Isomerases	76.2 ± 6.7	96.9 ± 0.8
Ligases	70.5 ± 8.8	97.2 ± 0.6
Average	74.65 ± 7.0	94.82 ± 2.7

Table 6 Sensibility and specificity for HMMER 2.2

Family	*Sensitivity*	*Specificity*
Oxidoreductases	40.5 ± 33.2	73.6 ± 23.5
Transferases	23.5 ± 13.9	93.1 ± 5.3
Hydrolases	51.3 ± 13.5	82.6 ± 12.2
Lyases	38.4 ± 14.8	95.2 ± 4.3
Isomerases	58.7 ± 8.8	94.7 ± 5.3
Ligases	54.9 ± 10.8	88.3 ± 15.9
Average	44.50 ± 20.1	87.9 ± 14.3

Table 7 Classification results for the NN system using 50 proteins per family in the training set

Family	*Accuracy rate (%)*					
	Partition					*Mean ± sd*
	1	*2*	*3*	*4*	*5*	
Oxidoreductases	55.5	72.8	75.5	65.3	61.0	66.0 ± 8.2
Transferases	68.3	43.0	48.0	57.5	61.1	55.5 ± 10.1
Hydrolases	59.6	59.3	56.7	68.7	51.4	59.1 ± 6.2
Lyases	72.1	79.0	62.5	75.5	70.8	71.9 ± 6.1
Isomerases	65.5	59.8	69.1	80.6	61.3	67.3 ± 8.3
Ligases	65.4	66.7	57.8	69.8	48.4	61.6 ± 8.6
Average						63.6 ±9.1

Table 8 Average length of the proteins in the training set

Partition					*Average*
1	*2*	*3*	*4*	*5*	
734.99	743.8	789.63	767.42	773.86	761.94

Table 9 Classification results using 20 neural networks

Family	*Accuracy rate (%)*					
	Partition					*Mean ± sd*
	1	*2*	*3*	*4*	*5*	
Oxidoreductases	84.2	85.3	83.5	79.2	72.4	80.9 ± 5.2
Transferases	76.2	74.2	75.8	65.0	71.8	72.6 ± 4.5
Hydrolases	75.0	74.7	78.0	77.0	65.2	73.9 ± 5.0
Lyases	74.2	77.7	82.2	64.0	72.0	74.0 ± 6.8
Isomerases	71.8	79.0	85.0	77.9	66.1	75.9 ± 7.2
Ligases	76.1	77.9	56.8	66.0	69.7	69.3 ± 8.4
Average						74.4 ± 6.8