Secondary Structure Prediction Using ANN Learning

Dr. Pankaj Agarwal

Professor & Head, Department of Computer Science & Engineering, IMS Engineering College, Ghaziabad

Sakshi Jain, Deepika, Swati Verma

Students of B.Tech (CSE) 2005-09 batch
IMS Engineering College, Ghaziabad

Abstract: *This paper presents two methods for predicting the secondary structure of proteins based on artificial neural network learning. Two variations of NN learning rule are employed using feedforward backpropogation architecture for predicting secondary structure of proteins from their promary sequences of amino acids. About 500 sequences and more than 10000 patterns were trained with variable size of patterns. After initial level of training, an accuracy rate of about 70%-75% is obtained through first learning rule wheras 80-85% of accuracy is obtained using the second variation of the learning rule. Both the methods are implemented within a software tool by the name NNSec developed on Visual.NET flatform.*

I. INTRODUCTION

Protein structure determination and prediction has been a focal research subject in the field of bioinformatics due to the importance of protein structure in understanding the biological and chemical activities of organisms. The experimental methods used by biotechnologists to determine the structures of proteins demand sophisticated equipment and time.

Secondary structure prediction is a set of techniques in bioinformatics that aim to predict the secondary structures of proteins and nucleic acid sequences based only on knowledge of their primary structure. For proteins, this means predicting the formation of protein structures such as alpha helices and beta strands, while for nucleic acids it means predicting the formation of nucleic acid structures like helixes and stem-loop structures through base pairing and base stacking interactions [12].

If the secondary structure of a protein is known, it is possible to derive a comparatively small number of possible tertiary (three-dimensional) structures using knowledge about the ways that secondary structural elements pack.

The best modern methods of secondary structure prediction in proteins reach about 80% accuracy; this high accuracy allows the use of the predictions in fold recognition and ab initio protein structure prediction, classification of structural motifs, and refinement of sequence alignments. The accuracy of current protein secondary structure prediction methods is assessed in weekly benchmarks such as LiveBench and EVA. [13]

A number of factors exist that make protein structure prediction a very difficult task, including:

- The number of possible structures that proteins may possess is extremely large.
- The physical basis of protein structural stability is not fully understood.
- The tertiary structure of a native protein may not be readily formed without the aid of trans-acting factors. For example, proteins known as chaperones are required for some proteins to properly fold; other proteins cannot fold properly without modifications such as glycosylation.
- A particular sequence may be able to assume multiple conformations depending on its environment, and the biologically active conformation may not be the most thermodynamically favorable.

Due to exponentially improving **computer power**, and **new algorithms**, much progress is being made to overcome these factors by the many research groups that are interested in the task. Prediction of structures for small proteins is now a perfectly realistic goal. A wide range of approaches are routinely applied for such predictions. [6]

Neural networks have been found quite suitable for SS Prediction. Judging by the published results in the protein biochemistry literature, neural networks have produced the most accurate secondary structure predictions for the majority of the past decade, starting with [Qian & Sejnowski, 1988]. ANNs have immense computational abilities and are particularly good at prediction tasks. They are able to achieve a higher degree of accuracy than deterministic algorithms without the tradeoff of execution speed or increased processing power. The relationship between an amino acid sequence and the structure of the protein it forms is currently unknown. Researchers do not understand the folding process which causes this transformation and have termed this the protein folding problem. An artificial neural network (ANN) approach may be successful in solving the problem by implementing an ANN to predict protein structure from the amino acid sequence. There can be no doubt that ANNs are very effective at combining multiple non-linear factors, such as those affecting folding. They have been applied successfully to medical diagnosis, financial prediction, and face recognition. Given their success in other areas, it is very likely that with more research, neural nets will prove themselves worthy for protein structural prediction and will solve the protein folding problem. [13]

Feed-forward nets are the most well-known and widely used class of neural network. The popularity of feed-forward networks derives from the fact that they have been applied successfully to a wide range of information processing tasks in such diverse fields as speech recognition, financial prediction, image compression, medical diagnosis and *protein structure prediction*; new applications are being discovered all the time. In common with all neural networks, feed-forward networks are trained, rather than programmed, to carry out the chosen information processing tasks. Training a feed-forward net involves adjusting the network so that it is able to produce a specific output for each of a given set of input patterns. Since the desired inputs are known in advance, training a feed-forward net is an example of *supervised learning*.

II. RELATED STUDY

Methods for Single Sequences

Secondary structure prediction has been around for almost a quarter of a century. The early methods suffered from a lack of data. Predictions were performed on single sequences rather than families of homologous sequences, and there were relatively few known 3D structures from which to derive parameters. Probably the most famous early methods are those of Chou & Fasman [1], Garnier [3], Osguthorbe & Robson (GOR) and Lim [2]. Although the authors originally claimed quite high accuracies (70-80 %), under careful examination, the methods were shown to be only between 56 and 60% accurate (see Kabsch & Sander [4], 1984 given below). An early problem in secondary structure prediction had been the inclusion of structures used to derive parameters in the set of structures used to assess the accuracy of the method. See few early methods on Single sequences [1, 2, 3, 4]. Find some later methods on single sequences in the references [5, 6, 7, 8]

Recent Improvements

The availability of large families of homologous sequences revolutionized secondary structure prediction. Traditional methods, when applied to a family of proteins rather than a single sequence proved much more accurate at identifying core secondary structure elements. The combination of sequence data with sophisticated computing techniques such as neural networks has lead to accuracies well in excess of 70 %. Though this seems a small percentage increase, these predictions are actually much more useful than those for single sequence, since they tend to predict the core accurately. Moreover, the limit of 70-80% may be a function of secondary structure variation within homologous proteins. [9- 22]

Automated Methods

There are numerous automated methods for predicting secondary structure from multiply aligned protein sequences. Some good references on the subject include. [23-31].

Current Known Automated methods

- There are numerous automated methods for predicting secondary structure from multiply aligned protein sequences. Some good references on the subject include (the acronyms in parentheses given after each reference refer to the associated WWW servers, given below):
- Zvelebil, M.J.J.M., Barton, G.J., Taylor, W.R. & Sternberg, M.J.E. (1987). Prediction of Protein Secondary Structure and Active Sites Using the Alignment of Homologous Sequences *Journal of Molecular Biology*, 195, 957-961. (ZPRED)
- Rost, B. & Sander, C. (1993), Prediction of protein secondary structure at better than 70 % Accuracy, *Journal of Molecular Biology*, 232, 584-599. PHD)
- Salamov A.A. & Solovyev V.V. (1995), Prediction of protein secondary sturcture by combining nearest-neighbor algorithms and multiply sequence alignments. *Journal of Molecular Biology*, 247,1 (NNSSP)
- Geourjon, C. & Deleage, G. (1994), SOPM : a self optimised prediction method for protein secondary structure prediction. *Protein Engineering*, 7, 157-16. (SOPMA)
- Solovyev V.V. & Salamov A.A. (1994) Predicting alpha-helix and beta-strand segments of globular proteins. (1994) *Computer Applications in the Biosciences*,10,661-669. (SSP)
- Wako, H. & Blundell, T. L. (1994), Use of amino-acid environment-depdendent substitution tables and conformational propensities in structure prediction from aligned sequences of homologous proteins. 2. Secondary Structures, *Journal of Molecular Biology*, 238, 693-708.
- Mehta, P., Heringa, J. & Argos, P. (1995), A simple and fast approach to prediction of protein secondary structure from multiple aligned sequences with accuracy above 70 %. *Protein Science*, 4, 2517-2525. (SSPRED)
- King, R.D. & Sternberg, M.J.E. (1996) Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Protein Sci*,5, 2298-2310. (DSC).
- Nearly all of these now run via the World Wide Web. For individual details, see the papers for the individual methods, or click on the underlined acronyms given after most of the references given above (note that you can also run the methods by going to the appropriate WWW site).

III. PROPOSED METHODS

We have proposed two variations of ANN learning for Secondary Structure prediction of proteins. The general procedure employed is as follows:

- Our method takes the entire sequence as input followed by dividing the entire sequence into patterns of defined window size (say 5) from left to right order. The learning algorithm is them applied on the patterns which after traing are stored within the database of leaned patterns.
- Similarly when a novel sequence is given as input, it is broken into patterns of same window sizes in the same order.
- Results of each pattern i.e the corresponding secondary structure assignment is concatenated in the same order in which it was broken from the sequence to give the secondary structure of the entire sequence.
- Result of each sequence is then compared with the actual secondary structure assignment of the sequence and success of the method is analyzed in terms of calculated error % w.r.t the actual prediction.

Features

- We have applied the feedforward back propogation method with a new learning rule.
- Currently patterns of window size 5 & 11 have been trained. We are working on patterns of sizes 17 & 21 as well.
- Initial weights were randomly taken between the range [-1,1] in the first method and between the range [0,1] in the second method. It can also be set by the users.
- Expected values for E, H & C structures can be also defined by the users and the network will accordingly be trained.
- The entire working of the net i.e. weight updaion procedure in each step, fetching of the patterns equal to defined window size, backpropogation of errors to hidden layer has been simulated within the developed s/w tool so as to help the users understand the working procedure completely.
-

Learning Rule 1: Use of Feed forward Backpropogarion Network

1. Select input as primary and secondary sequences of proteins from the data set as a part of supervised learning where the input and output is already known.
2. Insert "-"and "*" as first and last character of primary sequence taken.
3. Initialize the training set for each character of 20 amino acids already present for ex.
   trainingset = { "G", "P", "D", "E", "A", "N", "Q", "S", "T", "R", "K", "H", "V", "I", "M", "C", "L", "F", "Y", "W" };

4. function for creating patterns of window size required

   Fun Create_Pattern(windowsize)
   Repeat next step until counter<primarylength-4
   Window = primary[counter]+primary[counter+1]+……primary[counter+windowsize-1]

5. Function for creating input matrix
   For explanation let us considering the following short amino acid sequence ATSLVFW. Window size of three will be considered. For the sequence --ATSLVFW--, the corresponding sliding windows are –AT, ATS, TSL, SLV, LVF, VFW, FW- .Corresponding to window |--AT|,
   The input will be represented a

	--	A	T
G			
P			
D			
E			
A		X	

N			
Q			
S			
T			X
K			
R			
H			
V			
I			
M			
C			
L			
F			
Y			
W			
--	X		

The above matrix will be reduced to the following matrix at the hidden layer.

	--	A	T
A		X	
T			X
--	X		

Fun Create_Input_Matrix()
Repeat next steps for all patterns to obtain their input matrix
Charatpattern[]=pattern.ToCharArray(0,windowsize) //charatpattern contains all thecharacters of pattern obtained from primary sequence

For each pattern repeat these steps 22 times to check for each char of training set with charatpattern

```
if( trainingset[]==charatpattern)
        inputmatrix[i]=1
else
        inputmatrix=0;
```

6. Moreover a sample weight matrix is assumed initially which can be further be modified to map already analyzed sequences to some value lying between -1 and 1 ex

$w_{11} = 1$ $w_{12} = 0$ $w_{13} = -1$ $w_{14} = 1$ $w_{15} = -1$
$w_{21} = -1$ $w_{22} = 1$ $w_{23} = 0$ $w_{24} = -1$ $w_{25} = 1$
$w_{31} = 1$ $w_{32} = -1$ $w_{33} = 1$ $w_{34} = 0$ $w_{35} = -1$
$w_{41} = -1$ $w_{42} = 1$ $w_{43} = -1$ $w_{44} = 1$ $w_{45} = 0$
$w_{51} = 0$ $w_{52} = -1$ $w_{53} = 1$ $w_{54} = -1$ $w_{55} = 1$

7. Calculate activation for the input matrix and weight matrix initialized by using the formula
   Finally the calculation of the output is

Output=
$w_{11} * 0 + w_{12} * 1 + w_{13} * 0 + w_{14} * 0 + w_{15} * 0 +$
$w_{21} * 0 + w_{22} * 0 + w_{23} * 1 + w_{24} * 0 + w_{25} * 0 +$
$w_{31} * 1 + w_{32} * 0 + w_{33} * 0 + w_{34} * 0 + w_{35} * 0 +$
$w_{41} * 0 + w_{42} * 0 + w_{43} * 0 + w_{44} * 1 + w_{45} * 0 +$
$w_{51} * 0 + w_{52} * 0 + w_{53} * 0 + w_{54} * 0 + w_{55} * 1$
= 1*0 + 0*1 + (-1*0) + 1*0 + (-1*0) +
(-1*0)+1*0+0*1+ (-1*0) + 1*0+
1*1 + (-1*0) + 1*0 + 0*0 + + (-1*0) +
+ (-1*0) +1*0+ (-1*0) +1*1+0*0+
0*0 + (-1*0) + 1*0 + (-1*0) +1*1
= 3

Activation=1/ (1+e^ (-output))
=1/ (1+e^ (-3))
=0.8

8. Initializing range values for C, H,E according to their existence in secondary sequence as

If Output > 0 and Output<=0.2 then
Assume α-helix represented by 'h'
Else if Output >0.2 and Output <=0.4 then
Assume β-sheet represented by 'e'
Else if Output > 0.4
Assume Coil represented by 'c'

9. Now the target is calculated as the difference in deviation from starting range or final range whichever is less

```
Func find_target(ACTIVATION,Range of C,H,E already initialized)
If(ACTIVATION<=upper range)// Check for all three ranges of C,H,E
        If(ACTIVATION>lower range)
                Calculated_output="char represented by above range"//( C,H,E)
                If((ACTIVATION- lower range)>( ACTIVATION-upper range ))
                        Target= lower range
                Else
                        Target=upper range
```

10. Now if our Calculated output is same as secondary sequence then we say weight matrix are already trained and so with no change they are saved in database and if it is not same we apply back propogation rule for training the weight matrices.

Func Back_Propogation()
Initialize variables as follows:-
N<- number of iterations;
learningrate = 0.5;
two bias values i.e. bias1=-0.2 and bias2=0.3;
hiddenwt=0.7;

Repeat next steps N (total number of iterations) times and stop when calculated_output becomes actual output and we obtain the trained weight matrix.
Hidden=output+bias1
Activation2=1/ (1+e^ (-Hidden))
Final= (hiddenwt* Activation2)+bias2
FinalActivation=1/ (1+e^ (-Final))
Err_at_outputlayer= FinalActivation*(1- FinalActivation)*(target- FinalActivation)
// the error is propagated backward by updating the weights and bias for a unit j in the output layer .The output layer error Errj is computed as -: Errj=Oj*(1-Oj)*(Tj-Oj)
Err_at_hidden_layer= Activation2*(1- Activation2)*(Err_at_outputlayer* hiddenwt)
//Errk=Oj*(1-Oj)*(Errj*weight)

hiddenwt = hiddenwt + (learningrate * Err_at_outputlayer * output);
//hidden weight updation

wt=wt+(learning rate *Errj*Oj)
//weight matrix is also updated as

Weight[,]=Weight[,]+(learningrate * Err_at_hidden_layer * inputmatrix[,]);
Bias1 = Bias1 + (learningrate * Err_at_outputlayer);
//bias updation biasj=biasj+(learning rate*Oj)

Bias2 = Bias2 + (learningrate * Err_at_hidden_layer);
//again calculate activation value for new updated weight matrix to check output lies in the same range
New_output=summation(weight[]*input[])
New_Activation=1/ (1+e^ (-New_output))
//new activation of new output calculated from updated weight matrices
if (New_Activation <= upper range)
if (New_Activation > lower range)
ou = "(C,H,E)";//whichever range is taken
if (ou == char at secondary)
trainedweight = weight;
break;
// when output calculated matches the secondary output then iteration stops and weights are trained and are saved into database

Results Obtained

- Currently about 3434 patterns of size 11 and 3550 patterns of size 5 have been trained & evaluated. As expected patterns of size 11 have given more accurate results when compared with patterns of size 5 during prediction. This is in accordance with biological principle where it is known that secondary structure prediction of a residue depends on its neighboring residues. Longer patterns provide higher degrees of possibilities for accurate training & prediction. However based on the priliminary experimental studies on patterns of sizes say 21 it has been observed that degree of prediction accuracy degrades quite significantly. This clearly implies that their has to be an upper limit on the length of patterns to be trained corresponding to which prediction should be on higher side.

- On an average 1100 iterations were recorded for training patterns of size 11 and 789 iterations on average were recorded for patterns of size 5.
- An accuracy of 71% has been obtained while prediction of un-trained sequences when patterns of size 11 were trained. Wereas an accuracy of 67% has been obtained while prediction of non trained sequences when patterns of size 5 were trained

Learning Rule 2: Feedforward Backpropogation Network using Delta Rule

Training algorithm

1. Select input as primary and secondary sequences of proteins from the data set as a part of supervised learning where the input and output is already known.
2. Initialize the training set and the values for each character of 20 amino acids already present for ex.
   trainingset = { "G", "P", "D", "E", "A", "N", "Q", "S", "T", "R", "K", "H", "V", "I", "M", "C", "L", "F", "Y", "W" };
   trainingsetval = { 0.3, 0.6, 0.9, 0.12, 0.15, 0.18, 0.21, 0.24, 0.27, 0.33, 0.36, 0.39, 0.42, 0.45, 0.48, 0.51, 0.54, 0.57, 0.63, 0.66 };
3. Function for creating patterns of window size required

   Fun Create_Pattern(windowsize)
   Repeat next step until counter<primarylength-4
   Window = primary[counter]+primary[counter+1]+……primary[counter+windowsize-1]
4. Function for creating input matrix
   For explanation let us considering the following short amino acid sequence ATSLVFW. Window size of three will be considered. For the sequence --ATSLVFW--, the corresponding sliding windows are –AT, ATS, TSL, SLV, LVF, VFW, FW- .Corresponding to window |--AT|,

   The input will be represented as:

	--	A	T
G			
P			
D			
E			
A		X	

5.

N			
Q			
S			
T			X
K			
R			
H			
V			
I			
M			
C			
L			
F			
Y			
W			
--	X		

The above matrix will be reduced to the following matrix at the hidden layer.

	--	A	T
A		X	
T			X
--	X		

Fun Create_Input_Matrix()
Repeat next steps for all patterns to obtain their input matrix
Charatpattern[]=pattern.ToCharArray(0,windowsize) //charatpattern contains all thecharacters of pattern obtained from primary sequence
For each pattern repeat these steps 20 times to check for each char of training set with charatpattern
if(trainingset[]==charatpattern)
inputmatrix[]= trainingsetval[]
else
inputmatrix=1;

6. Initialize random weight matrix using random function

   Func Random_weights()
   Random rnd = new Random();
   Rand_wt=rnd.Next(0,1);//initializes weight randomly that has values lying between 0and 1

7. Calculate target Matrix using Secondary sequences and initializing C=0.3,H=0.6 E=0.9 target values according to their existence in secondary sequence

   Repeat next steps for each character in the pattern
   if (secpatternseq == "C,H,E")
   target[] = value(//to which they are initialized) ;

8. Final Training to be done using delta rule by applying it for any number of iterations
   N<-number of iterations
   Learningrate=0.2
   Func Delta_rule()
   Repeat next steps N (total number of iterations) times and stop when calculated_errorvalue at N-1 iteration becomes equal to calculated_errorvalue at Nth consecutive iteration and we obtain the trained weight matrix.
   Counter=0;
   Repeat next steps for Counter<windowsize times(ex 5 in this case)

yin = weight[windowsize-5] * input2[Counter, windowsize-5] + weight[windowsize-4] * input2[Counter, windowsize-4] + weight[windowsize-3] * input2[Counter, windowsize-3] + weight[windowsize-2] * input2[Counter, windowsize-2] + weight[windowsize-1] * input2[Counter, windowsize-1];

t_yin = target[t] - yin;
//wt changes
Wt_change = Learningrate * t_yin * input2[Counter, t];
//t<-counter repeat this step for t<windowsize times
//new wts
Weights=Weights+Wt_changes //repeat this step for t<windowsize times
Err_value = Err_value + t_yin * t_yin;// error value
}

double diff = err - finalerr;//diff of error value at 2 consecutive iterations
if (diff >= 0 && diff <= 0.00000001)
break;

```
if (finalerr != err)
    finalerr = err;//change in error value
```

9. When there is a diffrence of 0.00000001 in the error value then stop the iteration and add the weight matrix ,iteration number and error value at which iteration stops into the database so that these patterns and their calculated weight matrices can be used for further prediction.

Results Obtained

- Currently total of 8656 patterns of size 5 are trained with an average iteration of 348 per pattern.
- More than 100 new sequences were tested for prediction. Accuracy of 75-80% is observed based on the initial level of observations.
- We are currently working on patterns of size 11 and 21.
- It is expected that increase in the patterns size should not effect the accuracy of the prediction.
- There is significant improvement in the number of iterations required to train the patterns when compared with first learning rule.
- Further improvement in the learning strategies should definetly increase the prediction accuracies.

IV.CONCLUSION & FUTURE WORK

The methods employed in the current paper are quite simple & effective in approach.Sufficient number of patterns were trained to improve the prediction accuracy. This paper has encouraged us to try various types of ANN learning for prediction of not only secondary strutures but also teriary structure from promary sequences of proteins. Further improvement in the learning strategies should definetly increase the prediction accuracies. ANN does show enough scope and possibility to further improve upon the accuracy rates of existing similar tools.

References

[1] Chou, P.Y. & Fasman, G.D. *Biochemistry*, 211-222 (1974).
[2] Lim, V.I. *Journal of Molecular Biology*, 857-872. (1974)
[3] Garnier, J., Osguthorpe, D. Robson, B. *Journal of Molecular Biology*, 97-20 (1978).
[4] Kabsch, W. & Sander, C. *FEBS Letters*, 179-182. (1983)
[5] Deleage, G. & Roux, B. *Protein Engineering*, 289-294. (1987)
[6] Presnell, S.R., Cohen, B.I. & Cohen, F.E. *Biochemistry*, 983-993 (1992).
[7] King, R. & Sternberg, M. J.E. *Journal of Molecular Biology*, 441-457 (1990).
[8] D. G. Kneller, F. E. Cohen & R. Langridge Improvements in Protein Secondary Structure Prediction by an Enhanced Neural Network, *Journal of Molecular Biology*, 171-182. (NNPRED) (1990)
[9] Zvelebil, M.J.J.M., Barton, G.J., Taylor, W.R. & Sternberg, M.J.E. Prediction of Protein Secondary Structure and Active Sites Using the Alignment of Homologous Sequences *Journal of Molecular Biology*, 957-961. (ZPRED) (1987)
[10] Rost, B. & Sander, C. Prediction of protein secondary structure at better than 70 % Accuracy, *Journal of Molecular Biology*, **232**, 584-599. (PHD) (1993)
[11] Chandonia, J. M. & Karplus, M. New methods for accurate prediction of protein secondary structure. *Proteins*, 35, 293-306 (1999)
[12] Karplus, K., Barrett, C., Cline, M., Diekhans, M., Grate, L. et al. Predicting protein structure using only sequence information. *Proteins*, S3, 121-125 (1999)
[13] Petersen, T. N., Lundegaard, C., Nielsen, M., Bohr, H., Bohr, J. et al. Prediction of protein secondary structure at 80% accuracy. *Proteins*, 41, 17-20 (2000)
[14] Hua S, Sun Z. A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach. J Mol Biol ; 308:397–407 (2001).
[15] Albrecht, M., Tosatto, S.C., Lengauer, T., Valle, G. Simple consensus procedures are effective and sufficient in secondary structure prediction. *Protein Eng.*, 16, 459–462 (2003).
[16] Heringa, J. Computational methods for protein secondary structure prediction using multiple sequence alignments. *Curr. Protein Pept. Sci.*, 1, 273–301 (2000).
[17] Jones, D.T. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292, 195–202 (1999).
[18] Karchin, R., Cline, M., Mandel-Gutfreund, Y., Karplus, K. (2003) Hidden Markov models that use predicted local structure for fold recogniti**on**: alphabets of backb**on**e geometry. *Proteins*, 51, 504–514[CrossRef][ISI][Medline].
[19] McGuffin, L.J. and Jones, D.T. (2003) Benchmarking secondary structure prediction for fold recognition. *Proteins*, 52, 166–175[CrossRef][ISI][Medline].
[20] Przybylski, D. and Rost, B. (2002) Alignments grow, secondary structure prediction improves. *Proteins***,** 46, 197–205[CrossRef][ISI][Medline].
[21] Rost, B. (2001) Review: protein secondary structure prediction c**on**tinues to rise. *J. Struct. Biol.*, 134, 204–218[ISI][Medline].
[22] Simossis, V.A. and Heringa, J. (2004) Integrating secondary structure prediction and multiple sequence alignment. *Curr. Protein Pept. Sci.*, **5**, 1–15[ISI][Medline
[23] Salamov A.A. & Solovyev V.V. Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiply sequence alignments. *Journal of Molecular Biology*, (NNSSP) (1995)

[24] Geourjon, C. & Deleage, G., SOPM: a self optimized prediction method for protein secondary structure prediction. *Protein Engineering*, **7**, 157-16. (1994)
[25] Solovyev V.V. & Salamov A.A. Predicting alpha helix and beta-strand segments of globular proteins. (1994) *Computer Applications in the Biosciences*, 661-669. (1994).
[26] Wako, H. & Blundell, T. L. Use of amino-acid environment-dependent substitution tables and conformational propensities in structure prediction from aligned sequences of homologous proteins. *Journal of Molecular Biology*, 238, 693-708 (1994).
[27] Mehta, P., Heringa, J. & Argos, P., A simple and fast approach to prediction of protein secondary structure from multiple aligned sequences with accuracy above 70 %. *Protein Science*, 4, 2517-2525 (1995).
[28] King, R.D. & Sternberg, M.J.E. Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Protein Sci*, 5, 2298-2310 (1996).
[29] Koh, I.Y., Eyrich, V.A., Marti-Renom, M.A., Przybylski, D., Madhusudhan, M.S., Eswar, N., Grana, O., Pazos, F., Valencia, A., Sali, A., Rost, B. EVA: evaluation of protein structure prediction servers. *Nucleic Acids Res.*, 31, 3311–3315 (2003)
[30] Jones, D.T. and Swindells, M.B. Getting the most from PSI-BLAST. *Trends Biochem Sci.*, 27, 161–164(2002).
[31] Heringa, J. (2000) Computational methods for protein secondary structure prediction using multiple sequence alignments. *Curr. Protein Pept. Sci.*, 1, 273–301(2000).