Incorporating knowledge in multi-layer networks:
the example of proteins secondary structure prediction [1]

C. Mejía, F. Fogelman Soulié

Laboratoire de Recherche en Informatique
Bâtiment 490
Université de Paris Sud
91 405 ORSAY cedex, France.

Abstract

We present on the example of protein secondary structure prediction various ways by which domain specific knowledge can be incorporated into a multi-layer network so as to increase speed of learning and accuracy in prediction. In particular, we show how to set weight patterns so as to reproduce knowledge of the domain. We illustrate the use of linear learning and Discriminant Analysis to specify the appropriate number of hidden units. Finally, we show that "guided" pruning can improve accuracy while reducing the number of weights. These various techniques can be used for other applications as well.

1. INTRODUCTION [2]

Neural networks have recently been used for many different tasks, essentially of the classification type. In particular, multi-layer perceptrons, or MLPs, have been developed to learn to process data in perceptual problems, such as speech [Bottou, 1989], [Bourlard, 1987], [Waibel, 1987], image [Cottrell, to appear], [Le Cun, 1989], or signal [Gorman, 1988].

MLPs are said to be very efficient, but hard and slow to train. This drawback of the Gradient Back Propagation (GBP) learning algorithm can be overcome in essentially two ways. First, one can try various accelerating techniques instead of the basic gradient-following GBP: this includes conjugated gradients, second order rules [Becker, 1988], [Fahlman, 1988], [Watrous, 1987], … Or, one can try to reduce the number of weights, i.e. of parameters, in the network. This has also one interesting consequence, linked to the generalization ability of MLPs. The number of cells in the network must always be kept at a minimum: performances on the learning set might increase with an increasing number of cells, but will finally deteriorate on the test set, because of overfitting effects. With fully connected nets, there exists an "optimal"

---

[1] Publicated in “Neuro Computing, Algorithms, Architectures and Applications”. NATO ASI Series, Vol. F68 Neurocomputing. Edited by F. Fogelman Soulié and J. Hérault, © Springer-Verlag Berlin Heildelberg 1990

[2] The work presented at the meeting was realized in the fall of 1988. Since then, the work reported in [Holley, 1989] was published; their results seem rather similar to ours (on the same data base, but split differently between training and test sets). Their architectures were obtained experimentally, when we derived ours from detailed implementation of available knowledge. The comparison with this work thus perfectly illustrates our approach, i.e. to incorporate knowledge beforehand into the network architecture, instead of running many simulations with nets of varying architectures. This is why we decided to add references to Holley's work in the written version of our conference.

range for the number of hidden units, large enough to allow for good learning, while small enough for good generalization [Baum, 1989]. The problem, in practice, is to determine that range: trial and error might do, starting with an initial reasonable guess, such as e.g. w = mp, where w is the number of weights, m the number of examples and p the dimension of the outputs. Or, more efficiently, one can set the number of hidden units equal to the dimension of the Discriminant Analysis hyperplane [Gallinari, 1988]. Pruning has also been reported to produce good results by various authors.

However, the best way is certainly to set the connections according to knowledge of the domain. This has been achieved, for example, in the TDNN networks for speech [Bottou, 1989], [Waibel, 1987] or the shared weights networks for image processing [Le Cun, 1989]: the idea there is to impose a constraint on some of the cells of the network so as to make them behave as masks -or feature extractors- passed over the input. The initial structure of the masks can even be set up by looking into image or speech processing manuals and reproducing the classical feature extractors into the masks structures [Loncelle, 1989]. In these architectures, although the number of cells might still be large, the number of free parameters is largely reduced. The networks then learn rapidly and are easy to train, since they have few parameters to learn and these parameters are initialized at "reasonable" values.

In this paper, we will use a problem in Biotechnology as an example for testing these ideas. MLPs have been applied to various problems in Biotechnology: to predict secondary and/or tertiary structures of molecules from their sequence [Holley, 1989; Quian, 1988], to find homologies between sequences or regularities within a structure. Although toy problems are indeed quite illuminating, it is also important to test neural network techniques on hard real-sized problems. All the above problems are hard, of large economic potential, and, at present, do not have complete solutions; gaining a few percents on prediction accuracy is considered of real significance. The area is thus particularly worth investigating with neural net techniques.We will focus here on the problem of protein secondary structure prediction.

The paper is organized as follows: in section 2, we briefly introduce the problem of proteins secondary structure prediction. In section 3, we present various solutions by neural networks, and discuss their performances.

2. PROTEINS SECONDARY STRUCTURE PREDICTION

Proteins represent more than half the weight of cells. By adopting different spatial configurations, they realize many different functionalities: elaborate structures, receive or send messages, protect the organism, serve as enzymes in biochemical reactions… Finding the spatial configuration of a protein is still a very long and painful task, and few proteins have their spatial structure fully elucidated.

Proteins can be viewed as "words" on an alphabet of 20 aminoacids: this word is their **primary structure**. Their 3-D conformation or **tertiary structure** is characteristic of their capacity of entering into specific interactions with molecules, i.e. of their chemical activity. Although the tertiary structure is very often unknown, intermediate local structures have been identified, which are common to all proteins: the **secondary structure** of a protein is then the arrangement of these sub-structures. At present, biochemists have identified two main sub-structures: α-helices, ß-sheets, and more recently more complex sub-structures called domains.

Since systematically identifying the tertiary structure of proteins remains out of reach, many researchers have investigated the somewhat easier problem of predicting their secondary structure. The different methods developed sofar achieve around 60% accuracy; they include statistical methods: [Chou, 1978], [Garnier, 1978], [Gascuel, 1988], [Levin, 1986], knowledge-based methods: [Lim, 1974], or Neural Networks techniques: [Holley, 1989], [Qian, 1988]. It is not always easy to compare those techniques, since they do not use the same data bases. Table A1 shows the results reported by the authors.

Technique	accuracy (%)
Lim	59.3
Chou	49.9
Garnier	55.9
Levin	57.5
Gascuel	58.7
Quian	62.7
Holley	63.2

Table A1: Accuracy on protein secondary structure prediction.

All these techniques are local, in the sense that they predict the sub-structure associated to a residue on the basis of local information only. The fact that no technique can significantly do better than 60% accuracy is not clearly understood: it may be due to the limited size of the available data bases, or much more probably to the need of using long range information for predicting sub-structures. Neural net methods, as we will see in details, share this feature of using local-information-only, and thus have the same limitations. However, they seem to optimally exploit the statistical information included in the data bases, and allow to incorporate biochemical knowledge, thus producing the best existing results. We now describe the methodology used in Neural nets.

3. NETWORK SOLUTIONS

3.1. The data base

We will use Multi-layer networks or MLPs, trained by the Gradient Back Propagation algorithm [Le Cun, 1987], [Rumelhart, 1986]. Our work is intended to show how expert

knowledge can be usefully incorporated into an MLP so as to gain increased performances and speed. Our "expert" source has been [Gascuel, 1988].

The data base is taken from [Kabsch and Sanders, 1983] who provided a labelling of the Brookhaven Protein Data bank in three sub-structures: α-helix, ß-sheet and coil (those residues which are neither helix nor sheet). Among the proteins in the data base, some are homologous and their structures are then very close. The data base must be organized so as to take into account homologous proteins: the test set should not contain proteins homologous to those in the training set, since this would artificially **overestimate** the accuracy of the method. In [Holley, 1989], the training set contains the 48 first proteins in the Kabsch and Sanders data base, and the test set the last 14: hence no special care seems to have been taken to avoid homologies. In [Quian, 1988], to study this effect, the authors have investigated various test sets, with and without homologies with the training set, in a data base with 106 proteins from the Brookhaven Data base.

We have used the Kabsch and Sanders data base as described in [Gascuel, 1988]: proteins are considered homologous, if their homology is larger than 30% or if they contain segments of 60 amino acids at least with an homology larger than 30%. 19 homologous proteins have been found in the data base. We have thus used a training set of 41 proteins with no homology and a test set of 19 proteins with homologies (table A2).

1ABP, 1AZU, 1BP2, 1CAC, 1CPV, 1CRN, 1FDX, 1GCN, 1GPD, 1HIP, 1INS, 1LH1, 1LZM, 1MLT,
1OVO, 1PCY, 1PPT, 1REI, 1RHD, 1RNS, 1SBT, 1TIM, 2ADK, 2B5C, 2GRS, 2PAB, 2SNS, 2SOD,
2SSI, 3CNA, 3DFR, 3FXC, 3FXN, 3PGM, 3TLN, 4ADH, 4LDH, 4PTI, 4RXN, 5CPA, 7LYZ

155C, 351C, 3C2C, 3CYT
1APR , 2APP
1CTX, 1NXB
1ECD, 1LHB, 1MBN, 2MHB
1EST, 2ALP, 2GCH, 2PTN, 2SGA,
2ACT, 8PAP

Table A2: data base with Brookhaven identifiers.
The learning set has 41 proteins with no homology (shown in the first 3 lines) and the test set 19 proteins which have homologies among themselves, but not with the learning set: we show the 6 subsets of homologous proteins.

3.2. The network

The network used is of the "NetTalk" type [Sejnowski, 1987], [Quian, 1988]: a window of fixed size is passed over the protein "word" and the network is trained by GBP to produce at the output the sub-structure of the central residue (fig.A1). Each position in the window is coded in a group of 21 cells: one for each of the 20 amino-acids and one for the empty position. To determine the size of the window, we have used knowledge [Gascuel, 1988] about the importance of neighbors to determine the structure of a residue. Figure A2 shows these dependences for the three sub-structures: from this figure, we can infer that a window of size 17 would cover most of the important residues. In [Holley, 1989], windows of sizes 3 to

21 have been tried, size 17 happened to produce the best results. We saved these testings by directly using the knowledge available to us.

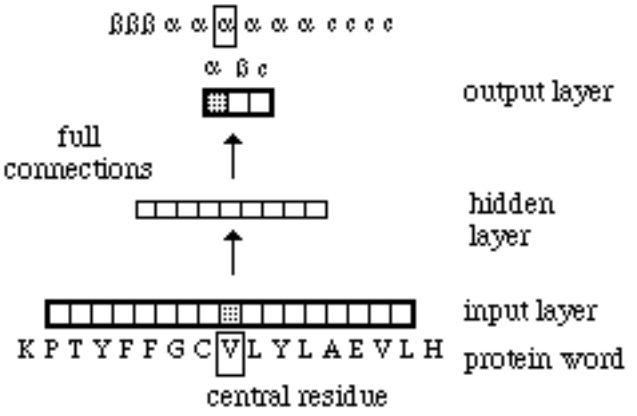

Figure A1: Multi-layer network for prediction of relative secondary structure of central residue

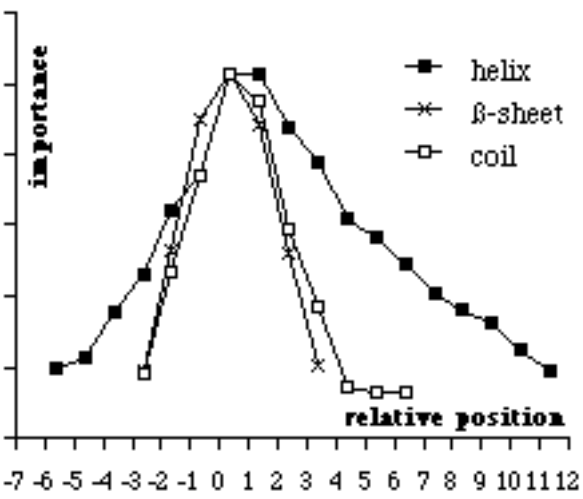

Figure A2: importance of neighbors, at position -7 to 12, for the determination of the structure of residue 0, for the three structures.

The output layer has three cells to code, by position, the three sub-structures. The residue is classified in the output category which has the highest activity. The prediction accuracy is measured by $n_\alpha + n_ß + n_{coil} / N$, where n_i is the number of residues of type i correctly classified and N is the total number of residues. Correlation coefficients can also be used and provide better indications about the quality of the method [Gascuel, 1988]. In addition, it seems reasonable to require that the prediction method be "well balanced", i.e. that the predicted proportions of structures are approximately equal to their observed proportions. Since coil structure is more probable than α-helix or ß-sheet (50% to 30 and 20%), overpredicting coil will overestimate the accuracy. This can be observed in [Holley, 1989] where the prediction accuracy on the test set is 66.9% for coils and only 59.2 and 53.3 for α-helix or ß-sheet.

The hidden layer has a varying number of cells, from 0 to 20: experiments showed that 2 to 3 hidden cells were sufficient. The layers have been fully connected, with random initial weights in the range $\pm 2.38 / \sqrt{\text{fan-in}}$ (which is provided by our simulator SN [Bottou, 1988]).

3.3. The trivial solution

The network structure just described involves very little knowledge, except the size of the window: it is the first trivial architecture that one can try. All our results are summarized in table A3: the first results in this trivial case (net 1) are slightly inferior to other classical techniques.

In [Gallinari, 1988], we have shown that MLPs with linear units performed Discriminant Analysis. It is thus reasonable to view MLPs as devices which start by performing DA,

especially when the weights are initialized so that the activity of the cells remain in the central, i.e. the linear, part of the sigmoïd. Afterwards, the MLP eventually improves, through non linearity, upon this solution, optimal in the linear case. We have thus used the same network as before, but, during training, we first started with 20 "linear" sweeps, by disabling the sigmoïd, and then followed with the usual sigmoïd sweeps. Table A3 shows that with this trick (net 2), we gained about 5% on test set. Notice that, since the dimension of the DA hyperplane is 2 here, we have a good theoretical reason to use 2 hidden units (which is the optimal number found experimentally in [Holley, 1989]).

Net	Window symmetric	Size of window	Coding of amino-ac.	weights triang.	linear learning	training 41 prot.	test 19 prot.
1	yes	17	21	no	no	55.3	51.5
2	yes	17	21	no	yes	61.2	55.6
3	yes	17	21	yes	yes	63.5	57.9
4	no	18	13	yes	yes	64.5	58.4

Table A3: accuracy (in %) for the different solutions

We have visualized the evolution of weights along learning. Figure A3 shows that, during learning, weights tend to develop triangular patterns similar to those shown in figure A2. We could have initialized those weights directly along triangular patterns and thus save the network the time required to learn this shape. This is what we now show.

3.4. Weights initialization

We have first tested a network identical to net 2, but where the weights are initialized with a triangular shape (fig. A4). Results are again improved (table A3: net 3). However, if we want to make full use of the information contained in figure A2, we can notice that the windows are not symmetric: for α-helix, the structure of a residue depends upon neighhbors -6 to 11, -3 to 3 for ß-sheet, and -3 to 6 for coils. To use this knowledge, we should thus use windows of different sizes and shapes for the three structures, which is not implementable with our model. We thus use a unique asymmetric window shape, corresponding to the largest case, α-helix, which makes about 26% of the residues in the data base. The window is now of size - 18, and is asymmetric: the network has to predict the structure of residue 0, taking into account residues at relative positions -6 to 11.

Figure A3: Evolution during learning of weights from one amino-acid, for each of the 17 (nets 1 to 3) or 18 (net 4) cells in the input window, to the hidden layer: the upper curve shows the weights for the amino-acid with maximum weight, the bottom curve the average on all amino-acids. Three periods (for net 1) or four (nets 2 to 4) are shown: period 1 shows initial weights, period 2 shows weights after 20 iterations (sigmoid for net 1, linear otherwise), period 3 after 20 more sigmoidal, period 4 after 100 more sigmoidal.

We have also used biochemical knowledge [Gascuel, 1988] which links together some amino-acids having similar physico-chemical properties and similar "preferences" for one of the

secondary structures. This allows to group those amino-acids and obtain a more compact coding along 13 cells, instead of 21 (fig.A5).With this coding, and asymmetric windows, we gain a little more on accuracy (see net 4 in table A3) and weights are almost perfect after 40 sweeps only (fig.A3). Notice that our best performances are still less than those reported in [Holley, 1989]: this is due to the fact that, in their test set, some proteins have homologies with proteins in the learning set, which is not the case in ours. Results in [Quian, 1988] are better than ours, but with a much larger data base: 89 proteins for the training set (and 15 for test set with no homology with training set).

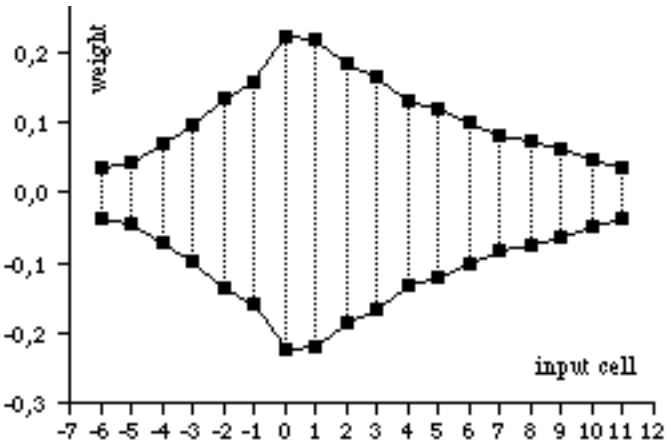

Figure A4: Triangular initialization of weights: for each cell of the window and each amino-acid, the weight is set at random between the two bounds shown.

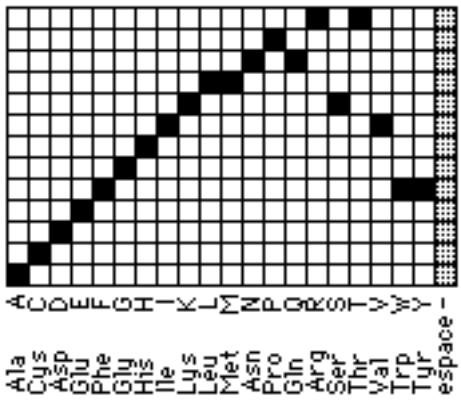

Figure A5: Grouping of "similar" amino-acids, and coding on 13 cells.

3.5. Pruning

Now, suppose we did not know about the DA dimension and we had used a network with too many hidden units, say for example with 20 hidden units. We train the network, and we want to see whether pruning would deteriorate the accuracy. With a network similar to case 4 before, but with 20 hidden units instead of 2, there are 4700 weights in layer 1 and 63 in layer 2. There are different methods for pruning.

We can first try random pruning. We can also use the information in figure A2, that we have partly ignored. Some cells should have a ß-sheet window [-3,3], others should have a coil window [-3,6]. Assume we keep 10 hidden units and out of the remaining 10, prune 6 of them the "coil" way (i.e. kill weights -6 to -4, 7 to 11) and 4 the "ß-sheet" way (kill weights -6 to -4 and 4 to 11). We will thus destroy (11*4+8*6)*13=1196 weights. To do so, we can choose 1196 weights at random or the smallest weights or the smallest cells (those with the smallest average weight) and prune 6 the coil way, 4 the ß-sheet way. The results are given in table A4 and show that, indeed, pruning is not destructive and is definitely more efficient when used with knowledge.

Pruning	learning	test
1. before pruning	64.51	58.31
2. random		
right after pruning	59.53	54.29
after 164 sweeps	65.27	57.79
3. smallest weights		
right after pruning	64.44	58.08
after 150 sweeps	65.23	58.18
4. smallest cells		
right after pruning	64.39	58.77
after 150 sweeps	65.35	58.70

Table A4: Effects of pruning on net 4, with 20 hidden units.

4. CONCLUSION

We have shown in this paper various ways by which a-priori domain knowledge could be used to set the architecture of MLPs: patterns of weights can be initialized so as to take into account the importance of particular factors in the input; the dimension of the Discriminant Analysis hyperplane provides a first guess for the number of hidden units; a-posteriori pruning can efficiently reduce the number of weights in the network.

All these techniques can be implemented in many different applications and guide the network designer: MLPs can then be trained faster and lead to better performances in generalization.

We wish to thank Olivier Gascuel, who provided us with his results and data base and very helpful indications at the beginning of our work.

5. REFERENCES

1. Baum, E., Haussler, D.: What Size Net Gives Valid Generalization ?, Neural Computation, Vol. 1. n° 1. 151-160. (1989).
2. Becker, S., Le Cun, Y.: Improving the convergence of bavk-propagation learning with second order methods. In "Proceedings, 1988 Connectionist models Summer School",Morgan Kaufman, 29-37, (1988).
3. Bottou, L.Y., Le Cun, Y.: SN: Un simulateur pour réseaux connexionnistes. "Neuro-Nîmes 88", EC2 Ed., 371-382, (1988).
4. Bottou, L., Fogelman Soulié, F., Liénard, J.S., Blanchet, P.: Speaker independent isolated digit recognition: multi layer perceptrons vs dynamic time warping. Neural Networks, to appear.
5. Bourlard, H., Wellekens, C.J.: Multilayer perceptrons and automatic speech recognition. In IEEE 1st ICNN, San Diego, M. Caudill, C. Butler eds, IEEE catalog n° 87TH0191-7, vol. IV, 407-416, (1987).
6. Chou, P.Y., Fasman, G.D.: Empirical prediction of protein conformation. ANN. Rev. Biochem., 47, 251-276, (1978).
7. Cottrell, G., Munro, P.W., Zipser, D.: Image compression by back propagation: a demonstration of extensional programming. In " Advances in Cognitive Science", N.E. Sharkey Ed., vol2, Norwood NJ Ablex. to appear.

8. Fahlman, S.E.: Faster learning variations on back-propagation: an empirical study. In "Proceedings, 1988 Connectionist models Summer School",Morgan Kaufman,38-51, (1988).

9. Fogelman Soulié, F., Gallinari, P., Le Cun, Y., Thiria, S.: Evaluation of network architectures on test learning tasks. In IEEE 1st ICNN, San Diego, M. Caudill, C. Butler eds, IEEE catalog n° 87TH0191-7, vol. II, 653-660, (1987).

10. Gallinari, P., Thiria, S., Fogelman Soulié, F.: Multilayer Perceptrons and Data Analysis. IEEE 2nd ICNN, San Diego, vol. I, 391-401, (1988).

11. Garnier, J., Osguthorpe, D.J., Robson, B.: Analysis of thje accuracy and implications of simple methods for predicting the secondary structure of globular proteins. J. Mol. Biol., 120, 97-120, (1978).

12. Gascuel, O., Golmard, J.L.: A simple method for predicting the secondary structure of globular proteins: implications and accuracy. Centre de Recherche en Informatique de Montpellier, Technical report, (1988).

13. Gorman, R.P., Sejnowski, T.J.: Analysis of hidden units in a Layered network trained to classify sonar targets. Neural Networks, vol.1, n°1, 75-89, (1988).

14. Holley, L.H., Karplus, M.: Protein structure prediction with a neural network. PNAS, USA, 86, 152-156, (1989).

15. Kabsch, W., Sander, C.: Dictionnary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers, 22, 2577-2637, (1983a).

16. Kabsch, W., Sander, C.: How good are predictions of protein secondary structure? FEBS Letters, 155, 179-182, (1983b).

17. Le Cun, Y.: Modèles connexionnistes de l'apprentissage, Thèse, Paris, (1987).

18. Le Cun, Y.: Generalization and network design strategies. In "Connectionism in perspective", R. Pfeifer, Z. Schreter, F. Fogelman Soulié, L. Steels eds, North Holland, 143-156, (1989).

19. Levin, J., Robson, B., Garnier, J.: An algorithm for secondary structure determination in proteins based on sequence similarity. FEBS Letters, 205, 303-308, (1986).

20. Lim, V.I.: Algorithms for prediction of alpha-helical and beta-structural regions in globular proteins. J. Mol. Biol., 88, 873-894, (1974).

21. Loncelle, J.: Détection de contours par rétro-propagation du gradient. "Neuro-Nîmes 88", EC2 Ed., 373-394, (1988).

22. Quian, N., Sejnowski, T.J.: Predicting the secondary structure of globular proteins using neural network models. J. Mol. Biol., 202, 865-884, (1988).

23. Rumelhart, D., Hinton, G., Williams, R.: Learning Internal Representations by Error Propagation, in Parallel Distributed Processing: Explorations in the MicroStructure of cognition, Rumelhart & McClelland eds, vol 1, chapter 8, MIT Press, (1986).

24. Sejnowski, T.J., Rosenberg, C.M.: Parallel networks that learn to pronounce english text. Complex Systems, vol.1, n°1, 145-168, (1987).

25. Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., Lang, K.: Phoneme recognition using Time-delay neural networks. Preprint ATR Interpreting Telephony Research Laboratories, (oct. 1987).

26. Watrous, R.L.: Learning algorithms for connectionist networks: applied gradient methods of non linear optimization. In IEEE 1st ICNN, San Diego, M. Caudill, C. Butler eds, IEEE catalog n° 87TH0191-7, vol. II, 619-627, (1987).