Neural network optimization for *E.coli* promoter prediction

Borries Demeler* and Guangwen Zhou

Department of Biochemistry and Biophysics, Oregon State University, Corvallis, OR 97331-6503, USA

Received November 30, 1990; Revised and Accepted February 5, 1991

ABSTRACT

Methods for optimizing the prediction of *Escherichia coli* RNA polymerase promoter sequences by neural networks are presented. A neural network was trained on a set of 80 known promoter sequences combined with different numbers of random sequences. The conserved –10 region and –35 region of the promoter sequences and a combination of these regions were used in three independent training sets. The prediction accuracy of the resulting weight matrix was tested against a separate set of 30 known promoter sequences and 1500 random sequences. The effects of the network's topology, the extent of training, the number of random sequences in the training set and the effects of different data representations were examined and optimized. Accuracies of 100% on the promoter test set and 98.4% on the random test set were achieved with the optimal parameters.

INTRODUCTION

An important objective in molecular biology is to analyze DNA sequences for structural and functional motifs. Efforts have been made to develop statistical algorithms for sequence analysis and motif prediction by searching for homologous regions or by comparing sequence information to a consensus sequence (1). These algorithms may fail or yield insufficiently accurate results when consensus sequences are difficult to define (1,2). In this paper we discuss the analysis of nucleic acid sequences by optimizing the parameters of a neural network implementation. The ability of optimized multilayered neural networks to predict structural and functional elements in DNA is demonstrated on bacterial RNA polymerase promoter sequences from *Escherichia coli* (*E. coli*).

Neural networks have previously been applied to biological problems: The utility of neural networks for protein structure prediction from amino acid sequence data is documented in references (3–7). Brunak *et al.* (8) report the ability of a neural network to detect errors in the assignment of mRNA splice sites in nucleic acid sequence databases. While a perceptron model for the classification of *E. coli* translational initiation sites has been investigated by Stormo *et al.* (9), and a neural network model for the prediction of promoter sequences has been proposed by Lukashin *et al.* (10), a comprehensive analysis of neural network implementations for nucleic acid sequence analysis has

not been presented so far. In this work we present an empirical approach to the optimal encoding of neural networks for promoter recognition with respect to parameters outlined in the abstract.

Neural networks were originally developed to simulate the learning process in the brain by modeling the brain's anatomy of neural cells connected by axons over different synaptic strengths. In a neural network, neural cells are replaced by interconnected computational units, called neurons. Axons are represented by symbolic connections between neurons, and synaptic strengths are represented by weights and thresholds (neuron biases), applied to the connections and neurons, respectively. During the training stage, a training set of inputs and corresponding targets is presented to the neural network. The initially random weights and thresholds are continuously modified according to a steepest gradient descent learning algorithm. With each iteration, the learning algorithm attempts to minimize an objective function, generally the error function of the training set, which describes the difference between the observed output and the desired output. In its learned stage, the weights constitute the networks ability to recognize a pattern intrinsic to the training set. We believe that the ability of a neural network to accomplish pattern recognition tasks in an automated fashion is well suited for sequence analysis.

The bacterial promoter sequences chosen for this research contain two conserved regions, the –10 and the –35 region, centered about 10 bp (basepairs) and 35 bp upstream from the transcriptional start site. The conserved –10 and –35 regions are connected by a spacer region, which can be categorized into 3 classes, depending on the number of bases contained in the spacer region (2). A previous attempt to predict promoter sequences relies on six empirically developed tests to filter out false positives predicted by a consensus sequence match algorithm, which also determines the spacer class the predicted sequence belongs to. However, overall prediction accuracy reaches only 77% (1). While Lukashin *et al.* (10) report a maximal overall prediction accuracy of 93%, our optimized neural network implementation significantly improves on the prediction accuracy to 98.4% and allows sequence analysis of regions with poorly definable consensus sequence. Also, our training set does not require the introduction of gaps for perfect alignment, because a continuous sequence is used for the training. This allows analysis of sequences where alignment is difficult or impossible to define, a question not addressed by Lukashin *et al.* (10).

* To whom correspondence should be addressed

METHODS

Data

The promoter sequences were taken from the compilation of *E. coli* RNA polymerase promoter sequences by Hawley and McClure (11). Of those, 80 bacterial and phage promoters were used for the training database. The remaining 30 plasmid and transposon promoter sequences and promoters generated by mutation were used in a test set to evaluate our method. The promoter sequences were arranged into three independent training and corresponding test sets:

(A) 20 bases centered around the -10 region, which includes the TATAAT consensus sequence with the first T at the 12th position;

(B) 20 bases centered around the -35 region, which includes the TTGACA consensus sequence with the first T at the 10th position.

(C) 44 bases aligned as in (A) and containing the -35 region without a gap between the conserved -10 and -35 region.

A pseudo-random number generating program was used to generate random sequences with equal composition A,G,T,C, which were combined with the promoter sequences in the training set in a ratio of 1:1–20:1. The sequences were arranged by having 1–20 random sequences follow each promoter sequence in the training set, depending on the ratio of random vs. promoter sequences tested. Different random sequences were tried to determine if exchanging one random sequence for another could alter the result. Each test set for the random predictions contained 1500 random sequences in a format corresponding to (A), (B) or (C).

Calculations and network formulation

The calculations were performed on a SUN 3/260 and on a 386 AT equipped with a math-coprocessor. The program was written in FORTRAN and was of our own design. It featured a feed-forward type network (Fig.1) using the delta rule in back-propagation and a momentum term as described in Wasserman (12). The input data was coded as a binary, orthonormal set of

Figure 1. Shown is a neural network architecture with 4 neurons in the hidden layer, $k=4$. The number of input units is either 4 times the number of bases (CODE-4) or 2 times the number of bases in the input sequence (CODE-2). The values of the input units can be 0 or 1. There is one output unit and it is in the range between 0 and 1. Weights₁ are applied to the connections between the input and the hidden layer, weights₂ are applied to the connections between the hidden and output layer. Only the neurons in the hidden and output layer serve a computing function, the inputs only serve to distribute.

dimension 4, called CODE-4 (0001=C, 0010=G, 0100=A, 1000=T), or a linearly dependent set of dimension 2, called CODE-2 (00=A, 01=T, 10=G, 11=C). If CODE-4 was used, the number of input units was 4 times the number of bases analyzed, similarly, in CODE-2 there were twice as many input units as bases in the test sequence. The target to each input sequence was coded as '1' for promoters, and '0' for random sequences. The forward pass was calculated by summing the products of all inputs to a neuron in the hidden layer with their respective weights (weight_{1,ik}), adjusting with a neuron bias and operating on the sum with an activation function F . The outputs of the activation function from the hidden layer neurons were multiplied with their respective weights (weight_{2,k}) and served as inputs into the single neuron of the output layer. For the k th neuron in the hidden layer:

$$X_{1,k} = \sum_{i=1}^n (\text{weight}_{1,ik} \cdot \text{input}_i) \quad (1)$$

$$F(X_{1,k}) = \frac{1}{1 + e^{(-X_{1,k} + t_{1,k})}} \quad (2)$$

where $t_{1,k}$ is the threshold adjustment for the k th neuron in the hidden layer. The output was determined by summing $F(X_k)$ over all k and calculating the activation function F :

$$X_2 = \sum_{i=1}^k \text{weight}_{2,i} \cdot F(X_{1,i}) \quad (3)$$

$$\text{output} = F(X_2) = \frac{1}{1 + e^{(-X_2 + t_2)}} \quad (4)$$

The backpropagation algorithm used is outlined in reference (12), with the modification that the training rate was continuously adjusted during training as a function of the derivative of the error function. This measure helps to avoid network paralysis in a local minimum of the parameter hyperspace. If the training process is trapped in a local minimum, the total error is larger than 0 and the derivative of the error function approaches 0 asymptotically. In such a case increasing the training rate increases the stepsize allowed in the gradient descent which aids in overcoming the hills of the errorsurface which surround the local minimum. The error function E was evaluated according to:

$$E_i = \sum_{j=1}^{240} (\text{target}_j - \text{output}_j)^2 \quad (5)$$

where i is the iteration and j is the number of the sequence in the training set. The training rate was adjusted according to

$$\text{training rate} = \frac{k}{(E_{i-1} - E_i)} \quad (6)$$

where k is a constant, generally set to 10^{-6} and i is the iteration. The thresholds were adjusted by backpropagation like the weights, except that their inputs were always taken as 1. The weights were initially assigned random values in the range of 0.01 to -0.01 .

The network was trained until the error level reached 10^{-4} . To determine the effect of the extent of training, weight matrices with error levels between $10-0.001$ were used to test the

prediction accuracy (training set C). Topologies ranging from 1 to 10 neurons in the hidden layer were tested in their prediction accuracy.

Evaluation of results

To evaluate our network, the trained network was used to predict both the random and promoter sequences in the test set. The output generated was in the range between 0 and 1 and in the format of real numbers. This allowed differentiation between strong and weak predictions. An output larger than 0.5 was interpreted as 'promoter sequence', while outputs less than 0.5 were interpreted as 'random sequences'. Prediction accuracy was measured as the fraction of sequences in the test set predicted correct. Predictions were performed independently on promoter sets and random sets. The overall prediction accuracy of the network is given by:

$$(F_{\text{random}} * F_{\text{promoter}}) * 100\% \quad (7)$$

where F_{random} denotes the fraction of correct predictions from the random test set and F_{promoter} the fraction of correct predictions from the promoter set.

RESULTS

Data representation

Initially, training set (A) was used to establish the better of the two encoding schemes. Each base in the training set was translated into binary code according to CODE-2 and CODE-4 and trained to an error level of 10^{-4} . The trained weight matrices were used to predict the test set of (A). At the same time, the number of neurons in the hidden layer were varied to determine the effect of network complexity. The results are summarized in Fig.2. A substantial difference in performance was noticed between CODE-2 and CODE-4.

Network topology

Network topologies of hidden layer size 1–10 neurons were tested. No clear indication for the optimal hidden layer size

Figure 2. Training results comparing CODE-2 with CODE-4. The %-accuracy for the combined predictions for CODE-2 is shown in black bars, prediction results for CODE-4 is shown in hatched bars. Network topologies ranging between 4 and 10 hidden layer neurons are compared, when the error level was trained to 10^{-4} .

emerged. Modification of other parameters at the same time did not change this result. Therefore, we restricted our analysis to topologies involving 1 to 10 hidden layer neurons (the case of 0 neurons (perceptron) in the hidden layer is basically equivalent to the case of 1 neuron in the hidden layer, where the output neuron merely serves to add another neuron bias, so that the neuron biases for both neurons are shared).

Training set composition

To determine the optimal ratio of random sequences : promoter sequences (r:p ratio) we varied the composition of the training set with r:p ratios between 1:1 to 20:1. For training sets containing 20 bases the r:p ratio was found to be optimal at 2:1. Increasing the r:p ratio from 1:1 to 2:1 added an average of 5.8%

Figure 3. (A) Accuracy of combined prediction results for r:p ratios ranging between 1:1 and 3:1 for 20 bases in the input sequence. The data was obtained by averaging results from network topologies ranging between 4 and 10 neurons in the hidden layer. (B) Accuracy of combined prediction results for r:p ratios ranging between 2:1 and 20:1 for 44 bases in the input sequence. (B1): promoter predictions. (B2): random predictions (B3): combined predictions.

Figure 4. The effect of varying the extent of training for training set (C) with a r:p ratio between 2:1 – 10:1. Results shown are averaged from network topologies between 1 and 10. From left (blank bar) to right (filled bar) error level = 1, 0.1, 0.01, 0.001, 0.0001.

Table 1. Effect of Extent of Training on Prediction Accuracy.

ERROR LEVEL:	r:p	1.0	0.1	0.01	0.001	0.0001
PROMOTER PREDICTIONS:	2:1	90.29	93.57	93.82	93.73	94.37
	3:1	93.59	93.59	93.10	93.10	93.59
	4:1	93.33	93.33	93.33	93.33	93.33
	5:1	91.93	91.79	93.10	93.97	95.64
	10:1	82.26	88.17	90.86	94.06	94.62
	20:1	76.34	79.03	82.26	83.87	84.41
RANDOM PREDICTIONS:	2:1	96.74	96.24	95.79	95.52	95.35
	3:1	95.52	95.89	95.45	95.33	95.29
	4:1	98.07	97.86	97.61	97.46	97.35
	5:1	98.30	98.22	97.87	97.92	97.79
	10:1	99.26	99.01	98.89	98.78	98.70
	20:1	99.52	99.56	99.55	99.60	99.34
COMBINED PREDICTIONS:	2:1	87.35	90.05	89.87	89.53	89.98
	3:1	89.40	89.74	88.86	88.75	89.18
	4:1	91.53	91.33	91.10	90.96	90.86
	5:1	90.37	90.16	91.12	92.02	93.53
	10:1	81.65	87.30	89.85	92.91	93.39
	20:1	75.97	78.95	81.89	83.53	83.85

Shown are combined prediction results for training set (C). Combined prediction results were obtained by averaging results from all networks with 1–10 neurons in the hidden layer. Results for different r:p ratios are shown for different stages in the error level of the training process. The combined prediction accuracy is highest for a r:p ratio of 5:1 and an error level of 0.0001.

accuracy over the different architectures examined. However, changing to a ratio of 3:1, the overall prediction accuracy dropped slightly by an average of 1% (Fig.3,A). For the training sets containing 44 bases, the optimal r:p ratio was found to be 5:1 (Fig.3,B), although results from training sets containing a ratio of 10:1 were very similar (0.15% difference). A larger increase in prediction accuracy was noticed in changing from 2:1 to 4:1 for (C) (0.9% difference). Changing to a r:p ratio of 20:1 decreased the promoter sequence prediction accuracy sharply, however, the random sequence prediction accuracy increased slightly (Fig.3,B). To determine if prediction results are sensitive to an exchange of one random sequence for another, different sets of random sequences were incorporated in the training set. No significant changes in the results for above parameters were observed (data not shown).

Extent of training

Training was performed until the total network error on the training set reached 10^{-4} . Training was interrupted at intermediate stages (starting at error levels around 10) and the weight matrices analyzed (Fig.4). Generally, once training became asymptotic, prediction accuracy did not improve substantially. Maximum prediction accuracy was usually attained between error levels of 1.0 and 0.0001. The accuracy for promoter, random and combined predictions from 6 different training sets for (C) at various error levels is shown in Table 1.

Training set results

After establishing these parameters, prediction accuracy of all three training sets with their optimal parameters were compared (Fig.5). The optimal parameters (the parameters which give the highest prediction accuracy) of the three different training sets are listed in Table 2. Prediction accuracy of the training sets was found to be best for (C) with 98.4%, then (A) with 90% and (B) only 68.4%.

Figure 5. Combined prediction results for the three different training sets. Results are shown for different network topologies, ranging from 4 to 10 neurons in hidden layer. (A): training = 10^{-4} , r:p = 2:1 (white bar). (B): training = 10^{-4} , r:p = 2:1 (black bar). (C): training = 10^{-4} , r:p = 5:1 (hatched bar).

Consensus sequence prediction

Using the weight matrix obtained from the network with one neuron in the hidden layer (this case is equivalent to the perceptron formulation), it is possible to predict a consensus sequence. The value of the weights is plotted in groups of four, each element of a group representing one of four nucleotides in that sequence position. The value of the weight can be negative or positive, depending on whether the corresponding neuron is inhibitory or excitatory. The consensus sequence is determined by taking the most positive element of a group and relating it to the corresponding nucleotide. Strongly negative weights indicate nucleotides not favored in a particular position. Using such a diagram it is possible not only to extract a consensus

Table 2. Optimal Parameters for Training Sets A, B and C:

training set:	r:p ratio:	encoding scheme:	extent of training:	maximal accuracy:	number of neurons
(A)	2:1	CODE-2	0.0001	88.85	9
(A)	2:1	CODE-4	0.0001	90.12	5
(B)	2:1	CODE-4	0.0001	68.39	8
(C)	5:1	CODE-4	0.0001	98.40	5

Shown are prediction results for the optimal configurations from each training set. The maximal accuracy is the combined accuracy of promoter and random predictions.

sequence for a promoter, but also to determine the significance of each nucleotide in a particular sequence position. The result for a network with a single neuron for training set (C) with a r:p ratio 5:1 is shown in Fig.6.

Genomic sequence predictions

Promoter positions in phage fd genomic sequence (6408 basepairs) were predicted with our neural network. The prediction results are shown in Table 3 for a weight matrix with one hidden layer neuron from training set C (with phage fd promoters excluded). No significant differences were found when other network architectures were employed. Promoters with prediction values between 0.5 and 0.9 were classified as weak promoters, those above 0.9 as strong promoters.

Of the eleven promoters determined by Schaller *et al.* (13), promoters fd V and fd I' were not predicted by the neural network, nine others were identified as strong promoters (r:p ratios between 2:1 and 10:1). Overprediction decreased for matrices from 2:1 to 10:1. Although overprediction was further decreased by changing from an r:p ratio of 10:1 to 20:1, only seven promoters were correctly determined with an r:p ratio of 20:1. Statistical methods only predicted five promoters correctly (14). Obviously, the major problem in the prediction of promoters is overprediction. Increased r:p ratios will, in general, decrease overprediction, but also decrease promoter prediction accuracy, a result which agrees with our artificial sequence predictions. The results from artificial sequences suggest that overprediction between 2%–4% is to be expected. For phage fd, 120–240 mispredicted promoters would be expected. Preliminary results using a larger database (including additional promoters from Harley and Reynolds (15)) indicate that overprediction can be reduced significantly (data not shown).

DISCUSSION

From the results summarized in Fig.2, it seems that CODE-4 is the better choice for DNA data representation. This is a unitary coding matrix with identical Hamming distance among each vector. Unitary coding has been used previously for encoding amino acids in secondary protein structure prediction by neural networks (5,7) and perceptron analysis of nucleic acids (8,9). CODE-2 representation features linearly dependent vectors, where the Hamming distance among vectors is different (Hamming distance is a measure for the difference between vectors by counting the number of different bit entries, i.e., a vector 1,1 has a Hamming distance of 1 compared to the vector 0,1 while the Hamming distance is 2 compared to the vector 0,0. In CODE-4 encoding all vectors have an equal Hamming distance of 2 between each other. Only comparisons between vectors of

Figure 6. Weight diagram for the single neuron case. The consensus sequence is indicated on top, together with the respective sequence position. Weights are shown as excitatory (above middle line) or inhibitory (below middle line). Each sequence position has a certain value for each possible nucleotide.

the same dimension is meaningful). This presumably could bias the learning process towards certain bases. Although we did not test the effects of CODE-2 vs. CODE-4 on training set (B) and (C), we expect the results to be similar. Combined prediction

Table 3. Prediction results for genomic sequence data of phage fd:

r:p	Error	1	0.1	0.01	0.001	0.0001
2:1	Weak	481	354	259	213	167
	Strong	153	441	567	637	687
4:1	Weak	387	257	184	142	105
	Strong	117	318	405	467	503
5:1	Weak	336	264	183	131	96
	Strong	135	329	434	487	523
10:1	Weak	224	179	122	96	77
	Strong	99	202	282	329	360
20:1	Weak	47	46	53	38	27
	Strong	152	179	179	141	150

Prediction results for phage fd. Promoters are classified as 'weak' (prediction between 0.5 and 0.9) and 'strong' (>0.9). More extensive training increases the number of strong promoter predictions, larger r:p ratios favor a decrease in overprediction.

accuracy increased by an average of 9% if CODE-4 was used. While Lukashin *et al.* (10) used CODE-2 encoding, Brunak *et al.* (8) and Stormo *et al.* (9) employed CODE-4 encoding for their studies. We checked the effect of different hidden layer size simultaneously with each parameter modified. Hidden layer size did not seem to influence the results obtained in a predictable fashion and differences were only minor. While the representational power of the network increases with added neurons, the generalization ability of the network decreases. If the amount of data increases (by using larger r:p ratios) the need for larger networks arises. This was exemplified by the failure of smaller networks to train below a certain error level. If the representational power was increased by adding neurons and weights, this problem was usually overcome. Training behavior was also strongly dependent on the initial, randomized weight matrix used. Failure to train could be overcome by modifying the initial weight matrix, thus providing the network with a starting point closer to the learned state.

The r:p ratio in training set (A) and (B) seems to be optimal at 2:1, although we did not test ratios higher than 3:1. Since the number of possible sequences is much lower for training sets containing 20 bases (1.1×10^{12}) than for training sets containing 44 bases (3.1×10^{26}), it seems not surprising that a higher r:p ratio results in better predictions for the 44 base training set.

As Holley and Karplus report in (5), we also observe that training beyond a certain error level (10^{-4}) does not add to the prediction accuracy. This might be due to loss of generalization through learning of the idiosyncracies of the sequences in the training set. Training to a level where the network error is below 1.0 means that all sequences in the training set can be correctly classified as 'random' and 'promoter'. The optimal extent of training for r:p ratios of 4:1 and 3:1 were obtained at an error of 1.0 and 0.1, respectively. When the r:p ratio is increased to 5:1, the best prediction is obtained at an error of 0.0001. This might be due to the additional information provided to the network with the additional random sequences in the training set. While promoter predictions generally improve with more extensive training, random prediction accuracy could be improved only by using a higher r:p ratio. There is a trade-off, which needs to be optimized, as promoter prediction accuracy naturally decreases with added random sequences (Fig.3,B).

In comparison of (A), (B) and (C), significant differences between predictions of the -10 region and the -35 region were observed. The differences between (A) and (B) are presumably due to a greater correlation between the promoter sequences in the -10 set. This suggests that the -35 region is less conserved than the -10 region of *E. coli* promoter sequences. An average difference of 23% was observed, with (A) giving the better result. Although the prediction accuracy of training set (B) is substantially lower (68.4%) than that of (A) (90%), it is still above 50%, which is the accuracy expected if no classification can be made at all. This suggests that the informational content of (B) could add to that of (A), if (A) and (B) were combined to (C). Here it is important to note that an alignment question arises: The sequences of (A) are separated by three different spacing classes from the sequences of (B) (2). We aligned the combined sequences both as in (A) and (B) in independent training sets. The results indicated that alignment of the TATAAT box regions in (A) is preferential to alignments of the TTGACA consensus sequence in (B) (data not shown). Combination of (A) and (B) as in (C) provided an additional prediction accuracy of 3.5%.

We believe that neural networks can provide a valuable alternative method in the analysis of DNA or RNA sequences, as demonstrated by the prediction of promoter sequences. In addition to attaining a prediction accuracy of 98.4%, our findings show that:

1. CODE-4 encoding for nucleic acid data should be used in preference to CODE-2 encoding.
2. The number of hidden layer units does not have a significant effect on prediction accuracy.
3. A low network error level improves promoter prediction accuracy, while a larger r:p ratio improves the network's ability to filter out false positives. For promoter predictions, we found the optimal r:p ratio to be between 5:1 and 10:1.
4. A combination of the conserved regions around -10 and -35 into a sequence of 44 bases length aligned at the -10 region yields better prediction results than a sequence of 20 bases length around either conserved region alone.
5. The utility of our method is characterized best by the improvement in prediction accuracy over previous methods. While the promoter prediction accuracy is high, the overall prediction error is caused mainly by overprediction, i.e. false

positives. Therefore, we believe that our method is useful for the localization of possible promoter sequences, which could be further verified by biochemical means.

We suggest that the overall prediction accuracy could be increased by including more promoter sequences in the training database. We plan to investigate effects of various AT:GC ratios in random training sequences on the prediction accuracy. Since promoter sequences contain a higher ratio of AT:GC than 1:1 (as in our random sequences), inclusion of random sequences with higher AT:GC ratios in the training database may lead to a more sensitive classification between promoters and non-promoters.

ACKNOWLEDGMENTS

We thank Dr. Curtis Johnson (Dept. of Biochemistry and Biophysics, OSU) for providing CPU time on a 386 AT and the staff for BIONETTE.CGRB.ORST.EDU for support on the SUN 3/260. We are grateful to Dr. Richard Schori, Robert Burton, Isa Yubran (Dept. of Mathematics, OSU) and Dr. Christopher Mathews (Dept. of Biochemistry and Biophysics, OSU) for their helpful advice. Both B.D. and G.Z. are graduate students, and are supported by a grant of PHS #30-262-0600 of Dr. Ken van Holde and by a grant from the Department of the Navy, #30-262-3025 of Dr. Pui Shing Ho, respectively.

REFERENCES

1. O'Neill, M.C. and Chiafari, F. (1989) *J. Biol. Chem.* **264**, 5531–5534.
2. O'Neill, M.C. (1989) *J. Biol. Chem.* **264**, 5522–5530.
3. Bohr, H., Bohr, J., Brunak, S., Cotterill, R.M.J., Lautrup, B. and Petersen, S.B. (1990) *FEBS Letters*. **261**, 43–46.
4. Bohr, H., Bohr, J., Brunak, S., Cotterill, R.M.J., Lautrup, B., Noerskov, L., Olsen, Q.H. and Petersen, S.B. (1988) *FEBS Letters*. **241**, 223–228.
5. Holley, L.H. and Karplus, M. (1989) *Proc. Natl. Acad. Sci. USA*. **86**, 152–156.
6. Kinoshita, J. (1990) *Sci. Amer.* **262**, 24–26.
7. Qian, N. and Sejnowski, T.J. (1988) *J. Mol. Biol.* **202**, 865–884.
8. Brunak, S., Engelbrecht, J. and Knudsen, S. (1990) *Nuc. Acids res.* **18**, 4797–4801.
9. Stormo, G.D., Schneider, T.D., Gold, L. and Ehrenfeucht, A. (1982) *Nuc. Acids Res.* **10**, 2997–3011.
10. Lukashin, A.V., Anshelevich, V.V., Amirkyan, B.R., Gragerov, A.I. and Frank-Kamenetski, M.D. (1989) *J. Biomol. Struct. & Dyn.* **6**, 1123–1133.
11. Hawley, D.K. and McClure, W.R. (1983) *Nuc. Acids Res.* **11**, 2237–2255.
12. Wasserman, P. D. (1989) *Neural Computing, Theory and Practice*. (Van Nostrand Reinhold, New York, NY). p. 6–59.
13. Schaller, H. and Beck, E. and Takanami, M. (1978) In Denhardt, D.T., Dressler, D. and Ray, D.S. (ed.) *Single Stranded DNA Phages*. Cold Spring Harbor Laboratory, Cold Spring Harbor, N.Y. pp 139–163.
14. Mulligan, M.E. and McClure, W.R. (1986) *Nucl. Acids Res.* **14**, 109–126.
15. Harley, B.C. and Reynolds, R.P. (1987) *Nucl. Acids Res.* **15**,