How to Explicit A Neural Network Trained to Predict Proteins Secondary Structure

I.Tchoumatchenko, F. Vissotsky, J.-G. Ganascia[1] *

ACASA, LAFORIA-CNRS, Université Paris-VI, 4 Place Jussieu, 75252 Paris, CEDEX 05 France
tel: 33-1-44-27-70-09
fax: 33-1-44-27-70-00
email:(irina,vissot,ganascia)@laforia.ibp.fr

Abstract. We have trained a neural network to predict the secondary structure of globular proteins with emphasis both on maximizing prediction accuracy and expliciting a neural network's work as a clear decision scheme. First, we present the results of our computational experiments with the basic neural network usually used to deal with the secondary structure prediction problem. Second, we impose some important constraints on the basic neural network units' weights and propose the modification of the backpropagation learning algorithm in order to extract a clear decision scheme learned by the so-constrained neural network. Third, we describe an experimentally found convergence technique respecting the imposed constraints almost without loosing the neural network's prediction accuracy. The decision scheme found by our constrained neural network is compared with the existing knowledge on the protein structure, in particular, with the Chou-Fasman and the GOR secondary structure prediction theories.

Keywords: neural networks, protein secondary structure prediction, extracting rules from neural networks

1 Introduction

It is widely hypothesized that the amino acid sequence is theoretically sufficient to determine the structure and a folding of a protein [1]. However, predicting the tertiary structure of folded proteins solely from their primary structure is extremely difficult. To facilitate the prediction, the tertiary structure was described in some intermediate alphabet called secondary structure, and the whole protein folding prediction was reduced to the secondary structure prediction as a first step towards the goal of understanding protein folding.

* This work is supported by a grant from the DRET and the COGNISCIENCE program

Many attempts have been made to solve the problem of predicting protein secondary structure from their primary sequence but the best performance results are still disappointing. Today the best methods of prediction achieve an accuracy about 63%. Established prediction methods include: the statistics-based method of Chou-Fasman [3], the GOR information theory-based method [4] and the method of hand-crafted expert secondary structure patterns [5] [6].

Recently both symbolic induction [9] [7] and artificial neural networks [10] have been applied to predict protein secondary structure. Using symbolic machine learning methods some rules characterizing the relationship between primary and secondary structure were obtained [9] [7] but a small learning set under selected circumstances (a given type of the protein structural domain) makes difficult the comparison with other prediction methods. The best prediction accuracy was achieved using neural networks [10], however, the neural network-based approach for the protein secondary structure prediction has the large disadvantage that the predictions made have little explanation in humanely comprehensible terms.

To tackle this problem of the neural network-based prediction we propose a method for extracting a decision scheme learned by a neural network. Such a scheme provides a clear description of neural network's decision making process, that is an important step towards our goal – a symbolic representation of the mapping between primarily and secondary protein structure discovered by the neural network. As actually we use no initial knowledge and rely only on the discrimination/generalization properties of the suitably constructed and appropriately constrained neural networkwe can hope to extract a decision scheme comparable to the Chou-Fasman statistical theory or to the GOR information-based theory.

In this paper, first, we report the results of our computational experiments with the basic neural network taking into account some particular features of the problem. Second, we propose a constrained-based method for extracting a decision scheme from a neural network trained to predict the protein secondary structure. Third, we describe the practical realization of the proposed method – a convergence technique respecting the imposed constraints. Finally, we analyze an extracted decision scheme in terms of existing knowledge about protein secondary structure, namely, in terms of the Chou-Fasman and the GOR theories.

2 Experiments with A Basic Neural Network

In most of neural network-based studies of proteins secondary structure prediction the Nettalk-like network is used [10], which we will call here a basic neural network.

The basic neural network is a one-hidden layer perceptron, which receives the sequence of amino acids as an input. A fixed size window is slipped through the input sequences and the network is trained by the backpropagation to produce at the output the secondary structure corresponding to the central residue. Each amino acid is coded by 21 bits: one for each of the 20 amino acids and one for the empty position tag. We have run several computational experiments with the basic neural network in order to determine the best size of the window and the number of hidden units. It turns out that the best neural network's performance corresponds to the size of window between 13 and 17 and to the number of hidden units between 3 and 20 as it was reported previously [10]. The gain in performance in the above-mentioned boundaries is about 1% both for the size of the window and for the number of the hidden units and can not be considered as significant. So, we fix the number of hidden units equal to 3 and the size of the window equal to 13 for further experiments, which makes us a $273-3-3$ neural network's architecture. Using the Sejnowski's data set we obtained the 62%-performance (see Fig 1) with the basic neural network, which is the same as reported previously [10]. However, the basic neural network is quite badly

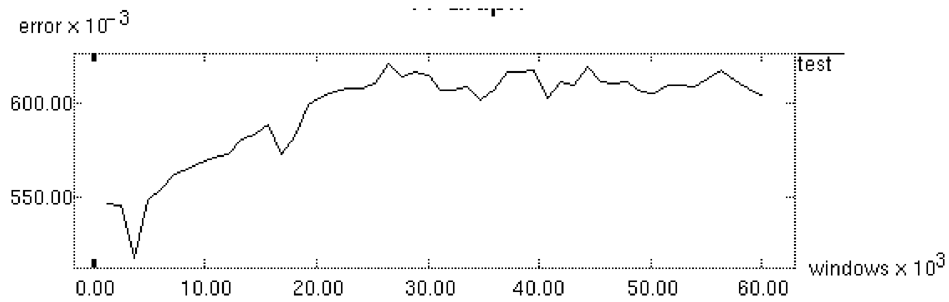

Fig. 1. Error evolution for the basic $273-3-3$ neural network trained on the Sejnowski's data set: learning rate $\eta = 0.2$ momentum $\alpha = 0.1$

adapted to the problem of the secondary structure prediction as it involves very little knowledge on the problem, except the size of the window: it is the first trivial architecture that one can try.

There are the difficulties in defining what really a secondary structure is. Depending on the considered properties of a folded protein, the location and extent of a secondary structure element can vary. Three most widely used assignment methods are the DSSP program, Define program and P-Curve program. We have used a low-homology ameliorated data set provided by biologists [2] and annotated with all three secondary structure assignment along with a 10-states consensus assignment constructed on their basis. Playing with this data set we have tried to achieve a better understanding of the nature of the problem. Lacking of space we do not present here the detailed description of our experiments. To summarize the results, we have to mention three important facts about the so-posed secondary structure prediction problem:

1. There is an important inherent noise due to the bad secondary structure

description. Such simple methods of noise cutting as:

  – increasing the size of the training set with the simultaneous decreasing its homology,
  – biologically and statistically plausible change of the input window's geometry,
  – using a simple re-description the secondary structure as the consensus combination of three known secondary structure assignment [2],
  – learning just on confident part of the data set where all three algorithm assign the same type of secondary structure,
  – localizing the noise by comparison of neural network's errors with all three available secondary structure assignment

   do not ameliorate the neural network's performance. It means that there is a tricky noise in the data set, which can not be cut by some statistical method. This crucial problem implies the necessity to search for a new less-noisy secondary structure representation developed from a new biological information.

2. The secondary structure context should be taken into consideration, i.e. training the neural network to predict rather the regions of the secondary structure than just the conformation of the center amino acid.
3. A pertinent metric in the input space should be determined to adjust appropriately the neural network's capacity for the best learning/generalization behavior;

The observations 2 and 3 allow to achieve a non-negligible performance gain (several percents order) even using the available high-noisy data set. In the next section we describe the method for a decision scheme extraction from a trained neural network inspired by [12], which we apply to the neural network trained to predict the protein secondary structure.

3 A Constrained Basic Neural Network

Extracting knowledge from an arbitrary neural network is recognized as a complex task. One possible approach to the extraction problem is to constrain appropriately a neural network making it interpretable in humanly comprehensible terms. Naturally, neural network learning algorithm should be modified with respect to the imposed constraints.
We impose the constraint on the basic neural network's weights to be in the finite set of values $\{-1, 0, 1\}$.
The working of the so-constrained basic neural network can be approximately interpreted as the following majority vote decision scheme:

– Let I_α, I_β, I_γ be the counters calculating the contribution of each input unit (i.e. the contribution of an amino acid at a given place) into the final decision of the neural network on the type of secondary structure (helix, sheet, coil respectively).

– Let path p^i_{nm} be a conjunction of two non-zeros weighted connections, where the first one comes from n-th input unit to m-th hidden unit and the second one comes from m-th hidden unit to i-th output unit ($i \in \{\alpha, \beta, \gamma\}$). We call p^i_{nm} a positive (negative) path if the product of its weights is positive (negative), we denote a positive (negative) path as p^{i+}_{nm} (p^{i-}_{nm}).
– So, the working of the binary-weighted neural network can be approximated as:
  - calculate the counters: $\mathrm{I}_i = \sum_{p^{i+}_{nm}} 1 - \sum_{p^{i-}_{nm}} 1$, $\mathrm{i} \in \{\alpha, \beta, \gamma\}$
    This means that the presence of each amino acid having positive (negative) path to i-unit, $i \in \{\alpha, \beta, \gamma\}$ will cause the increasing (decreasing) of the I_i counter and encourage (discourage) the neural network to predict the i-conformation. We say that such an amino acid "votes" for (against) the i-conformation.
  - compare counters and make a decision:
    if $I_i = \max \{I_\alpha,\ I_\beta,\ I_\gamma\}$ then the center amino acid conformation is predicted as i,
    i.e., neural network counts all for-i-conformation voices and all against-i-conformation voices of the amino acids within the window and make a "democratic" decision on the conformation of the center amino acid.

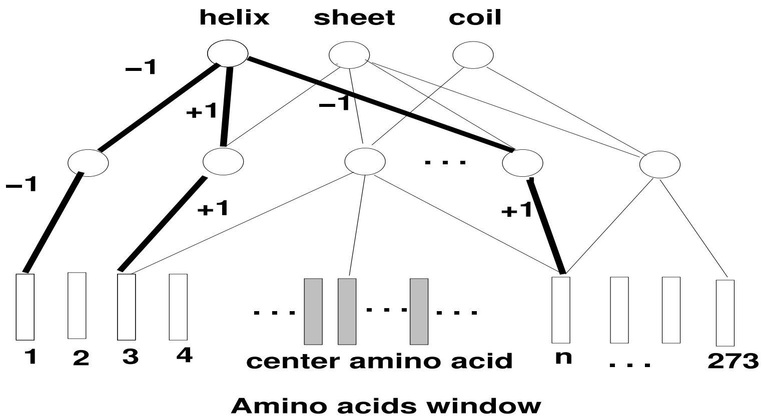

Fig. 2. Extracting a majority vote decision scheme from a constrained neural network. Sticks present non-zero connections after training. Thick sticks present all helix-contributing paths.

Example 1. A state of a constrained neural network after training is shown in Fig 2. The k-th bit codes the presence of an amino acid represented as $mod(k, 21)$-th bit at the $\lfloor \frac{k}{21} \rfloor$-th place of the input window. The presence of an acid coded by the first or the third bit at the first place of the input window will lead to vote for the α-conformation, while the presence of the amino acid $mod(n, 21)$ at the place $\lfloor \frac{n}{21} \rfloor$ (the fact coding by the n-th bit of the input window) will lead to vote against the α-conformation. In the same manner all bits of the input window connected by positive (negative) path with a particular output unit are interpreted as the presence of an amino acid AA at a place k, voting for (against) the particular conformation. All votes are counted and final majority vote decision is made.

The majority vote scheme approximation explicits the relationship between the inputs and the outputs of a neural network. This approximation neglecting the threshold values was proved to be good-working for the secondary structure prediction problem (see below) partly due to the calculating of the sigmoid function by our neural network simulator ($f(x) = 1, x < 0.8$ and $f(x) = -1, x > -0.8$), partly due to the nature of the mapping under approximation. However, this approximation should be studied in more details to determine its degree of generality.
The main problem for extracting decision scheme using the proposed constrained-based method is to modify a standard neural network learning algorithm to keep the neural network's weights into a finite set of values $\{-1, 0, 1\}$. We have found that the following modification of the backpropagation algorithm works reasonably well for simple boolean problems [12]. We penalize any unwanted weight changes during training by adding an extra term $P(W)$ to the standard error function $E(W)$. The penalty term $P(W)$ will force weights to approach -1 or 0 or 1 during training. Thus, the function to minimize using the standard gradient descent method becomes $E(W) + \lambda P(W)$, where λ is an adjustable parameter to get the best convergence.

4 Convergence Technique

In practice, it was appeared to be non-trivial to achieve a satisfactory convergence with the method described above for our constraints-based $273 - 3 - 3$ neural network. The subtle trade-off between the backpropagation and the constraint brought us to the development of the convergence method explained below.
First, we had to increase the number of hidden units up to 10 to compensate the decreasing of the neural network's capacity by constraints imposition.
Second, we put the constraints dynamically, so that the penalty term $P(W)$ is represented as:

$$P(W, t) = P(W_1, t) + k(t)P(W_2, t)$$

where W_1 is the set of weights between the input and the hidden layers, W_2 is a set of weights between the hidden and the output layers, t is a number of window presentations, the shape of the $P(W_i, t_0)$, $i = 1, 2$ is presented in Fig

3. Parameter $k(t)$ is introduced to compensate the low capacity of the hidden-output mapping in comparison with the input-hidden mapping, it is adjusted during training as it is shown in the Table 1. The penalty term $P(W,t)$ is a differentiable juxtaposition of ten paraboles changed during training with the help of the set of convergence parameters $a(t), b(t), c(t)$, whose dynamic is shown in Fig 4.

$$
\text{P(x,t)=} \begin{cases}
0.1(x+1)^2 & x < 1 \\
d_1(t)(x+1)^2 & -1 \leq x < -c(t) \\
E(t) - c_1(t)(x+b(t))^2 & -c(t) \leq x < -b(t) \\
E(t) - b_1(t)(x+b(t))^2 & -b(t) \leq x < -a(t) \\
a_1(t)x^2 + E_0(t) & -a(t) \leq x \leq a(t) \\
E(t) - b_1(t)(x-b(t))^2 & a(t) < x \leq b(t) \\
E(t) - c_1(t)(x-b(t))^2 & b(t) < x \leq c(t) \\
d_1(t)(x-1)^2 & c(t) < x \leq 1 \\
0.1(x-1)^2 & x > 1
\end{cases}
$$

Where

$$a_1(t) = \frac{E(t) - E_0(t)}{a(t)b(t)}$$

$$b_1(t) = \frac{E(t) - E_0(t)}{b(t)(b(t) - a(t))}$$

$$c_1(t) = \frac{E(t)}{(c(t) - b(t))(1 - b(t))}$$

$$d_1(t) = \frac{E(t)}{(1 - c(t))(1 - b(t))}$$

Our convergence method consists of three steps.

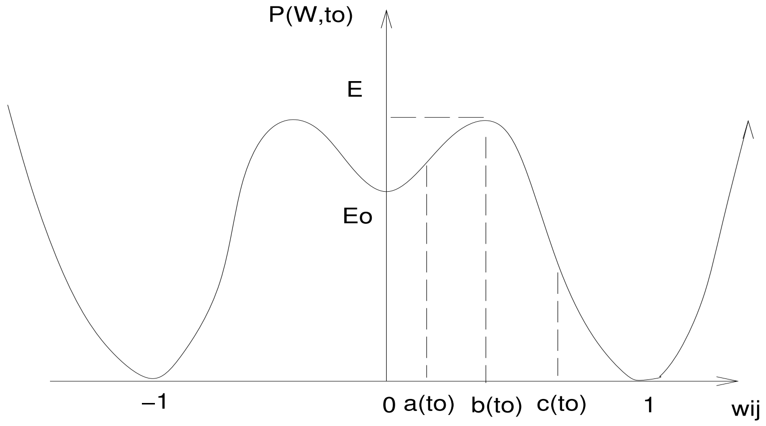

Fig. 3. Instantaneous penalty term shape $P(W,t)$, $t = t_0$ $a(t_0), b(t_0), c(t_0)$ are the transition points between the paraboles

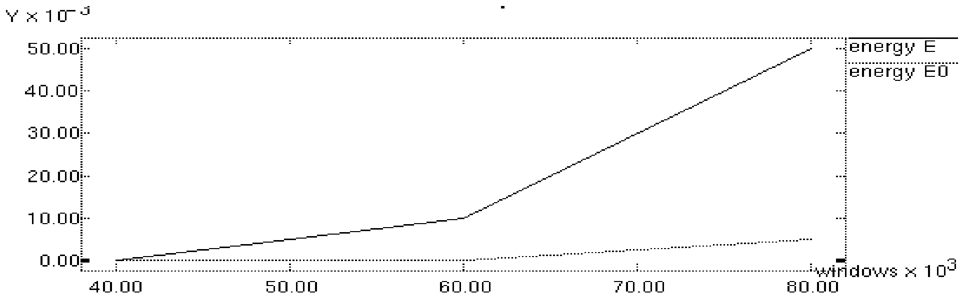

Fig. 4. Evolution of the penalty term's energy parameters parameters E and E_0 during training

Table 1. Linear dynamic of the convergence parameters

Convergence parameters	Number of window presentation, t		
	40000	60000	80000
$a(t)$	0.2	0.1	0.01
$b(t)$	0.3	0.15	0.6
$c(t)$	0.4	0.3	0.99
$k(t)$	10	30	40
$E(t)$	0.00005	0.01	0.05
$E_0(t)$	0.000005	0.0001	0.005

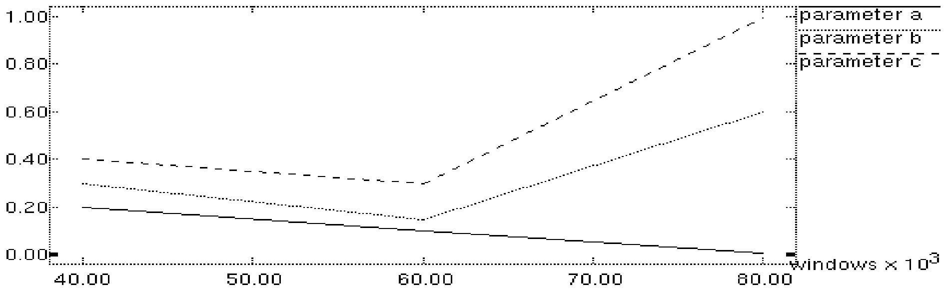

Fig. 5. Evolution of convergence parameters $a(t), b(t), c(t)$ during training

- *Step 1: usual learning phase*
  Each example is presented at the entrance of the neural network and the error is back-propagated within. We stop the convergence when the test rate reaches 60%.
- *Step 2: gradual constraints introduction*
  Without stopping the training we gradually add the penalty (see Fig 3) to each weight of the network. The value of the penalty term around zero is not null (E_0 in Fig 3) because we wish to reduce the number of the zero-valued weights in the final network, as it was checked that the large amount of the zero weights prevents the neural network to reach the final easily-explicitable configuration. So, we "help" the weights with outwards-of-zero tendency to jump over the penalty term barrier (denoted as E in Fig 3).

  Slowly we increase the barrier to push each weight into a cluster around $-1, 0$ or $+1$. At this time the learning and the constraining are opposed but the weights of particular interest (i.e. connections bringing a discriminating information) are kept far from zero and the other weights (without statistical significance) are gradually reduced to 0. This resistance is related to the amount of the discriminant information brought by a particular connection. If the penalty term shape is "smart" enough – not too strong at the beginning, not too wide, with a correct rate – one can see that a neural network under this constraint keeps its correctness of prediction even if its weights belong to three clusters around $-1, 0$ or $+1$.

  The introduction of the constraint must be gradual. Experimentally we found that a good way of constraining a network is to "push" its weights towards zero. If these weights are significant (they are useful for the prediction) they will "resist" and overcome the "smart" constraint. To do so we progressively increase the value of the penalty. First, we put a small penalty error for the weights near 0.3 then we increase the value of the error but move the position of the maximum to 0.1. So the significant weights can "jump" the maximum penalty error. They are pushed towards zero for a while (when their value is under the decreasing position of the maximum of the penalty error) but if they can resist to the constraint they will be pushed towards +1 or −1 when their value will become over the position of the maximum of the penalty.

– *Step 3: final transformation*
  Keeping on with the backpropagation training we increase the constraint penalty and so reduce the size of the clusters. At the end of this short period the values of the weights are as close as wanted to $-1, 0$ or $+1$.

This method was applied to train the constrained $273 - 10 - 3$ neural network to predict the protein secondary structure. We have constrained the "average" neural network with 62%-performance (our "best" neural network has a 64%-performance). The constrained neural network achieved about 61.2%-performance, so we do not loose the performance with the constraints introduction, that is extremely important. In the next section we present a resulting majority vote scheme extracted from the so-constrained neural network.

5 Extracting A Decision Scheme

Having applied our method for extracting a decision scheme to the constrained $273-10-3$ neural network trained using the described above convergence technique, we have obtained the following vote scheme summarized in the Tables 2-4. For each secondary structure type the positive/negative votes of a given amino acid at a given place are presented. One can easily see some interesting partic-

Table 2. Amino acids increasing (AA) and decreasing (!AA) the helix counter

Residue position	Amino acids																			
1		!C		E																
2		!C		E																
3	A			E		!G							!P							Y
4	A			E		!G	!H						!P							
5	A			E		!G				L	M		!P			!S				
6	A			E		!G				L	M		!P			!S	!T			
7	A			E	F	!G				L	M	!N	!P			!S	!T			
8	A			E	F	!G			K	L	M		!P	Q	R					
9	A		!D		F		H		K	L	M		!P			!S	!T			
10	A		!D		F	!G	H		K	L	M		!P			!S				
11	A		!D				H		K	L	M		!P							
12	A						H		K	L	M		!P							
13		!C	!D	!E		!G			K				!P			!S	!T			

ularities in the extracted decision scheme. For example, as it follows from the tables, some amino acids are check by the neural network on several positions

Table 3. Amino acids increasing (AA) and decreasing (!AA) the sheet counter

Residue position	Amino acids																			
1					!F	G										S				
2		!C				G				!L	!M	N				S			!W	
3						G				!L						S				
4		!C	!D	!E		G				!L										
5	!A		!D	!E			!H	I	!K			!N	!P					V		Y
6	!A		!D	!E	F	!G	!H	I	!K	L		!N	!P				T	V	W	Y
7	!A		!D	!E	F	!G		I	!K	L		!N	!P					V	W	Y
8	!A	C	!D	!E	F			I	!K	L			!P	!Q				V	W	Y
9				!E	F			I	!K				!P	!Q				V		
10									!K					!Q		S				
11						G			!K	!L						S				
12						G		!I	!K	!L						S				
13								!I	!K	!L	!M					S	T			

Table 4. Amino acids increasing (AA) and decreasing (!AA) the coil counter

Residue position	Amino acids																			
1																				
2		C																		
3																!S				
4											!M		P							
5			D			G	H	!I	K	!L	!M	N	P							
6			D	!E	!F	G		!I	K	!L	!M	N	P			S		!V		
7			D		!F	G		!I		!L	!M	N	P			S		!V	!W	!Y
8	!A	!C	D		!F	G		!I		!L	!M		P					!V	!W	!Y
9			D		!F			!I		!L			P					!V		
10			D				!H			!L	!M		P							
11	!A												P							
12													P							
13									!K											

within the window. In Table 2. the presence of glutamic acid (E) is detected from position 1 to position 8 to predict helix and is detected (see Table 3) from position 4 to position 9 to predict negatively sheet. This shows the positional influence of amino acid on the prediction. The extracted decision scheme should be investigated in more details and carefully compared with the existing biological knowledge on the secondary structure prediction, but the first results on its analysis is given in the next section.

6 Analysis of The Extracted Decision Scheme

While the constrained neural network does almost as good as the non-constrained one (62.3% against 61.4%), the extracted vote scheme does as good as the constrained neural network (being tested on the Sejnowski's test set they give different predictions only on 126 examples from 3520). The performance comparison between all three "devices" for the secondary structure prediction is summarized in the Table 5, where C_α, C_β C_γ are the Mathews coefficients. The correlation Mathews coefficient is defined by the formula:

$$C_s = \frac{TP_sTN_s - FP_sFN_s}{\sqrt{(TP_s+FN_s)(TN_s+FP_s)(TN_s+FN_s)}}$$

where s is the secondary structure whose coefficient C_s is being calculated, and TP_s, TN_s, FP_s and FN_s are the number of true positives, true negatives, false positives and false negatives for that structure respectively. This correlation coefficient is usually used to evaluate the effectiveness of the prediction for each of the classes of secondary structure separately. It is interesting to note, that the vote scheme is "faithful" to the constrained neural network, and both of them overperform the non-constrained neural network for the helix prediction. We

Table 5. Estimation of performance on the secondary structure prediction

	constrained neural network	vote scheme	non-constrained neural network
Correct helix predictions	10.6	10.6	11.6
Uncorrect helix predictions	11.4	11.4	12.3
Correct sheet predictions	6.6	7.4	5.8
Uncorrect sheet predictions	7.0	8.3	4.9
Correct coil predictions	44.1	43.7	44.8
Uncorrect coil predictions	20.3	19.1	20.3
Total correct	61.4	61.2	62.3
Total uncorrect	38.4	38.8	37.7
C_α	0.43	0.42	0.34
C_β	0.39	0.40	0.43
C_γ	0.32	0.32	0.34

have roughly compared the amino acids prediction tendencies with respect to a particular secondary structure discovered by our constrained neural network with their tendencies estimated by the GOR and the Chou-Fasman theories. Table 6 summarizes the favorable/unfavorable amino acid votes, obtained from the vote scheme, for all three secondary structure types. As it follows from the Table 6 we have retrieved almost the same predictive behavior of the amino acids as it's described by the Chou-Fasman and the GOR theories (see Table 7).

Being applied to the test examples, the extracted vote scheme has an interesting property to increase the amount of useful information as it points out the discriminant example features. Table 8 shows the trace of the vote scheme application on the test example. It is easy to see that only 6 amino acids from 13

Table 6. Summary of the amino acids prediction tendencies discovered by constrained neural network

Amino Acid	helix		sheet		coil	
	favorable	unfavorable	favorable	unfavorable	favorable	unfavorable
Ala	10	0	0	4	0	2
Cys	0	3	1	2	1	1
Asp	0	4	0	5	6	0
Glu	8	1	0	6	0	1
Phe	4	0	4	1	0	4
Gly	0	8	6	2	4	0
His	4	1	0	2	1	1
Ile	0	0	5	2	0	5
Lys	6	0	0	9	2	1
Leu	8	0	3	6	0	6
Met	8	0	0	2	0	6
Asn	0	1	1	3	3	0
Pro	0	11	0	5	9	0
Gln	1	0	0	3	0	0
Arg	1	0	0	0	0	0
Ser	0	6	7	0	2	1
Thr	0	4	2	0	0	0
Val	0	0	5	0	0	4
Trp	0	0	3	1	0	2
Tyr	0	1	4	0	0	2

Table 7. Amino acids prediction tendencies accepted in the Chou-Fasman and the GOR theories

Prediction theory	favorable to helix	favorable to sheet	favorable to coil
GOR	Ala, Glu, Leu	Val, Ile, Tyr, Trp, Cys	Pro, Gly, Asn, Asp, Ser
Chou -Fasman	Glu, Ala, Leu	Met, Val, Ile	Pro, Gly, Glu
Vote Scheme	Ala, Glu, Met, His	Ile, Val, Tyr	Pro, Asp, Asn

available are checked by the neural network to predict the conformation of the center amino acid. It means that the constrained neural network points out the example features useful for prediction, and being applied to examples, gives a "re-description" of the initial example set within a larger amount of discriminant information. However, the information brought by the neural network should be more carefully measured.

Finally, the vote scheme can be roughly refined as its application to the test examples reveals "uncorrect" votes (see Table 8), which can be removed. We have scored each vote on the test set. The votes that are mostly uncorrect have been removed, the obtained slightly refined vote scheme overperforms the original one by 0.2%. The vote scheme can be refined more as it's indicated in the next section.

Table 8. The trace of the vote scheme on test example AIDSLAASGAKGF – helix

number of the vote activated on example	amino acid AA on position k ($AA@k$) leading to the vote activation	the activated vote		Correctness of the vote
		votes for + against –	coil (_) helix (h) sheet (e)	
38	L@4	–	_	yes
131	A@5	–	e	yes
146	A@6	–	e	yes
183	K@10	–	e	yes
186	G@11	+	e	no
214	L@4	+	h	yes
218	A@5	+	h	yes
226	A@6	+	h	yes
256	A@9	+	h	yes
269	K@10	+	h	yes

7 Discussion and Future work

We have succeed in expliciting the work of a neural network trained to predict protein secondary structure. The neural network's work was expressed as a clear decision scheme without loosing performance. This is an important result both for the protein secondary structure prediction and for the understanding of the neural network's behavior, even if it's generality is not tested yet. The extracted decision scheme predicts the conformation of the center amino acid on the basis of presence of particular amino acids on particular positions in its around. The prediction accuracy of the extracted scheme is 61.2%, that is better than both the Chou-Fasman and the GOR predictions. Moreover, the extracted prediction scheme can be refined. Our constraints-based method:

- explicits the neural network's work as a clear decision scheme;
- increases the information containing in the set of examples (Table 8), that is an important step to constructing a good-working generalization;
- facilitates the secondary structure pattern integration, which is an interesting property as the known prediction theories are often formulated in some pattern-based language [5] [6];
- presents itself an efficient heuristic for the further symbolic generalization;
- facilitates the refinement of the neural network's prediction method as symbolic refinement methods become available;

Our future work will be both to ameliorate the quality of prediction and to obtain humanly comprehensible, explicit relationships between protein primary and secondary structures.

- *The quality of prediction* (closely related to the neural network behavior understanding)
  - The performance of the basic neural network can be ameliorated by

  * taking into account the secondary structure context,
  * understanding of the primary-secondary structure mapping in formal terms,
  * re-representation of the training set in a less noisy manner;
- So, it is expected to be possible to obtain a basic neural networkof better performance. The question is could we achieve the almost-the-same convergence constraining a more neural network of better performance or there exists a limit for the performance of the so-constrained neural network?
- Our input coding takes into account the presence of the amino acids in the chosen window, but not the absence of some of them. Could the absence of some amino acid be a discriminating feature for the secondary structure prediction? A priori, we would have a tendency to answer yes, as, for example, proline absence is recognized to be discriminate. So, some gain in performance is probably achievable by expliciting the "absence" information for the neural network.
- The vote scheme should be refined more using available symbolic techniques.
- Try to learn recursively using the "information increasing" property of the constrained neural network, generalizing each time on a new set of examples issue of application of decision scheme on the initial training set;

– *How to obtain the comprehensible prediction rules?*
  - To obtain the comprehensible prediction rules rather the secondary structure region prediction should be predicted. So, we need either to train our neural network to learn the structural regions or to transform the center amino acid predictions to the region predictions after training;
  - The vote scheme applied to the set of secondary structure examples gives a new set of examples containing more information than the initial one as the descriptors of examples judged non-important by the neural network are omitted. This new set of examples can be generalized by symbolic methods, using the vote scheme as a searching heuristic.
  - From the viewpoint of the knowledge insertion, it's interesting to have the truth-preserving transformation from a neural network to a decision scheme as it facilitates considerably the insertion of pattern-based knowledge (like as Lim's and Cohen's patterns [5] [6]).
  - Change of the searching space by a biologically-interesting one. We can insert different a priori knowledge into neural network (the insertion of biologically-interesting example descriptors results in a new neural network's input coding, while the insertion of known relationships results in a new neural network's topology) and extract a neural network's decision scheme after training. Such a scheme will be consistent with the generally admitted knowledge about protein secondary structure and can reveal new biologically-interesting dependencies. The first attempt to use an a priori knowledge, namely, the Chou-Fasman theory for the neural

network-based secondary structure prediction made in [8] gave the statistically significant gain in the neural network's prediction performance. As the Chou-Fasman theory is of statistical nature, its insertion does not provide a principally new information in comparison with the neural network-based technique, it just helps to place a neural network about the interesting minima.

Acknowledgement We wish to thank Nathalie Colloc'h for supplying us the secondary structure data assigned by three different methods along with consensus assignment as well as for her useful discussions.

References

1. Anfinsen, C.B. et al. (1961) Proc. Natl. Acad. Sci. USA, 47, 1309-14.
2. Colloc'h N., et al.(1992) Comparison of three algorithms for the assignment of secondary structure in proteins: the advantages of a consensus assignment, submitted to Protein Engineering.
3. Chou, P.Y., Fasman, G.D. (1978) Advan. Enzymol., 47, pp.45-148
4. Gibrat, J.F. et al (1987). J. Mol. Biol. 198, 425-443.
5. Lim, V.I. (1974) J. Mol. Biol., 88, pp.873-894
6. Cohen, f.e. et al (1986). Biochemistry 25, 266-275.
7. King R.D., Sternberg M.J. (1990) J. Mol. Biol. 216, 441-457.
8. Maclin R., Shavlik J.W. (1991) Refining Algorithms with Knowledge-based Neural Networks: Improving the Chou-Fasman Algorithm for Protein Folding, Machine Learning Research Group Working Paper, CS department, University of Wisconsin, Madison, available via anonymous ftp from steves.cs.wisc.edu
9. Muggleton S., King R.D., Sternberg M.J. (1992), Protein Engineering, Vol. 5, no 7.
10. Qian N., Sejnowski T. (1988) Predicting the secondary structure of globular proteins using neural network models, J. Mol. Biol., 202, pp.865-884
11. Towell, G.G., Shavlik J. (1992) Interpretation of artificial neural networks: mapping knowledge-based neural networks into rules, in NIPS, Vol.4, pp.977-984
12. Kane R., I. Tchoumatchenko, M.Milgram. (1992) Extracting knowledge from data using constrained neural networks, to appear in the Proc. of the European Conference on Machine Learning, Austria, Vienne, 5-8 Apr, 1992.