International Journal on Artificial Intelligence Tools
Vol. 0, No. 0 (1993) 000—000

BIOLOGICAL DATA MINING USING BAYESIAN NEURAL NETWORKS: A CASE STUDY

QICHENG MA and JASON T. L. WANG
Department of Computer and Information Science, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA

Received (4 January 1999)
Revised (15 March 1999)

Biological data mining is the activity of finding significant information in biomolecular data. The significant information may refer to motifs, clusters, genes, and protein signatures. This paper presents an example of biological data mining: the recognition of promoters in DNA. We propose a two-level ensemble of classifiers to recognize E. Coli promoter sequences. The first-level classifiers include three Bayesian neural networks that learn from three different feature sets. The outputs of the first-level classifiers are combined in the second-level to give the final result. Empirical study shows that a precision rate of 92.2% is achieved, indicating an excellent performance of the proposed approach.

Keywords: Bayesian neural networks, computational biology and medicine, data mining and classification, promoter recognition.

1. Introduction

As a result of the ongoing Human Genome Project,[9] DNA and protein data are accumulated at a speed growing at an exponential rate. Mining these biological data to extract significant information becomes extremely important in accelerating genome processing.[33] Classification, or supervised learning, is one of the major data mining processes. Classification is to partition a set of data into two or more categories. When there are only two categories, it is called *binary classification*. In this paper we focus on binary classification of DNA sequences.

In binary classification, we are given some training data including both positive and negative examples. The positive data belongs to a target class, whereas the negative data belongs to the non-target class. The goal is to assign unlabeled test data to either the target class or the non-target class. In our case, the test data are some unlabeled DNA sequences, the positive data are promoters and the negative data are non-promoters. Since our goal is to identify promoters in the unlabeled DNA sequences, we use the terms "classification" and "recognition" interchangeably in the paper.

Currently, techniques used for biological sequence classification fall into two categories:

(1) **Similarity search** – This approach is to classify unlabeled test sequences by searching for either global similarities or local similarities in the sequences. Global similarity search involves either pairwise sequence comparison,[1,24] or multiple sequence alignment.[2] Local similarity search is to find patterns in the sequences; see Brazma *et al.* (1998)[3] for an excellent survey.

(2) **Machine learning** – This approach was surveyed in Haussler (1997).[11] Various machine learning techniques have been applied to biological sequence classification. For example, hidden Markov models have been used in gene identification[16] as well as protein family modeling.[15] Neural networks have been applied to the analysis of biosequences; see Wu (1997)[34] for a survey. In Salzberg (1997),[27] a decision tree was used to find genes in DNA.

Here, we propose a two-level approach to recognizing E. Coli promoters in DNA sequences. The first-level classifiers include Bayesian neural networks[18,19,20,22] trained on different feature sets. The outputs of the first-level classifiers are combined in the second level to give the final classification result. Dietterich[8] recently indicated that using an ensemble of classifiers can achieve a better recognition rate than using a single classifier when (i) the recognition rate of each individual classifier of the ensemble is greater than 0.5; and (ii) errors made by each individual classifier are uncorrelated. Our experimental results show that our combined classifiers indeed outperform the individual classifiers made up solely by Bayesian neural networks.

Using an ensemble of classifiers to process biomolecular data has been studied in Brunak *et al.* (1991),[4] Wang *et al.* (1996),[32] Zhang *et al.* (1992).[36] In Brunak *et al.* (1991),[4] the complementary relation between exon and splice site classification was exploited to build a joint recognition system by allowing the exon signal to control the threshold used to assign splice sites. Specifically, a higher threshold was required to avoid false positives for regions where there are only small changes in the exon activity. A lower threshold was used to detect the donor site for regions where the exon activity decreases significantly. Similarly, a lower threshold was used to detect the acceptor site for regions where the exon activity increases significantly. In Zhang *et al.* (1992),[36] a hybrid system, which included a neural network, a statistical classifier and a memory-based reasoning classifier, was developed to predict the secondary structures of proteins. Initially, the three classifiers were trained separately. Then a neural network used as a combiner was trained to combine the outputs of the three classifiers by learning the weights for each classifier from the training data. The result of the classification was given by the combiner. In Wang *et al.* (1996),[32] the complementarity of five classifiers for protein sequence recognition was studied, and an ensemble of the classifiers was proposed, which outperformed each individual classifier.

In contrast to the previous work, we apply Bayesian neural networks to recognize promoters in DNA. Each Bayesian neural network combines the constituent neural networks by marginalisation. The uncertainty about each constituent neural

network is taken into account by weighting the neural network by its posterior probability. Furthermore, the Bayesian neural network controls the model complexity to avoid the overfitting problem.[18]

The rest of the paper is organized as follows. Section 2 describes the characteristics of E. Coli promoters and our feature extraction methods. Section 3 and Section 4 present our two-level classification approach. Section 5 presents experimental results. Section 6 concludes the paper.

2. Promoter Recognition

2.1. *Encoding methods*

One important issue in applying neural networks to biosequence analysis is regarding how to encode the biosequences, i.e. how to represent the biosequences as the input of the neural networks. Good input representations make it easier for the neural networks to recognize the underlying regularities. Thus, good input representations are crucial to the success of the neural network learning.[13]

One of the encoding methods is *orthogonal encoding*. In orthogonal encoding, nucleotides or amino acids in a biosequence are viewed as unordered categorical values, and are represented by C dimensional orthogonal binary vectors, where C is the cardinality of the 4-letter DNA alphabet $\mathcal{D} = \{\texttt{A, T, G, C}\}$, or the cardinality of the 20-letter amino acid alphabet $\mathcal{A} = \{\texttt{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}\}$. That is, we use C binary (0/1) variables, among which only one binary variable is set to 1 to represent one of the C possible categorical values and the rest are all set to 0. For instance, we represent the nucleotide `A` by "1000", and amino acid `Y` by "00000000000000000001". The orthogonal encoding was frequently used in the early 1990s.[7,14] Figure 1 shows an example of the orthogonal encoding of a DNA sequence.

The orthogonal encoding requires that the biosequences be equal in length, or one must sample the biosequences of variable lengths by a window of fixed size. Another disadvantage is that it wastes a lot of input units in the input layer of a neural network. For instance, for a protein sequence of 100 amino acids, 2000 input units are required to represent the protein sequence. This requires many neural network weight parameters as well as many training data, making it difficult to train the neural network.

An alternative encoding method is to use high-level features extracted from biosequences. The high level features should be relevant and biologically meaningful. By "relevant", we mean that there should be high mutual information between the features and the output of the neural network, where the mutual information measures the average reduction in uncertainty about the output of the neural network given the values of the features. By "biologically meaningful", we mean that the features should reflect the biological characteristics of the sequences.

2.2. *Characteristics of E. Coli promoters*

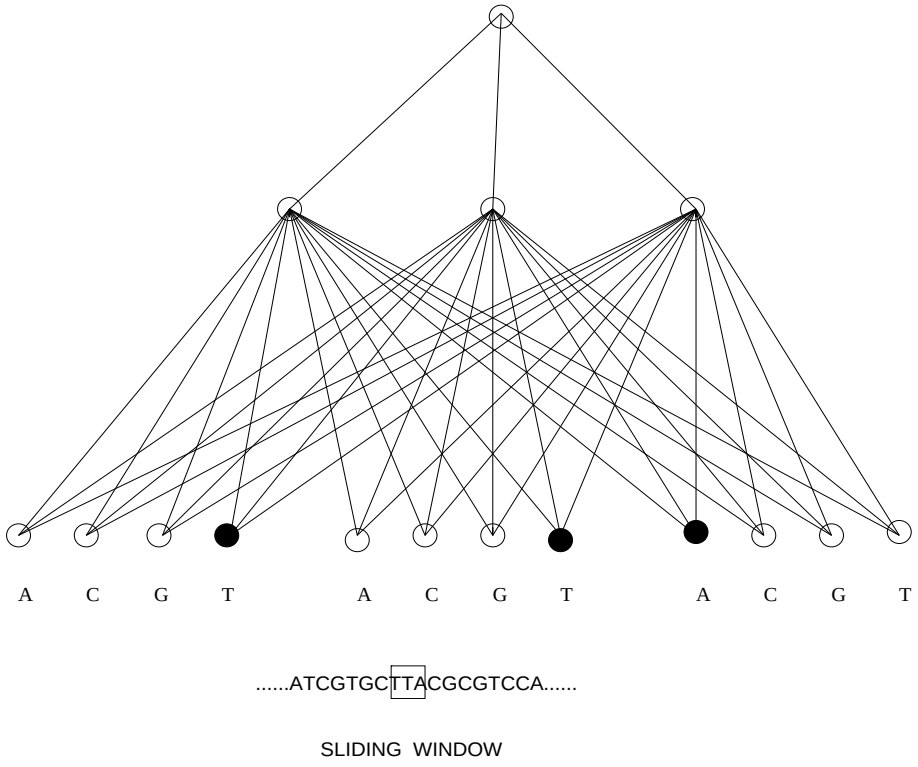

Figure 1: The orthogonal encoding.

The E. Coli promoter is located immediately before the E. Coli gene. Thus, successfully locating the E. Coli promoter conduces to identifying the E. Coli gene. The uncertain characteristics of the E. Coli promoters contribute to the difficulty in the promoter recognition. The E. Coli promoters contain two binding sites to which the E. Coli RNA polymerase, a kind of protein, binds.[17] The two binding sites are the -35 hexamer box and the -10 hexamer box, respectively. The central nucleotides of the two binding sites are roughly 35 bases and 10 bases, respectively, upstream of the transcriptional start site. The transcriptional start site is the first nucleotide of a codon where the transcription begins; it serves as a reference point (position +1). The consensus sequences, i.e., the prototype sequences composed of the most frequently occurring nucleotides in each position, for the -35 binding site and the -10 binding site are `TTGACA` and `TATAAT`, respectively. But none of the promoters can exactly match the two consensus sequences. The average conservation is about 8 nucleotides, meaning that a promoter sequence can match 8 out of 12 nucleotides in the two consensus sequences on average. Figure 2 shows a promoter sequence with the -35 binding site being `TAGCGA` and the -10 binding site being `AAAGAT`. The conservation here includes only 6 nucleotides.

The two consensus regions are separated by a spacer. The length of the spacer

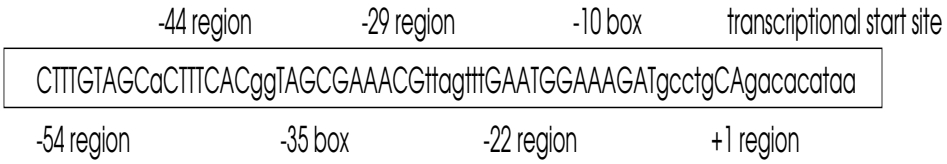

Figure 2: A promoter sequence. The regions are highlighted by upper case letters. The -54 region, -44 region, -35 box, -29 region, -22 region, -10 box, and +1 region are `CTTTGTAGC`, `CTTTCAC`, `TAGCGA`, `AACG`, `GAATGG`, `AAAGAT` and `CA` respectively.

varies from 15 nucleotides to 19 nucleotides. The length of the spacer has an effect on the relative orientation between the -35 region and the -10 region. A spacer of 17 nucleotides is most probable. The promoter sequence in Figure 2 has a spacer of 17 nucleotides. Another spacer between the -10 hexamer box and the transcriptional start site also has a variable length. The most probable length of this spacer is 7 nucleotides. The promoter sequence in Figure 2 has a spacer of 6 nucleotides. Because of the variable spacing, it is not appropriate to use the orthogonal encoding to encode or view a promoter sequence as an n attribute tuple, where n is the length of the promoter sequence. Many promoter sequences have the pyrimidine (`C` or `T`) at the position -1 (one nucleotide upstream of the transcriptional start site), while the purine (`A` or `G`) is at the transcriptional start site (position +1). The +1 region includes the nucleotides at the position -1 and the transcriptional start site. The promoter sequence in Figure 2 has a nucleotide `C` at the position -1 and a nucleotide `A` at the transcriptional start site.

Other than these salient features at the two binding sites and the transcriptional start site, some non-salient features in other regions have been reported in the literature. In Galas *et al.* (1985)[10] and Mengeritsky *et al.* (1987),[21] a pattern matching method was applied to the characterization of E. Coli promoters. Some weak motifs were found around the -44 and the -22 regions. In Cardon *et al.* (1992),[6] the Expectation-Maximization algorithm was used to locate the two binding sites within unaligned promoter sequences. As many as 8 nucleotides within the spacer region between the two binding sites were observed to have contributions to the specificity of the promoter sequences. Recently, Pedersen and Engelbrecht[26] adopted a neural network to characterize E. Coli promoters. The significance of a feature was measured by the decrease in the maximum correlation coefficient when all features except that feature were fed into the neural network. By using this method, the features in the +1, -10 (the -10 Box), -22, -35 (the -35 Box), -44 regions were observed. It is interesting to observe that these features are spaced regularly with a period of 10–11 nucleotides corresponding to one helical turn. This phenomenon suggests that the RNA polymerase makes contact with the promoter on one face of the DNA. Subsequently, the characterization of E. Coli promoter sequences was carried out by the hidden Markov model.[25] It was observed that the position of the -35 box relative to the transcriptional start site is very flexible. More recently, a

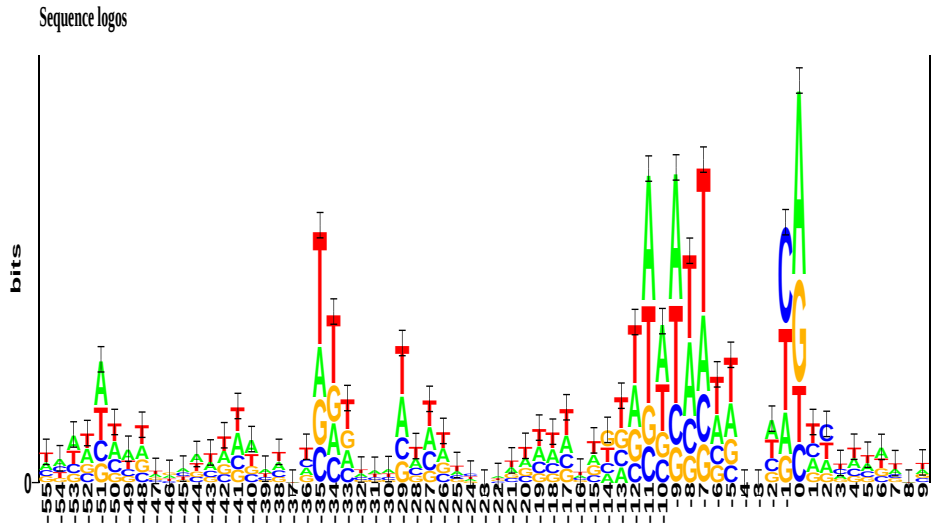

Figure 3: The sequence logos of 291 E. Coli promoter sequences. Position 0 is the transcriptional start site.

clustering analysis was carried out on a larger set of E. Coli promoter sequences containing 441 promoters.[23] Weak motifs were observed in the -54 region.

The above features were also revealed by the sequence logos described in Schneider *et al.* (1990).[29] Figure 3 displays the sequence logos of 291 E. Coli promoters aligned according to the transcriptional start site.* Given a set of aligned sequences, the sequence logos measure the non-randomness of each position l independently by the Shannon entropy for that position:

$$R(l) = log_2(|\mathcal{D}|) - (-\sum_{b \in \mathcal{D}} f(b,l) log_2 f(b,l)), \tag{1}$$

where $|\mathcal{D}|$ is the cardinality of the 4-letter DNA alphabet $\mathcal{D}$, $log_2(|\mathcal{D}|) = 2$ is the maximum uncertainty at any given position, $-\sum_{b \in \mathcal{D}} f(b,l) log_2 f(b,l)$ is the Shannon entropy of position l, and $f(b,l)$ is the frequency of base b at position l.

The height of the logos at each position represents the non-randomness of that position. The size of each base at each position of the logos is proportional to the

*The sequence logos were produced by using the software available at http://www-lecb.ncifcrf.gov/~toms/delila.html

frequency of the base. From Figure 3, we can see some features in the +1, -10, -22, -29, -35, -44, and -54 regions.

Based on the characteristics of the E. Coli promoter sequences reported in the literature, we explore two methods for extracting features in the following regions in promoters: the -54 region (9 nucleotides long), the -44 region (7 nucleotides long), the -35 region (6 nucleotides long), the -22 region (6 nucleotides long), the -10 region (6 nucleotides long), and the +1 region (2 nucleotides long) (Figure 2). The first method is the Maximal Dependence Decomposition (MDD) technique and the second one is a Motif based method. Because the -29 region is 4 nucleotides long, and the 4 nucleotide long motif in the -29 region is not statistically significant, we apply only the MDD method to extracting features in the -29 region.

2.3. *MDD technique*

The MDD technique was first proposed to detect the splice site in human genomic DNA in the gene prediction software GENSCAN.[5] It was later adopted in the latest version of the gene prediction software MORGAN.[27] MDD was derived from the Position Weight Matrix (PWM) described in Staden (1984)[30] to overcome the limitation of the consensus sequence by modeling the nucleotide distribution at each position. One disadvantage of PWM is that it assumes the positions are independent. This disadvantage was removed in the Weight Array Model (WAM),[35] which generalizes PWM by allowing for the dependencies between the adjacent positions.

WAM is essentially a first order Markov chain (conditional probability on the upstream adjacent nucleotide) which can be further generalized by the second-order Markov chain, third-order Markov chain, etc. However, the more dependencies one tries to model, the more free parameters the model has, thus requiring more training data to appropriately estimate the parameters in the model. In general, there is a danger when one tries to use more complex models, which have more free parameters, and does not have enough training data to estimate the free parameters. For instance, suppose we have 291 promoter sequences available to estimate the parameters $P_i(X)$ in PWM where $P_i(X)$ represents the probability distribution of X in position i, $X \in \mathcal{D}$ and $\mathcal{D}$ is the 4-letter DNA alphabet. Equivalently we have roughly $291/4 = 72$ promoter sequences available to estimate the parameters $P_i(X_i|X_{i-1})$ in WAM, where $P_i(X_i|X_{i-1})$ represents the conditional probability distribution in position i given X_{i-1} in position $i-1$ which is the upstream neighbor of X_i in position i, X_{i-1}, $X_i \in \mathcal{D}$. 72 promoter sequences are too few to reliably estimate the free parameters.

MDD provides a flexible solution to the above problem by iteratively clustering the dataset based on the most significant adjacent or non-adjacent dependencies. It essentially models the first-order, second-order, third-order and even higher order dependencies depending on the amount of training data available. More specifically, MDD works as follows. Given a set D of aligned sequences, it first chooses the consensus nucleotide K_i at each position i. In our case, the set D includes all

the positive training sequences. Then the chi square statistic χ_{ij} is calculated to measure the dependencies between K_i and the nucleotides at position j ($i \neq j$). If no significant dependencies are detected, then the simple PWM is used. If there are significant dependencies detected, but the dependencies exist only between adjacent positions, then WAM is used. Otherwise the MDD procedure is carried out.

The MDD procedure is an iterative process: calculate the sum $S_i = \sum_{i \neq j} \chi_{ij}$ for each i; select the position m such that S_m is maximal, and decompose the dataset D into two disjoint subsets, D_m (containing all sequences that have consensus nucleotide K_m at position m) and $D - D_m$ (containing all sequences that do not have consensus nucleotide K_m at position m).

The MDD procedure is then applied recursively to D_m and $D - D_m$ respectively until any one of the following conditions holds: no further decomposition is possible, no significant dependencies between positions in the resulting subsets exist, or the number of sequences in the resulting subsets is below a threshold so that reliable estimation of parameters is not possible after further decomposition.

We apply the MDD method to the -54 region, the -44 region, the -35 region, the -29 region, the -22 region, the -10 region, and the +1 region respectively, of the training promoter sequences. As a result, the -44 region and the +1 region are modeled by PWM, and one level decomposition is carried out in the other regions. To align subsequences in the -35 region, the -29 region, the -22 region and the -10 region, we need to locate the two binding sites in the E. Coli promoters: the -35 box and the -10 box. Locating the -35 box and the -10 box is done by searching for the best match of 12 consensus nucleotides. When different locations have the same conservations, the best location is chosen according to the following hierarchical criteria:[17] homogeneous conservation in the two regions is preferred; a spacer of 17 nucleotides between the -35 region and the -10 region is preferred; a spacer of 7 nucleotides between the -10 region and the transcriptional start site is preferred.

2.4. *Motif based method*

To calculate the motif feature values of each sequence, we first apply our pattern matching tool, Sdiscovery[31] to the positive training data (i.e., the E. Coli promoter sequences) to find weak motifs (common subsequences) in the -54 region, the -44 region, the -35 region, the -22 region and the -10 region respectively, in these sequences.

Sdiscovery is a two phase process. In phase (i), it creates the generalized suffix tree (GST) from a sample of a given set of sequences, and traverses the GST to find all segments (candidate motifs) that satisfy the minimum length requirement. The set includes all the subsequences in the same region (e.g., the -54 region) of the positive training sequences. In phase (ii), Sdiscovery ranks the candidate motifs according to their occurrence numbers in the sample, and then evaluates the most likely candidate motifs with respect to the entire set of sequences. The occurrence number of a motif (segment) refers to the total number of sequences in the set in which the motif occurs. For each region, the length of the motifs is fixed. In the

study presented here, the length is 6 for the -54, -35 and -10 regions, the length is 5 for the -44 and -22 regions. The minimum occurrence number is 2. The occurrence number of a motif is assigned as the weight of the motif.

3. Basic Classifiers

We have developed three basic classifiers: $Classifier\_0$, $Classifier\_1$ and $Classifier\_2$. Each of the classifiers is a Bayesian neural network. All the training sequences and test sequences are encoded by the high level features as described in the previous section.

The first classifier, $Classifier\_0$, is a Bayesian neural network with 5 hidden units and 9 input units including 7 MDD features and 2 distance features, which are the distance (i.e., the number of nucleotides) between the -35 box and the -10 box and the distance between the -10 box and the transcriptional start site. The second classifier, $Classifier\_1$, is a Bayesian neural network with 5 hidden units and 8 input units including 6 Motif features and 2 distance features. The third classifier, $Classifier\_2$, is a Bayesian neural network with 6 hidden units and 15 input units including the above 7 MDD features, the above 6 Motif features and 2 distance features. The number of hidden units is determined experimentally.

The Bayesian neural network we use has one hidden layer with sigmoid activation functions. The output layer of the neural network has one output unit. The output value is bounded between 0 and 1 by the logistic activation function $f(a) = \frac{1}{1+e^{-a}}$. The neural network is fully connected between the adjacent layers. Figure 4 shows the Bayesian neural network architecture of the $Classifier\_2$.

3.1. *Bayesian neural networks*

The Bayesian neural network is the integration of Bayesian inference and the neural network. In Bayesian inference, a model (e.g. a neural network) M_i consists of a set of free parameters which are viewed as random variables. The *prior* of a model M_i is represented by $P(M_i)$. The *likelihood*, i.e., the probability of the data D given the model M_i, is specified by $P(D|M_i)$. The *posterior probability* of the model M_i is quantified by $P(M_i|D)$. From Bayes' rule, we have:

$$P(M_i|D) = \frac{P(D|M_i)P(M_i)}{P(D)}, \tag{2}$$

where $P(D) = \int P(D|M_i)P(M_i)dM_i$ is a normalizing constant.

In our case, $D = \{\mathbf{x}^{(m)}, t_m\}, m = 1, 2, \ldots, N$, denotes the training dataset, where N is the total number of training sequences in D, $\mathbf{x}^{(m)}$ is an input feature vector which contains 9 (8, 15, respectively) input values for $Classifier\_0$ ($Classifier\_1$, $Classifier\_2$, respectively), and t_m is the binary (0/1) target value for the output unit. That is, if $\mathbf{x}^{(m)}$ represents a promoter sequence, t_m is 1; otherwise, t_m is 0. Let $\mathbf{x}$ denote an input feature vector for a DNA sequence, which could be a training sequence or a test sequence. Given the architecture $\mathbf{A}$ and the weights $\mathbf{w}$ of the neural network, the output value y can be uniquely determined from the input

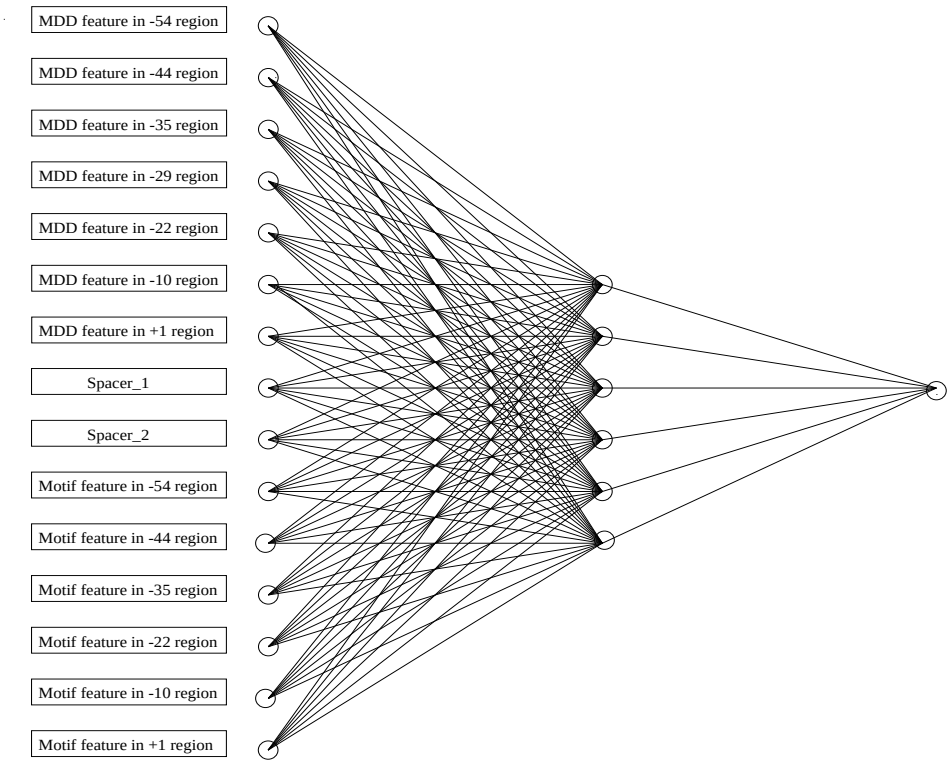

Figure 4: The Bayesian neural network architecture of the $Classifier\_2$. Spacer_1 is the distance between the -35 box and the -10 box. Spacer_2 is the distance between the -10 region and the transcriptional start site.

vector $\mathbf{x}$. The output value $y(\mathbf{x}; \mathbf{w}, \mathbf{A})$ can be interpreted as $P(t = 1|\mathbf{x}, \mathbf{w}, \mathbf{A})$, i.e., the probability that $\mathbf{x}$ represents a promoter sequence given $\mathbf{x}, \mathbf{w}, \mathbf{A}$.

The likelihood, i.e., the probability of the data given the model is calculated by[20]

$$P(D|\mathbf{w}, \mathbf{A}) = \Pi_{m=1}^{N} y^{t_m} (1 - y)^{1-t_m} = expG(D|\mathbf{w}, \mathbf{A}), \tag{3}$$

where

$$G(D|\mathbf{w}, \mathbf{A}) = \sum_{m=1}^{N} t_m logy + (1 - t_m) log(1 - t_m). \tag{4}$$

Equation (3) is the objective function and maximized (or its negation is minimized) in the non-Bayesian neural network training process, which is the maximum likelihood estimation based method, and assumes all possible weights are equally likely.

In the non-Bayesian neural network that contains only one model, the weight

decay is often used to avoid overfitting on the training data and poor generalization on the test data by adding a term $\frac{\alpha}{2}\sum_{i=1}^{k} w_i^2$ to the objective function, where α is the weight decay parameter (hyperparameter), $\sum_{i=1}^{k} w_i^2$ is the sum of the square of all the weights of the neural network, and k is the number of weights. This object function is minimized, to penalize the neural network with weights of large magnitudes, penalizing the over-complex model and favoring the simple model. But there is no precise way to specify the appropriate value of α, which is often tuned offline.

In the Bayesian neural network, the hyperparameter α is interpreted as the parameter of the model, and optimized online during the Bayesian learning process. The weight decay term $\frac{\alpha}{2}\sum_{i=1}^{k} w_i^2$ can be scaled to $(\frac{\alpha}{2\pi})^{\frac{k}{2}} exp(-\frac{\alpha}{2}\sum_{i=1}^{k} w_i^2)$ and interpreted as a prior probability of the weight vector $\mathbf{w}$ in the Gaussian distribution with zero mean and standard deviation $\sqrt{\frac{1}{\alpha}}$. Thus, larger neural network weights are less probable. The Bayesian neural network further generalizes the previous weight decay term by associating the weights in different layers with different variances. Thus, the hyperparameter becomes a vector α. Let $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ represent the vector α, where α_i is associated with a group of weights w_j^i, $j = 1, 2, \ldots, q_i$, $i = 1, 2, \ldots, n$, q_i is the number of weights associated with α_i. Let E_W^c denote $\frac{1}{2}\sum_{j=1}^{q_c}(w_j^c)^2$, $c = 1, 2, \ldots, n$. Then the prior is:[20]

$$P(\mathbf{w}|\alpha, \mathbf{A}) = \frac{exp(-\sum_{c=1}^{n} \alpha_c E_W^c)}{Z_W}, \tag{5}$$

where $Z_W = \int exp(-\sum_{c=1}^{n} \alpha_c E_W^c) d\mathbf{w}$ is the normalizing constant. From (4) and (5), we can get a posterior probability[20]

$$P(\mathbf{w}|D, \alpha, \mathbf{A}) = \frac{exp(-\sum_{c=1}^{n} \alpha_c E_W^c + G(D|\mathbf{w}, \mathbf{A}))}{Z_M}, \tag{6}$$

where $Z_M = \int exp(-\sum_{c=1}^{n} \alpha_c E_W^c + G(D|\mathbf{w}, \mathbf{A})) d\mathbf{w}$ is also a normalizing constant.

3.2. *The training phase*

For the three basic classifiers we have developed, in the training phase, the input feature vector $\mathbf{x}^{(m)}$ is calculated for each training sequence. For $Classifier\_0$, $\mathbf{x}^{(m)}$ contains only MDD feature values; for $Classifier\_1$, $\mathbf{x}^{(m)}$ contains only Motif feature values; for $Classifier\_2$, $\mathbf{x}^{(m)}$ contains both MDD feature values and Motif feature values.

Given a set of training sequences, the MDD feature values of each training sequence are calculated as follows. First, the MDD technique described in Section 2.3 is applied to all the positive training data (i.e., the E. Coli promoter sequences). The results are probability matrices for the -44 region and the +1 region as well as conditional probability matrices for the -54 region, the -35 region, the -29 region, the -22 region, and the -10 region. Secondly, for all the positive and negative training sequences, these matrices are used to calculate the MDD feature values of each training sequence. In particular, the feature value of a subsequence $X =$

$x_1x_2x_3\ldots x_n$, where $x_i \in \mathcal{D}$, $i = 1, 2, \ldots, n$, in the -44 region or the +1 region in a training sequence S is calculated by

$$P(X) = p_1(x_1)p_2(x_2)p_3(x_3)\ldots p_n(x_n), \tag{7}$$

where $p_i(x_i)$ is the probability of x_i in position i. For example, suppose the probability matrix in the +1 region of the positive training sequences is

$$\begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{array}{c} \begin{array}{cc} Position-1 & Position+1 \end{array} \\ \left(\begin{array}{cc} 0.168 & 0.481 \\ 0.392 & 0.100 \\ 0.110 & 0.271 \\ 0.330 & 0.148 \end{array}\right) \end{array}$$

Then, $P(TG) = 0.330 * 0.271 = 0.089$.

The feature value of a subsequence $X = x_1x_2x_3\ldots x_n$ in the other regions in the training sequence S is calculated by

$$P(X) = \begin{cases} p_m(K_m)cp_1^m(x_1)\ldots cp_{m-1}^m(x_{m-1})cp_{m+1}^m(x_{m+1})\ldots cp_n^m(x_n) & \text{if } K_m = x_m \\ p_m(x_m)cp_1^{\overline{m}}(x_1)\ldots cp_{m-1}^{\overline{m}}(x_{m-1})cp_{m+1}^{\overline{m}}(x_{m+1})\ldots cp_n^{\overline{m}}(x_n) & \text{otherwise.} \end{cases} \tag{8}$$

where $p_m(K_m)$($p_m(x_m)$, respectively) represents the probability of K_m(x_m, respectively) in position m, $cp_i^m(x_i)$($cp_i^{\overline{m}}(x_i)$, respectively), $i = 1, 2, \ldots, m-1, m+1, m+2, \ldots, n$, represents the conditional probability of x_i in position i given $x_m = K_m$($x_m \neq K_m$, respectively).

Given a set of training sequences, the motif feature values in the -54, -44, -35, -22, -10 regions of each training sequence are calculated as follows. First, the motif based method described in Section 2.4 is applied to the -54, -44, -35, -22, -10 regions of all the positive training data. The result is five sets of motifs in the -54, -44, -35, -22, -10 regions. Secondly, for each region of a training sequence, the subsequence in that region is matched against the motifs in that region. If there are matched motifs, the feature value of that region for the training sequence is the maximum weight of the matched motifs; otherwise the feature value is assigned to 0.

The motif feature value in the +1 region of a training sequence is assigned to 1 if the nucleotide at position -1 is the pyrimidine (`C` or `T`) and the nucleotide at the transcriptional start site is the purine (`A` or `G`); otherwise it is assigned to 0.

The Bayesian training of neural networks is an iterative procedure. In the implementation of the Bayesian neural network that we adopt[†], each iteration involves two level inferences. At the first level, given the value of hyperparameter α, which is initialized to the random value during the first iteration, we can infer the Most Probable value of the weight vector $\mathbf{w}^{mp}$ corresponding to the maximum of $P(\mathbf{w}|D, \alpha, \mathbf{A})$ by the neural network training, which minimizes $\sum_{c=1}^{n} \alpha_c E_W^c - G(D|\mathbf{w}, \mathbf{A})$. The Bayes' rule for the first level inference is:[20]

[†]Software available at http://wol.ra.phy.cam.ac.uk/pub/mackay/README.html

$$P(\mathbf{w}|D,\alpha,\mathbf{A}) = \frac{P(D|\mathbf{w},\mathbf{A})P(\mathbf{w}|\alpha,\mathbf{A})}{P(D|\alpha,\mathbf{A})}, \tag{9}$$

where $P(D|\mathbf{w},\mathbf{A})$, $P(\mathbf{w}|\alpha,\mathbf{A})$, and $P(\mathbf{w}|D,\alpha,\mathbf{A})$ are given by (3), (5) and (6) respectively, and D is the set of vectors of feature values extracted from the training sequences.

At the second level, the hyperparameter α is optimized. The Bayes' rule for the second inference level is:[20]

$$P(\alpha|D,\mathbf{A}) = \frac{P(D|\alpha,\mathbf{A})P(\alpha|\mathbf{A})}{P(D|\mathbf{A})}. \tag{10}$$

Because of the lack of the prior knowledge of $P(\alpha|\mathbf{A})$, we assume $P(\alpha|\mathbf{A})$ to be a constant value and is ignored. Since the normalizing factor $P(D|\mathbf{A})$ is also a constant value, the value of α maximizing a posterior $P(\alpha|D,\mathbf{A})$ can be inferred by maximizing $P(D|\alpha,\mathbf{A})$, which is the normalizing factor in (9), and thus is equal to $\int P(D|\mathbf{w},\mathbf{A})P(\mathbf{w}|\alpha,\mathbf{A})d\mathbf{w}$. The integrand is approximated as a Gaussian centered around $\mathbf{w}^{mp}$ so that $P(D|\alpha,\mathbf{A})$ can be maximized. The new hyperparameter value α is then used in the next iteration. The process iterates a number of times as specified by the user. The iteration number we use is 9.

3.3. *The classification phase*

In the classification phase, for the three basic classifiers we have developed, the output of a Bayesian neural network, y, is based on all models rather than on one model. Each model is weighted by its posterior probability. That is, for a new input vector $\mathbf{x}$ of an unlabeled test sequence which is calculated by the same method as described in Section 3.2, the output, y, is given by the marginalisation of the output of each model, $p(t=1|\mathbf{x},\mathbf{w},\alpha,\mathbf{A})$, weighted by its posterior $p(\mathbf{w},\alpha|D,\mathbf{A})$.[20] Thus,

$$y = P(t=1|\mathbf{x},D,\mathbf{A}) = \int P(t=1|\mathbf{x},\mathbf{w},\alpha,\mathbf{A})P(\mathbf{w},\alpha|D,\mathbf{A})d\mathbf{w}d\alpha, \tag{11}$$

where

$$P(\mathbf{w},\alpha|D,\mathbf{A}) = P(\mathbf{w}|\alpha,D,\mathbf{A})P(\alpha|D,\mathbf{A}). \tag{12}$$

The output of a Bayesian neural network, $P(t=1|\mathbf{x},D,\mathbf{A})$, is the probability that the unlabeled test sequence is a promoter. If it is greater than a predetermined positive threshold, the test sequence is classified as a promoter; if it is less than a predetermined negative threshold, the test sequence is classified as a non-promoter; otherwise the test sequence is unidentified. The positive threshold and negative threshold we use are 0.505 and 0.495, respectively.

4. Combination of Basic Classifiers

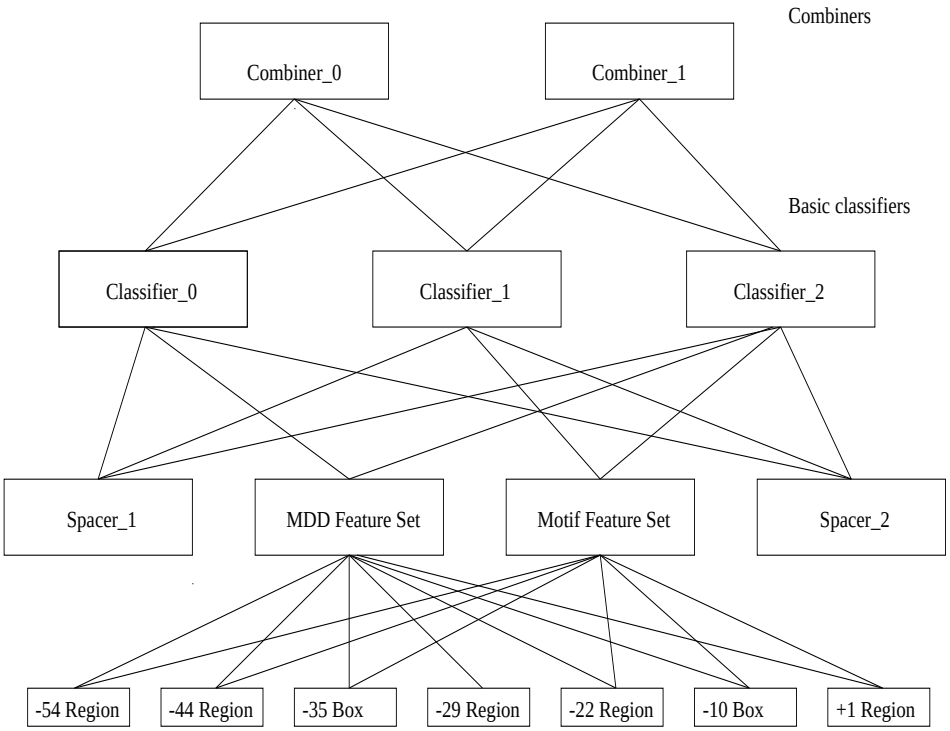

Figure 5: Our classification scheme.

The three basic classifiers described in the previous section are combined into one classifier in the second level (Figure 5). We explore two methods for combining the three basic classifiers: $Combiner\_0$ and $Combiner\_1$.

The $Combiner\_0$ employs an unweighted voter. Let $output_i$, $0 \leq i \leq 2$, be the output value of the $Classifier\_i$. If the three basic classifiers agree on the classification results (Promoter, Non-promoter, Unidentified), the final result will be the same as the results of the three classifiers; if two classifiers agree on the classification results, the final result will be the same as the results of these two classifiers; if none of the classifiers agrees on the classification results, the final result will be the same as the result of a classifier whose $\min(1 - output_i, output_i)$ is minimal; otherwise the final result is unidentified.

$Combiner\_1$ employs a weighted voter. Its output is the weighted sum of the outputs of the three basic classifiers. That is, let w_i represent the weight of $Classifier\_i$, where the weight is the precision rate of $Classifier\_i$ in the training phase. The output of $Combiner\_1$ is given by

$$\frac{1}{w_0 + w_1 + w_2} \sum_{i=0}^{2} w_i * output_i. \tag{13}$$

Table 1: Performance of the three basic classifiers.

	$Classifier\_0$	$Classifier\_1$	$Classifier\_2$
Precision Rate	87%	89%	90%
Specificity	90%	90.67%	89.67%
Sensitivity	83.85%	86.94%	89.35%

Note that if we assign equal weights to the three basic classifiers, then $Combiner\_1$ is reduced to $Combiner\_0$.

5. Experiments and Results

5.1. *Data*

In this study, we adopted E. Coli promoter sequences taken from the recent E. Coli promoter compilation.[17] There were 300 E. Coli promoters aligned by the transcriptional start site. When there was uncertainty about transcriptional start sites, the most probable nucleotide `A` was chosen as the transcriptional start site. We trimmed each promoter sequence to 65 nucleotides including nucleotides from -55 (55 nucleotides upstream of the transcriptional start site) to +10 (10 nucleotides downstream of the transcriptional start site). This gave us 296 promoter sequences. As promoter alignment is a crucial point in the promoter recognition, the uncertainty about the transcriptional start sites was further eliminated by manually checking the transcriptional start sites of the 296 promoter sequences with the recent E. Coli promoter compilation.[23] The final set contained 291 promoter sequences, which were used as the positive data in the classification study.

The negative data was retrieved from the well known machine learning data repository at the University of California at Irvine at: `http://www.ics.uci.edu/AI/ML/MLDBRepository.html`. There were 53 non-promoter DNA sequences, each being 57 nucleotides long. We concatenated these DNA sequences to one sequence S with 3021 nucleotides. We then randomly chose 300 subsequences from S, each being 65 nucleotides long, and used them as the negative data.

We did not use the 53 promoter sequences individually because the 53 promoter sequences were retrieved from the early E. Coli promoter compilation[12] where the alignment was done according to the -35 region, instead of the biologically meaningful transcriptional start site. Furthermore the amount of data was so small that the classification result on those data may not be statistically significant.[28]

5.2. *Results*

Table 1 gives the ten-fold cross validated classification results of the three basic classifiers and Table 2 gives the results of the combined classifiers. In ten-fold cross validation, the dataset containing both the positive data (Promoters) and

Table 2: Performance of the two combined classifiers.

	*Combiner*_0	*Combiner*_1
Precision Rate	91.3%	92.2%
Specificity	92.67%	92%
Sensitivity	89.7%	92.44%

the negative data (Non-promoters) was randomly split into ten mutually exclusive folds $D_1, D_2, \ldots, D_{10}$ of approximately equal size. Each Bayesian neural network was trained and tested ten times. During the ith time, it was trained on $D - D_i$, and tested on D_i. We allocated the data in such a way that the training dataset $D - D_i$ (the test dataset D_i respectively) has approximately $\frac{9}{10}$ ($\frac{1}{10}$, respectively) positive data and $\frac{9}{10}$ ($\frac{1}{10}$, respectively) negative data. The average over the ten tests was taken.

We use the *precision rate* to measure the performance of the studied classifiers. The precision rate is defined as

$$\frac{C}{N} * 100\% \tag{14}$$

where C is the number of test sequences classified correctly, N is the total number of test sequences. A *false positive* is a non-promoter test sequence that was misclassified as a promoter sequence. A *true positive* is a promoter test sequence that was also classified as a promoter sequence. The *specificity* is defined as

$$(1 - \frac{N_{fp}}{N_{ng}}) * 100\% \tag{15}$$

where N_{fp} is the number of false positives and N_{ng} is the total number of negative test sequences. The *sensitivity* is defined as

$$\frac{N_{tp}}{N_{po}} * 100\% \tag{16}$$

where N_{tp} is the number of true positives and N_{po} is the total number of positive test sequences.

From Table 2, we can see that *Combiner*_0 and *Combiner*_1 outperform the three basic classifiers. The *Combiner*_1 gives the best precision rate 92.2%. The reason that *Combiner*_0 has a higher precision rate than that of any one of the three basic classifiers can be explained by the Bernoulli model. For instance, assume that the three basic classifiers have the same precision rate of 88% and make classification errors completely independently. Then the *Combiner*_0 makes a classification error when more than one classifier make errors at the same time. Thus, the precision rate of the unweighted voter of the three basic classifiers would be given by:

$$100\% - (\binom{3}{3}(1 - 88\%)^3 + \binom{3}{2}88\%(1 - 88\%)^2) = 96\%. \tag{17}$$

Table 3: The complementarity of $Combiner\_0$ and $Combiner\_1$.

Classification results	Percentage of the test sequences
$Combiner\_0$ and $Combiner\_1$ agreed & both were correct	88.8%
$Combiner\_0$ and $Combiner\_1$ agreed & both were wrong	5.4%
$Combiner\_0$ and $Combiner\_1$ disagreed & $Combiner\_0$ was correct	2.4%
$Combiner\_0$ and $Combiner\_1$ disagreed & $Combiner\_1$ was correct	3.4%
$Combiner\_0$ and $Combiner\_1$ disagreed & both were wrong	0.0%

The practical precision rate is a bit lower. The reason is that the $Classifier\_0$, $Classifier\_1$ and $Classifier\_2$ can not make errors completely independently.

Table 3 illustrates the complementarity between the $Combiner\_0$ and $Combiner\_1$. When $Combiner\_0$ and $Combiner\_1$ agree, the classification has a higher likelihood of being correct. When both agree, the probability that the classification is correct is given by 88.8%/(88.8%+5.4%)=94.2%. From Table 3, we can see that when $Combiner\_0$ and $Combiner\_1$ disagree, the probability that one is correct is 100%.

6. Conclusion

We have proposed a two-level ensemble of classifiers to recognize E. Coli promoter sequences. The first-level classifiers include three Bayesian neural networks trained on three different feature sets. The outputs of the first-level classifiers are combined in the second-level to give the final result. A recognition rate of 92.2% was achieved. This result is better than the previous work[13] on a similar dataset. Currently we are extending the approach to classify protein sequences and to recognize the full gene structure.

Acknowledgements

The sequence logos software was developed by Dr. Thomas Schneider. We thank Dr. David Mackay for sharing the Bayesian neural network software with us. We also thank Dr. Shlomit Lisser and Dr. Hanah Margalit for providing the promoter sequences used in the paper.

References

[1] Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. Gapped Blast and PSI-Blast: A new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, 1997.

[2] Apostolico, A. and Giancarlo, R. Sequence alignment in molecular biology. *Journal of Computational Biology*, 5(2):173–196, 1998.

[3] Brazma, A., Jonassen, I., Eidhammer, I., and Gilbert, D. Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, 5(2):279–305, 1998.

[4] Brunak, S., Engelbrecht, J., and Knudsen, S. Prediction of human mRNA donor and acceptor sites from the DNA sequence. *Journal of Molecular Biology*, 220(1):49–65, 1991.

[5] Burge, C. and Karlin, S. Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology*, 268(1):78–94, 1997.

[6] Cardon, L. R. and Stormo, G. D. Expectation maximization algorithm for identifying protein-binding sites with variable lengths from unaligned DNA fragments. *Journal of Molecular Biology*, 223(1):159–170, 1992.

[7] Craven, M. W. and Shavlik, J. W. Machine learning approaches to gene recognition. *IEEE Expert*, 9(2):2–10, 1994.

[8] Dietterich, T. G. Machine learning research: Four current directions. *AI magazine*, 18(4):97–136, 1997.

[9] Frenkel, K. A. The human genome project and informatics. *Communications of the ACM*, 34(11):41–51, 1991.

[10] Galas, D. J., Eggert, M., and Waterman, M. S. Rigorous pattern-recognition methods for DNA sequences: Analysis of promoter sequences from *Escherichia Coli*. *Journal of Molecular Biology*, 186(1):117–128, 1985.

[11] Haussler, D. A brief look at some machine learning problems in genomics. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 109–113, 1997.

[12] Hawley, D. K. and McClure, W. R. Compilation and analysis of *Escherichia Coli* promoter DNA sequences. *Nucleic Acids Research*, 11(8):2237-2255, 1983.

[13] Hirsh, H. and Noordewier, M. Using background knowledge to improve inductive learning of DNA sequences. In *Proceedings of the Tenth Conference on Artificial Intelligence for Applications*, pages 351–357, 1994.

[14] Hirst, J. D. and Sternberg, M. J. E. Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks. *Biochemistry*, 31(32):7211–7218, 1992.

[15] Krogh, A., Brown, M., Mian, I. S., Sjolander, K., and Haussler, D. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, 1994.

[16] Kulp, D., Haussler, D., Reese, M. G., and Eeckman, F. H. A generalized hidden Markov model for the recognition of human genes in DNA. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pages 134–142, 1996.

[17] Lisser, S. and Margalit, H. Compilation of E. Coli mRNA promoter sequences. *Nucleic Acids Research*, 21(7):1507–1516, 1993.

[18] Mackay, D. J. C. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

[19] Mackay, D. J. C. A practical Bayesian framework for Backprop networks. *Neural Computation*, 4(3):448–2472, 1992.

[20] Mackay, D. J. C. The evidence framework applied to classification networks. *Neural Computation*, 4(5):698–714, 1992.

[21] Mengeritsky, G. and Smith, T. F. Recognition of characteristic patterns in sets of functionally equivalent DNA sequences. *Computer Applications in the Biosciences*, 3(3):223–227, 1987.

[22] Neal, R. M. *Bayesian Learning for Neural Network, Lecture Notes in Statistics*,

volume 118. Springer-Verlag, New York, 1996.

[23] Ozoline, O. N., Deev, A. A., and Arkhipova, M. V. Non-canonical sequence elements in the promoter structure. cluster analysis of promoters recognized by *Escherichia Coli* RNA polymerase. *Nucleic Acids Research*, 25(23):4703–4709, 1997.

[24] Pearson, W. R. and Lipman, D. J. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the USA*, 85(8):2444–2448, 1988.

[25] Pedersen, A. G., Baldi, P., Brunak, S., and Chauvin, Y. Characterization of prokaryotic and eukaryotic promoters using hidden Markov models. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pages 182–191, 1996.

[26] Pedersen, A. G. and Engelbrecht, J. Investigations of *Escherichia Coli* promoter sequences with artificial neural networks: New signals discovered upstream of the transcriptional startpoint. In *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pages 292–299, 1995.

[27] Salzberg, S. A decision tree system for finding genes in DNA. Technical Report CS-97-03, Johns Hopkins Unversity, 1997.

[28] Salzberg, S. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1:317–327, 1997.

[29] Schneider, T. D. and Stephens, R. M. Sequence logos: A new way to display consensus sequences. *Nucleic Acids Research*, 18(20):6097-6100, 1990.

[30] Staden, R. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Research*, 12(1):505–519, 1984.

[31] Wang, J. T. L., Marr, T. G., Shasha, D., Shapiro, B. A., and Chirn, G. Discovering active motifs in sets of related protein sequences and using them for classification. *Nucleic Acids Research*, 22(14):2769–2775, 1994.

[32] Wang, J. T. L., Marr, T. G., Shasha, D., Shapiro, B. A., Chirn, G., and Lee, T. Y. Complementary classification approaches for protein sequences. *Protein Engineering*, 9(5):381–386, 1996.

[33] Wang, J. T. L., Shapiro, B. A., and Shasha, D. (eds.). *Pattern Discovery in Biomolecular Data: Tools, Techniques and Applications*, Oxford University Press, New York, 1999.

[34] Wu, C. Artificial neural networks for molecular sequence analysis. *Computers and Chemistry*, 21(4):237–256, 1997.

[35] Zhang, M. O. and Marr, T. G. A weight array method for splicing signal analysis. *Computer Applications in the Biosciences*, 9(5):499–509, 1993.

[36] Zhang, X., Mesirov, J. P., and Waltz, D. L. Hybrid system for protein secondary structure prediction. *Journal of Molecular Biology*, 225(4):1049–1063, 1992.