Mining Biomolecular Data Using Background Knowledge and Artificial Neural Networks

Qicheng Ma* Jason T. L. Wang† James R. Gattiker‡

Abstract

Biomolecular data mining is the activity of finding significant information in protein, DNA and RNA molecules. The significant information may refer to motifs, clusters, genes, protein signatures and classification rules. This chapter presents an example of biomolecular 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. To enhance the recognition rate, we use the background knowledge (i.e., the characteristics of the promoter sequences) and employ new techniques to extract high-level features from the sequences. We also use an expectation-maximization (EM) algorithm to locate the binding sites of the promoter sequences. Empirical study shows that a precision rate of 95% is achieved, indicating an excellent performance of the proposed approach.

1 Introduction

As a result of the ongoing Human Genome Project [9], DNA, RNA and protein data are accumulated at a speed growing at an exponential rate. Mining these biomolecular data to extract significant information becomes extremely important in accelerating genome processing [32]. Classification, or supervised learning, is one of the major data mining processes. Classification is to classify a set of data into two or more categories. When there are only two categories, it is called *binary classification*. In this chapter we focus on binary classification of DNA sequences.

*Department of Computer and Information Science, New Jersey Institute of Technology, Newark, NJ 07102 (qicheng@homer.njit.edu).

†Department of Computer and Information Science, New Jersey Institute of Technology, Newark, NJ 07102 (jason@cis.njit.edu). Corresponding author, phone: (973) 596-3396, fax: (973) 596-5777.

‡Los Alamos National Laboratory, Mail Stop E541, Los Alamos, NM 87544 (gatt@lanl.gov).

In binary classification, we are given some training data including both positive and negative examples. The positive data belong to a target class, whereas the negative data belong 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 chapter.

1.1 Related Work

Table 1 summarizes some work in biomolecular data mining. The first column indicates the "knowledge" to be mined in the biomolecular data, the second column shows the techniques used and the third column provides references. The knowledge to be mined includes DNA sequence signals such as splice sites and promoters, protein sequence classification rules and protein sequence motifs.

Knowledge to be mined	Techniques used	References
Genes in DNA	Hidden Markov model Neural networks	Kulp *et al.* [14] Xu *et al.* [34]
Splice sites in DNA	Pattern matching Markov chain	Wang *et al.* [31] Salzberg [26]
Promoters in DNA	Neural networks Decision tree	Opitz *et al.* [21] Hirsh *et al.* [11]
Protein classification rules	Hidden Markov model Neural networks	Krogh *et al.* [13] Wu *et al.* [33]
Protein motifs	Minimum description length	Brazma *et al.* [2]

Table 1: A summary of work performed for biomolecular data mining.

In this chapter we propose a two-level approach to recognizing E. Coli promoters in DNA sequences. The first-level classifiers include Bayesian neural networks [18, 20] 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 the proposed 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 by Brunak *et al.* [3], Wang *et al.* [30], and Zhang *et al.* [36]. In [3], Brunak *et al.* exploited the complementary relation between exon and splice sites 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 the regions where there are only small changes in the exon activity. A lower threshold was used to detect the donor site for the regions where the exon activity decreases significantly. Similarly, a lower threshold was used to detect the acceptor site for the regions where the exon activity increases significantly. In [36], Zhang *et al.* developed a hybrid system, which included a neural network, a statistical classifier and a memory-based reasoning classifier, 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 [30], Wang *et al.* studied the complementarity of five classifiers for protein sequence recognition, and proposed an ensemble of the classifiers, which outperformed each individual classifier.

In contrast to the previous work, we apply Bayesian neural networks to recognizing promoters in DNA. The Bayesian neural networks make predictions by marginalization over the weight distribution. Furthermore, the Bayesian neural networks control the model complexity to avoid the overfitting problem [17].

The rest of the chapter 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 chapter.

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 [11].

One of the encoding methods is *orthogonal encoding* [21]. 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

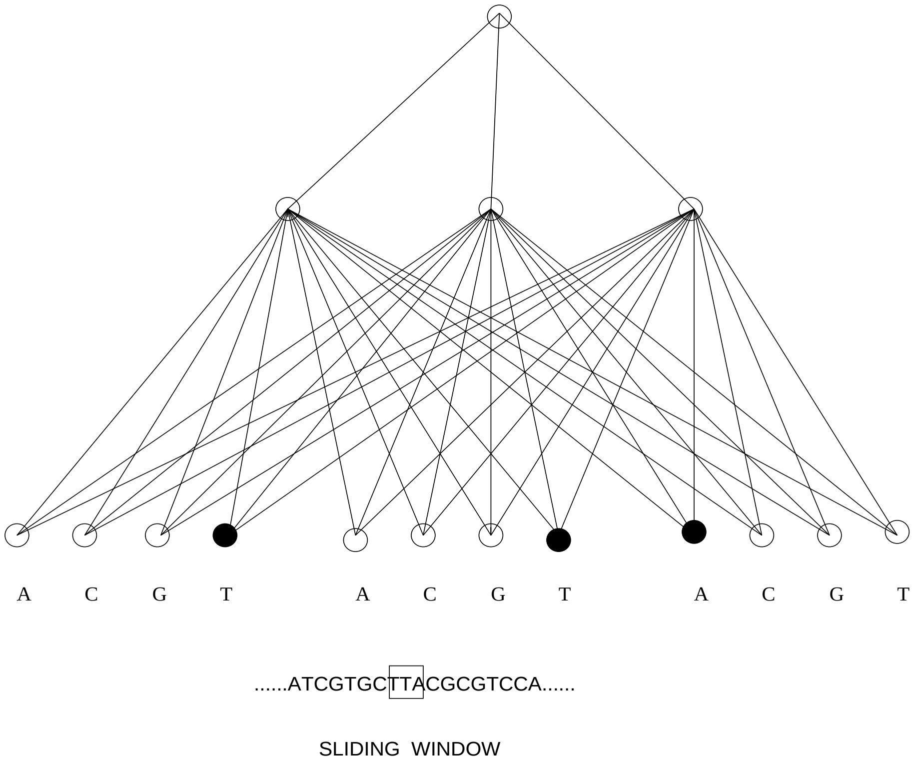

Figure 1: An example of the orthogonal encoding of a DNA sequence.

alphabet $\mathcal{D} = \{$`A, T, G, C`$\}$, or the cardinality of the 20-letter amino acid alphabet $\mathcal{A} = \{$`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 [6, 12]. 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, as proposed in this chapter, 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

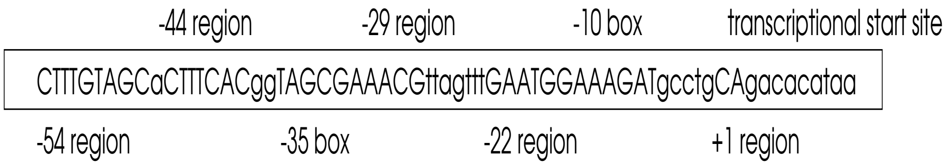

Figure 2: An example promoter sequence. The regions are highlighted by upper case letters. The -54 region, -44 region, -35 box, -29 region, -22 region, -10 box, and +1 region are `CTTTGTAGC`, `CTTTCAC`, `TAGCGA`, `AACG`, `GAATGG`, `AAAGAT` and `CA`, respectively.

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

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 [16]. The two binding sites are the -35 hexamer box and the -10 hexamer box, respectively. Each binding site consists of 6 bases (nucleotides). 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 nucleotide at 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, on average, 8 out of the 12 nucleotides in the two consensus sequences. Figure 2 shows an example 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 binding sites are separated by a spacer. 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.[1] 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.

In addition to these salient characteristics in the two binding sites and the transcriptional start site, there are some non-salient characteristics in other regions. In Galas *et al.* [10] and Mengeritsky *et al.* [19], 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. A weak motif is a subsequence, which occurs frequently in a region. We use the term "weak", since the frequency of a base of the motif is not as significant as the frequency of a base of the consensus sequences occurring in the binding sites. In Cardon *et al.* [5], as many as 8 nucleotides (weak motifs) within the spacer region between the two binding sites were found to have contributions to the specificity of the promoter sequences. Recently, Pedersen and Engelbrecht [24] adopted a neural network to characterize E. Coli promoters. The significance of a weak motif was measured by the decrease in the maximum correlation coefficient when all motifs except that weak motif were fed into the neural network. By using this method, the authors found some weak motifs in the +1, -22, and -44 regions. It is interesting to observe that these weak motifs 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 [23]. It was observed that the position of the -35 box relative to the transcriptional start site is very flexible. More recently, a clustering analysis was carried out on a larger set of E. Coli promoter sequences containing 441 promoters [22]. Some weak motifs were found in the -54 region.

These weak motifs were also revealed by the sequence logos described in Schneider and Stephens [27]. Figure 3 displays the sequence logos of 438 E. Coli promoters aligned according to the transcriptional start site.[2] Given a set of aligned sequences, the sequence logos measure

[1] In general, the distance between the -10 binding site and the transcriptional start site varies from 3 to 11 bases. The distance between the -35 binding site and the -10 binding site varies from 15 to 21 bases. These varying distances render promoter recognition difficult, as both the contents and positions of the binding sites are uncertain. In the following subsection, we present an expectation-maximization (EM) algorithm to locate the binding sites of the promoter sequences.

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

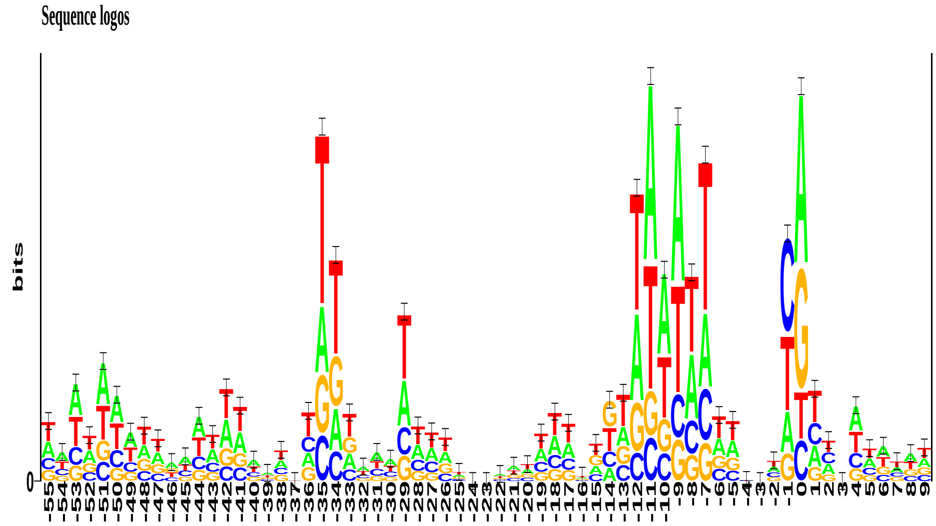

Figure 3: The sequence logos of 438 E. Coli promoter sequences. Position 0 in the figure is the transcriptional start site, which is equivalent to position +1 described in the text. The negative positions in the figure are consistent with those described in the text.

the non-randomness of each position l independently by the Shannon entropy for that position:

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

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 at each position represents the information content of that position. The higher the information content, the less random that position is. The size of each base at each position of the logos is proportional to the frequency of the base. Recall that a weak motif is a frequently occurring subsequence in a region. In the sequence logos, a weak motif consists of positions (bases) with non-zero information content. From Figure 3, it can be seen that some weak motifs exist in the +1, -22, -29, -44, and -54 regions.

Based on the characteristics of the E. Coli promoter sequences reported in the literature, we explore two methods for extracting high-level features in the following regions in the 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) (see 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. In order to calculate these feature values, we must know, as precisely as possible, where these regions are. In the following subsection, we present an expectation-maximization (EM) algorithm for locating the binding sites of promoter sequences. Then we describe our feature extraction methods in detail.

2.3 Locating Binding Sites by the EM Algorithm

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. Locating the -35 box and the -10 box may be done by the EM algorithm [7]. In general, the EM algorithm can be applied for the maximum likelihood estimation when data are incomplete. Locating the binding sites by the EM algorithm was first proposed by Lawrence *et al.* [15]. It was then generalized by Cardon *et al.* [5] to allow for different spacers between the two binding sites. These published methods [1, 5, 15] either assumed that the location of the binding sites is uniformly distributed or attempted to locate one "continuous region" that included the -35 box, the -10 box and a spacer of variable length between them. By contrast, our method does not make the assumption of uniform distribution and considers the binding sites separately from the spacer.

Let T represent the set of promoter sequences in the training set, i.e., T contains all positive training sequences. Let K denote the cardinality of T. For a promoter sequence $S_i \in T$, the length of the spacer between the -10 region and the transcriptional start site, denoted sp_{10}, and the length of the spacer between the -35 region and the -10 region, denoted sp_{35}, are unobserved, though S_i is observed. Specifically, we refer to the positive training sequences as "observed" data since they are given. These observed data are incomplete, because the lengths of the two spacers are not given (the lengths are referred to as "unobserved" or "missing" data). The proposed EM algorithm estimates the model parameters, defined later, from the incomplete data. Then based on the estimates of the model parameters, the algorithm determines the locations of the two binding sites for any DNA sequence.

In general, sp_{10} varies from 3 to 11 and sp_{35} varies from 15 to 21. Assume that the nucleotides at all positions are independent. Then one can use the Position Weight Matrix (PWM) described in Staden [28] to model nucleotides at each position. Each binding site consists of 6 bases. Let $P_{10,j}(x), j = 1, \ldots, 6$, denote the probability of x, $x \in \mathcal{D}$, occurring at position j in the -10 region.

Let $\mathbf{P}_{10}$ denote $(P_{10,1}, \ldots, P_{10,6})$. Let $P_{35,j}(x), j = 1, \ldots, 6$, denote the probability of x occurring at position j in the -35 region. Let $\mathbf{P_{35}}$ denote $(P_{35,1}, \ldots, P_{35,6})$. Let $P_o(x)$ denote the probability of x occurring in the regions outside the two binding sites. Let $P_s(Sp_{10} = m, Sp_{35} = n)$ denote the probability of $Sp_{10} = m$ and $Sp_{35} = n$ where Sp_{10} (Sp_{35}, respectively) is the random variable denoting the distance between the transcriptional start site (the -35 region, respectively) and the -10 region, $m \in \{3, \ldots, 11\}$ and $n \in \{15, \ldots, 21\}$. Let $\theta = (\mathbf{P_{10}}, \mathbf{P_{35}}, P_o)$.

The EM algorithm proceeds iteratively to converge. Each iteration consists of two steps: Expectation step (E step) and Maximization step (M step). The E step calculates the expected complete-data log likelihood, where the expectation is over the distribution of the missing data given the observed data and current estimates of θ. Assume that the promoter sequences in the training set T are independent. Then the E step calculates

$$\begin{aligned} & E_{Sp_{10},Sp_{35}|T,\theta} log P(T, Sp_{10}, Sp_{35}|\theta) \\ = \quad & E_{Sp_{10},Sp_{35}|T,\theta} log(P(T|Sp_{10}, Sp_{35}, \theta) P_s(Sp_{10}, Sp_{35})) \\ = \quad & \textstyle\sum_{i=1}^{K} \sum_{m=3}^{11} \sum_{n=15}^{21} P(Sp_{10} = m, Sp_{35} = n|S_i, \theta) log(\\ & P(S_i|Sp_{10} = m, Sp_{35} = n, \theta) P_s(Sp_{10} = m, Sp_{35} = n)). \end{aligned} \tag{2}$$

Suppose that all promoter sequences in the training set T are 65 nucleotides long and are aligned with respect to the transcriptional start site, which is at position 56. Let $S_{i,j}$ denote the nucleotide at position j of promoter sequence S_i. Define

$$I_{i,j,x} = \begin{cases} 1 & \text{if } S_{i,j} = x \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

Let $O_{i,m,n}(x)$ denote the number of occurrences of the nucleotide x outside the two binding sites of promoter sequence S_i given $Sp_{10} = m$ and $Sp_{35} = n$. Then

$$P(S_i|Sp_{10} = m, Sp_{35} = n, \theta) = \Pi_{j=1}^{6} P_{10,j}(S_{i,49-m+j}) \Pi_{j=1}^{6} P_{35,j}(S_{i,43-m-n+j}) \Pi_{x=\mathtt{A}}^{\mathtt{T}} P_o(x)^{O_{i,m,n}(x)}. \tag{4}$$

From (4), according to Bayes' rule, we have

$$\begin{aligned} & P(Sp_{10} = m, Sp_{35} = n|S_i, \theta) \\ = \quad & \frac{P(S_i|Sp_{10}=m, Sp_{35}=n, \theta) P_s(Sp_{10}=m, Sp_{35}=n)}{P(S_i|\theta)} \\ = \quad & \frac{P(S_i|Sp_{10}=m, Sp_{35}=n, \theta) P_s(Sp_{10}=m, Sp_{35}=n)}{\sum_{m=3}^{11} \sum_{n=15}^{21} P(S_i|Sp_{10}=m, Sp_{35}=n, \theta) P_s(Sp_{10}=m, Sp_{35}=n)} \\ = \quad & \frac{\Pi_{j=1}^{6} P_{10,j}(S_{i,49-m+j}) \Pi_{j=1}^{6} P_{35,j}(S_{i,43-m-n+j}) \Pi_{x=\mathtt{A}}^{\mathtt{T}} P_o(x)^{O_{i,m,n}(x)} P_s(Sp_{10}=m, Sp_{35}=n)}{\sum_{m=3}^{11} \sum_{n=15}^{21} \Pi_{j=1}^{6} P_{10,j}(S_{i,49-m+j}) \Pi_{j=1}^{6} P_{35,j}(S_{i,43-m-n+j}) \Pi_{x=\mathtt{A}}^{\mathtt{T}} P_o(x)^{O_{i,m,n}(x)} P_s(Sp_{10}=m, Sp_{35}=n)}. \end{aligned} \tag{5}$$

Previous applications of the EM algorithm [1, 5, 15] assumed $P_s(Sp_{10}, Sp_{35})$ to be uniformly distributed. The term $P_s(Sp_{10} = m, Sp_{35} = n)$ was deleted from (5). We do not assume

$P_s(Sp_{10}, Sp_{35})$ to be uniformly distributed, as the most probable values of m are 6, 7, 8, and the most probable values of n are 16, 17, 18. Substituting (4) and (5) into (2), we have

$$K \sum_{j=1}^{6} \sum_{x=\mathtt{A}}^{\mathtt{T}} f_{10,j}(x) log P_{10,j}(x) + K \sum_{j=1}^{6} \sum_{x=\mathtt{A}}^{\mathtt{T}} f_{35,j}(x) log P_{35,j}(x) +$$

$$K(65-12) \sum_{x=\mathtt{A}}^{\mathtt{T}} f_o(x) log P_o(x) + K \sum_{m=3}^{11} \sum_{n=15}^{21} f_s(m,n) log P_s(m,n) \tag{6}$$

where

$$f_{10,j}(x) = \frac{1}{K} \sum_{i=1}^{K} \sum_{m=3}^{11} \sum_{n=15}^{21} I_{i,49-m+j,x} P(Sp_{10}=m, Sp_{35}=n|S_i,\theta),$$

$$f_{35,j}(x) = \frac{1}{K} \sum_{i=1}^{K} \sum_{m=3}^{11} \sum_{n=15}^{21} I_{i,43-m-n+j,x} P(Sp_{10}=m, Sp_{35}=n|S_i,\theta),$$

$$f_o(x) = \frac{1}{K(65-12)} \sum_{i=1}^{K} \sum_{m=3}^{11} \sum_{n=15}^{21} O_{i,m,n}(x) P(Sp_{10}=m, Sp_{35}=n|S_i,\theta),$$

$$f_s(m,n) = \frac{1}{K} \sum_{i=1}^{K} \sum_{m=3}^{11} \sum_{n=15}^{21} P(Sp_{10}=m, Sp_{35}=n|S_i,\theta).$$

Let θ^0 denote the value of θ at the beginning of the first iteration. θ^0 was initialized randomly so that the E step can proceed. In each iteration, we use the current estimate θ^t to calculate the expected complete data log likelihood.

The M step maximizes (6) with respect to θ. The maximum likelihood estimates of θ when the complete data were known are just sample frequencies $f_{10,j}$, $f_{35,j}$, f_o, and f_s, $j = 1, \ldots, 6$. That is,

$$P_{10,j}^{t+1}(x) = f_{10,j}(x), x \in \mathcal{D},$$

$$P_{35,j}^{t+1}(x) = f_{35,j}(x), x \in \mathcal{D},$$

$$P_o^{t+1}(x) = f_o(x), x \in \mathcal{D},$$

$$P_s^{t+1}(m,n) = f_s(m,n). \tag{7}$$

The new value of θ can be used in the next iteration. The process iterates to convergence. Given the model parameters calculated from the positive training sequences (i.e., the promoter sequences in the training data set T), we can determine the locations of the two binding sites of any DNA sequence S_n, which could be a training sequence or a test sequence, a positive sequence or a negative sequence, by choosing the two spacer lengths sp_{10} and sp_{35} that maximize $P(S_n, Sp_{10}, Sp_{35}|\theta) = P(S_n|Sp_{10}, Sp_{35}, \theta) P_s(Sp_{10}, Sp_{35})$. We then align the two binding sites of the training promoter sequences and extract features from the different regions using the MDD technique and the motif based method, described below.

2.4 The MDD Technique

The MDD technique was first proposed to detect the splice site in human genomic DNA in the gene prediction software GENSCAN [4]. It was later adopted in the latest version of the gene prediction software MORGAN [25]. MDD was derived from the PWM model 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 438 promoter sequences available to estimate the parameters $P_i(x)$, $x \in \mathcal{D}$, in PWM where $P_i(x)$ represents the probability of nucleotide x occurring at position i in the sequences. Equivalently we have roughly $438/4 = 109$ 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 of x_i at position i given x_{i-1} at position $i-1$, which is the upstream neighbor of x_i at position i, x_{i-1}, $x_i \in \mathcal{D}$. 109 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 U of aligned sequences, it first chooses the consensus nucleotide C_i at each position i. In our case, the set U includes subsequences in the same region of all the positive training sequences (i.e., promoter sequences). Then the χ^2 statistic χ_{ij} is calculated to measure the dependencies between C_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 U into two disjoint subsets, U_m (containing all sequences that have the consensus nucleotide C_m at position m) and $U - U_m$ (containing all sequences that do not have the consensus nucleotide C_m at position m).

The MDD procedure is then applied recursively to U_m and $U - U_m$ respectively until any one of the following conditions holds: no further decomposition is possible, no significant dependencies between positions exist in the resulting subsets, 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.

Given a set of sequences, the MDD feature values of each sequence are calculated as follows. First, the MDD technique 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 sequences, these matrices are used to calculate the MDD feature values of each 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 sequence S is calculated by

$$P(X) = p_1(x_1)p_2(x_2)p_3(x_3) \ldots p_n(x_n), \tag{8}$$

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

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

Then, for example, for the subsequence TG, $P(\mathtt{TG})$=0.330 $\times$ 0.271=0.089.

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

$$P(X) = \begin{cases} p_m(C_m)cp_1^m(x_1) \ldots cp_{m-1}^m(x_{m-1})cp_{m+1}^m(x_{m+1}) \ldots cp_n^m(x_n) & \text{if } C_m = x_m \\ p_m(x_m)cp_1^{\overline{m}}(x_1) \ldots cp_{m-1}^{\overline{m}}(x_{m-1})cp_{m+1}^{\overline{m}}(x_{m+1}) \ldots cp_n^{\overline{m}}(x_n) & \text{otherwise} \end{cases}, \tag{9}$$

where $p_m(C_m)$ ($p_m(x_m)$, respectively) represents the probability of C_m (x_m, respectively) at 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 at position i given $x_m = C_m$ ($x_m \neq C_m$, respectively).

2.5 The Motif Based Method

To calculate the motif feature values of each sequence, we first apply our pattern matching tool, Sdiscover [29], to the positive training data (i.e., the E. Coli promoter sequences) to find weak motifs in the -54 region, the -44 region, the -35 region, the -22 region and the -10 region respectively, in these sequences.

Let V be a set of sequences. We define the occurrence number of a motif (subsequence) to be the number of sequences in V that contain the motif. The Sdiscover tool can find all the motifs M where M is within the allowed Mut mutations of at least $Occur$ sequences in the given set V and $|M| \geq Length$ where $|M|$ represents the number of nucleotides in M. Mut, $Occur$, $Length$ are user-specified parameters.

In our case, the set V includes all the subsequences in the same region (e.g., the -54 region) of the positive training sequences. The required length of the motifs is fixed for each region. 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 required is 2 and the allowed mutation number is 0. The occurrence number of a motif is assigned as the weight of the motif. Intuitively, the more frequently a motif occurs in a region of the positive training sequences, the higher weight it has.

Given a set of sequences, which could be training sequences or test sequences, positive sequences or negative 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 above 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 sequence S, the subsequence in that region of S is matched against the motifs in that region. If there are matched motifs, the feature value of that region for S 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 any 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.

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. Each training sequence or test sequence is encoded by the high level features as described in the previous section and the feature values are used as the input of the neural networks.

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 the above 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 the above 2 distance features. The number of hidden units is determined experimentally according to the evidence of the model.

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 $Classifier\_2$.

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:

$$P(M_i|D) = \frac{P(D|M_i)P(M_i)}{P(D)}, \tag{10}$$

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 (including both positive and negative training data), where N is the total number of the 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

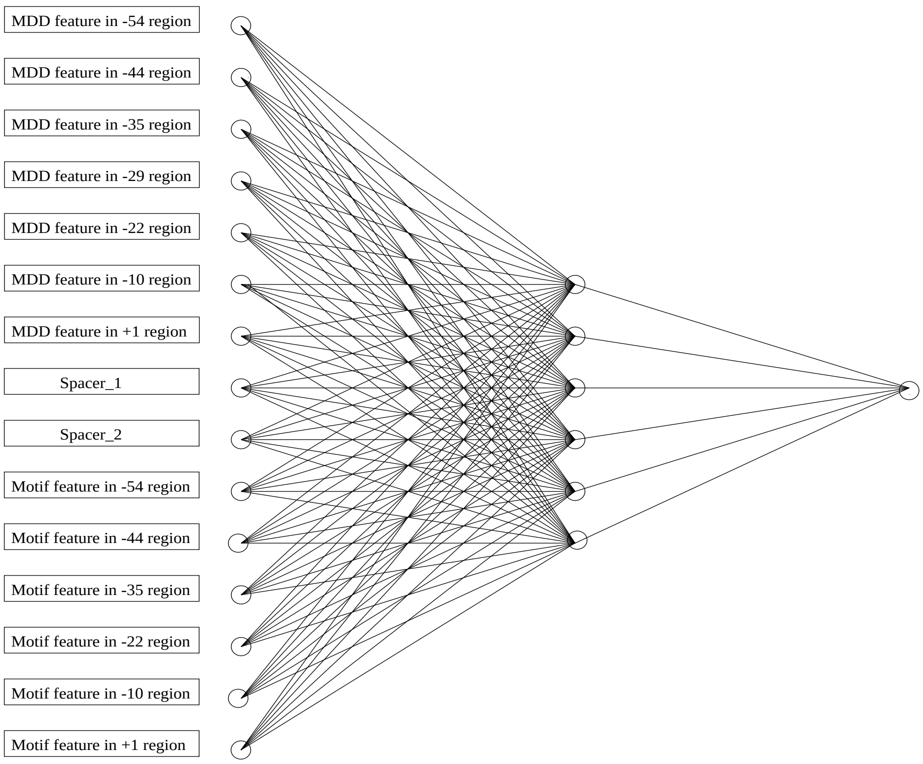

Figure 4: The Bayesian neural network architecture of $Classifier\_2$. Spacer_1 is the distance between the -35 box and the -10 box. Spacer_2 is the distance between the -10 region and the transcriptional start site.

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 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:

$$P(D|\mathbf{w},\mathbf{A}) = \Pi_{m=1}^{N} y^{t_m}(1-y)^{1-t_m} = exp(-G(D|\mathbf{w},\mathbf{A})), \tag{11}$$

where $G(D|\mathbf{w},\mathbf{A})$ is the cross-entropy error function

$$G(D|\mathbf{w},\mathbf{A}) = -(\sum_{m=1}^{N} t_m logy + (1-t_m)log(1-y)). \tag{12}$$

$G(D|\mathbf{w},\mathbf{A})$ is the objective function and is minimized in the non-Bayesian neural network training process, which is a maximum likelihood estimation based method, and assumes all possible weights are equally likely.

In the non-Bayesian neural network, 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 objective function is minimized, to penalize the neural network with weights of large magnitudes, penalizing the over-complex model and favoring the simple model. However, 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 is 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 variance $\frac{1}{\alpha}$. 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:

$$P(\mathbf{w}|\alpha, \mathbf{A}) = \frac{exp(-\sum_{c=1}^{n} \alpha_c E_W^c)}{Z_W}, \tag{13}$$

where $Z_W = \int exp(-\sum_{c=1}^{n} \alpha_c E_W^c) d\mathbf{w}$ is a Gaussian integral. From (12) and (13), we can get a posterior probability:

$$P(\mathbf{w}|D, \alpha, \mathbf{A}) = \frac{exp(-\sum_{c=1}^{n} \alpha_c E_W^c - G(D|\mathbf{w}, \mathbf{A}))}{Z_M}, \tag{14}$$

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

The Bayesian training of neural networks is an iterative procedure. In the implementation of the Bayesian neural network that we adopt,[3] each iteration involves two level inferences. Figure 5 illustrates the training process.

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 $\sum_{c=1}^{n} \alpha_c E_W^c + G(D|\mathbf{w}, \mathbf{A})$. For the first level inference, Bayes' rule is

$$P(\mathbf{w}|D, \alpha, \mathbf{A}) = \frac{P(D|\mathbf{w}, \mathbf{A}) P(\mathbf{w}|\alpha, \mathbf{A})}{P(D|\alpha, \mathbf{A})}, \tag{15}$$

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

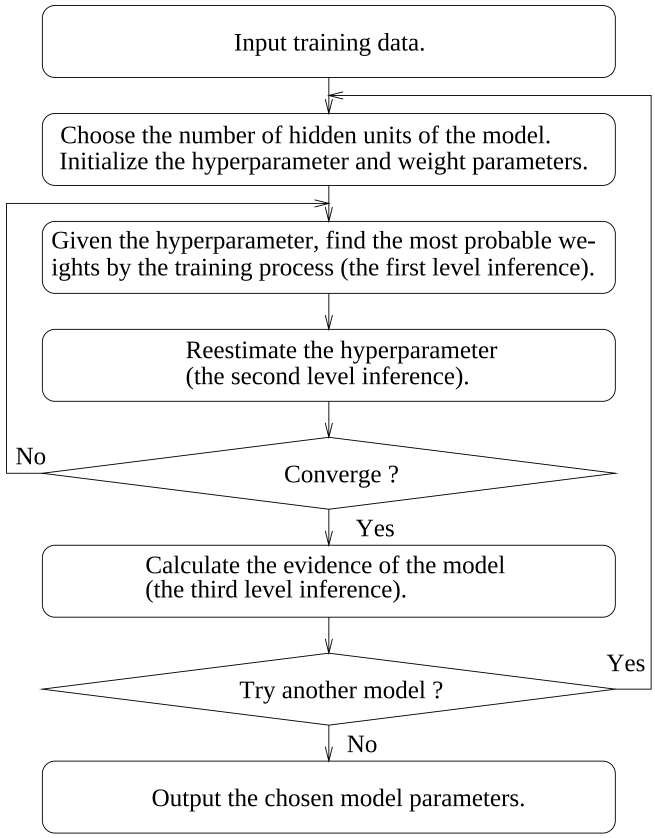

Figure 5: The training process of the Bayesian neural network.

where $P(D|\mathbf{w}, \mathbf{A})$, $P(\mathbf{w}|\alpha, \mathbf{A})$, and $P(\mathbf{w}|D, \alpha, \mathbf{A})$ are given by (11), (13) and (14) respectively. The $P(\mathbf{w}|D, \alpha, \mathbf{A})$ can be approximated by a Gaussian centered around $\mathbf{w}^{mp}$ [17]:

$$P(\mathbf{w}|D, \alpha, \mathbf{A}) \simeq P(\mathbf{w}^{mp}|D, \alpha, \mathbf{A}) exp(-\frac{1}{2}(\mathbf{w} - \mathbf{w}^{mp})^T \mathbf{H}(\mathbf{w} - \mathbf{w}^{mp})), \tag{16}$$

where $\mathbf{H} = - \bigtriangledown \bigtriangledown log P(\mathbf{w}|D, \alpha, \mathbf{A})|_{\mathbf{w}^{mp}}$ is the Hessian matrix evaluated at $\mathbf{w}^{mp}$.

At the second level, the hyperparameter α is optimized. For the second level inference, Bayes' rule is

$$P(\alpha|D, \mathbf{A}) = \frac{P(D|\alpha, \mathbf{A}) P(\alpha|\mathbf{A})}{P(D|\mathbf{A})}. \tag{17}$$

Because of the lack of the prior knowledge of $P(\alpha|\mathbf{A})$, we assume $P(\alpha|\mathbf{A})$ to be a constant and ignore it. Since the normalizing factor $P(D|\mathbf{A})$ is also a constant, the value of α maximizing the posterior $P(\alpha|D, \mathbf{A})$ can be inferred by maximizing the evidence of α, $P(D|\alpha, \mathbf{A})$, which is the normalizing factor in (15). So $P(D|\alpha, \mathbf{A}) = \int P(D|\mathbf{w}, \mathbf{A}) P(\mathbf{w}|\alpha, \mathbf{A}) d\mathbf{w}$. The evidence $P(D|\alpha, \mathbf{A})$ is maximized by differentiation with respect to α. We can find the new hyperparameter value α^{new} by setting the differentiation to zero,

$$\alpha_i^{new} = \frac{\gamma_i}{\sum_{j=1}^{q_i} (w_j^{mp})^2}, \tag{18}$$

where γ_i is the number of "well-determined parameters" in the group i [17]. The new hyperparameter value α^{new} is then used in the next iteration. The process iterates until the convergence is reached.

The third level inference is the model comparison. This level is carried out manually. For the third level inference, Bayes' rule is

$$P(\mathbf{A_i}|D) = \frac{P(D|\mathbf{A_i}) P(\mathbf{A_i})}{P(D)}, \tag{19}$$

where $P(\mathbf{A_i})$ is the prior probability and is assumed to be a constant; $P(D)$ is also a constant. The posterior probability $P(\mathbf{A_i}|D)$ can be determined by the evidence $P(D|\mathbf{A_i})$, which is the normalizing factor at the second level inference. So $P(D|\mathbf{A_i}) = \int P(D|\alpha, \mathbf{A_i}) P(\alpha|\mathbf{A_i}) d\alpha$. Different models with a different number of hidden units were tested. The model with the largest evidence value was chosen.

3.3 The Testing Phase

In the testing phase, for the three basic classifiers we have developed, the output of a Bayesian neural network, y, is given by the marginalization of the network weight distribution. That is,

$$P(t = 1|\mathbf{x}, D, \mathbf{A}) = \int P(t = 1|\mathbf{x}, \mathbf{w}, \mathbf{A}) P(\mathbf{w}|D, \mathbf{A}) d\mathbf{w} = \int y(\mathbf{x}; \mathbf{w}, \mathbf{A}) P(\mathbf{w}|D, \mathbf{A}) d\mathbf{w}. \tag{20}$$

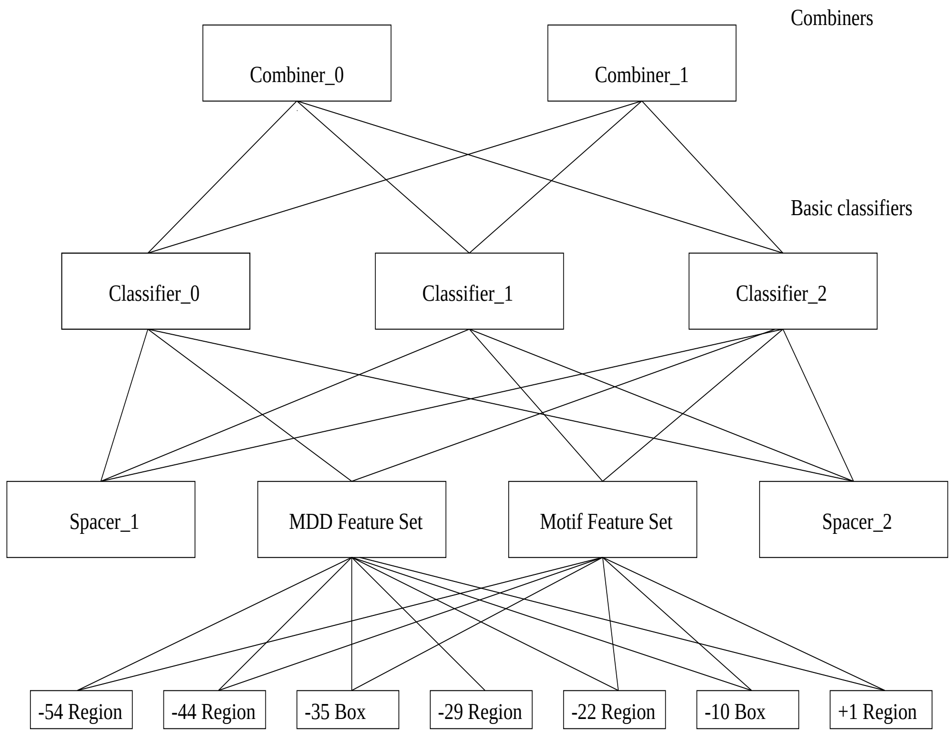

Figure 6: Our classification scheme.

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 the decision boundary 0.5, the test sequence is classified as a promoter; if it is less than the decision boundary 0.5, the test sequence is classified as a non-promoter; otherwise the test sequence gets the "no-opinion" verdict.

4 Combination of Basic Classifiers

The three basic classifiers described in the previous section can be combined into one classifier in the second level (see Figure 6). We explore two methods for combining the three basic classifiers: $Combiner\_0$ and $Combiner\_1$.

$Combiner\_0$ employs an unweighted voter. Let $output_i$, $0 \leq i \leq 2$, be the output value of $Classifier\_i$. If the three basic classifiers agree on the classification results (Promoter, Non-promoter, No-opinion), 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 "no-opinion".

$Combiner\_1$ 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

$$\frac{1}{w_0 + w_1 + w_2} \sum_{i=0}^{2} w_i \times output_i. \tag{21}$$

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 latest E. Coli promoter compilation [22]. There were 441 E. Coli promoters aligned by the transcriptional start site. We trimmed each promoter sequence to a sequence of 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 438 promoter sequences.

The negative data (i.e., non-promoter sequences) was retrieved from Genbank.[4] The negative data are E. Coli genes with the preceding promoter region deleted. Each negative sequence is also 65 nucleotides long. There were 1,314 negative sequences.

5.2 Results

Table 2 gives the ten-fold cross validation results for the three basic classifiers and Table 3 gives the results for the combined classifiers. In ten-fold cross validation, the dataset containing both the positive data (promoters) and 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 run, the neural network 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.

The machine used to conduct the experiments was a 350 MHz Pentium II PC running a Linux operating system. The time spent in extracting features by a basic classifier in a run was 0.69 seconds on average. The time spent in training the basic classifier in a run was 328.4 seconds on average. The time spent in the testing phase in a run was 0.35 seconds on average.

[4] Available from ftp://ncbi.nlm.nih.gov/repository/Eco/COLIBANK.

	$Classifier\_0$	$Classifier\_1$	$Classifier\_2$
Precision rate	92.8%	92.9%	93.0%
Specificity	96.5%	95.5%	94.9%
Sensitivity	81.5%	85.2%	87.4%

Table 2: Performance of the three basic classifiers.

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

$$\frac{C}{N} \times 100\%, \tag{22}$$

where C is the number of test sequences classified correctly and 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

$$(1 - \frac{N_{fp}}{N_{ng}}) \times 100\%, \tag{23}$$

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

$$\frac{N_{tp}}{N_{po}} \times 100\%, \tag{24}$$

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

	$Combiner\_0$	$Combiner\_1$
Precision rate	93.9%	95.0%
Specificity	96.4%	97.6%
Sensitivity	86.3%	87.4%

Table 3: Performance of the two combined classifiers.

From Table 3, we can see that $Combiner\_0$ and $Combiner\_1$ outperform the three basic classifiers. The $Combiner\_1$ gives the best precision rate 95%. 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 93% 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:

$$100\% - (\binom{3}{3}(1-93\%)^3 + \binom{3}{2}93\%(1-93\%)^2) = 98.5\%. \tag{25}$$

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.

Classification results	Percentage of the test sequences
$Combiner\_0$ and $Combiner\_1$ agreed & both were correct	92.9%
$Combiner\_0$ and $Combiner\_1$ agreed & both were wrong	3.9%
$Combiner\_0$ and $Combiner\_1$ disagreed & $Combiner\_0$ was correct	1.0%
$Combiner\_0$ and $Combiner\_1$ disagreed & $Combiner\_1$ was correct	2.2%
$Combiner\_0$ and $Combiner\_1$ disagreed & both were wrong	0.0%

Table 4: The complementarity of $Combiner\_0$ and $Combiner\_1$.

Table 4 illustrates the complementarity between $Combiner\_0$ 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 92.9%/(92.9%+3.9%) =95.9%. From Table 4, we can see that when $Combiner\_0$ and $Combiner\_1$ disagree, the probability that one is correct is 100%.

6 Conclusion

In this chapter 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 95% was achieved. Currently we are extending the approach to classify protein sequences and to recognize the full gene structure.

Acknowledgments

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. O. N.

Ozoline for providing the promoter sequences used in the chapter. The anonymous reviewers gave useful comments, which helped to improve both the presentation and the quality of the chapter.

References

[1] Bailey, T. L. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning*, 21:51–83, 1995.

[2] Brazma, A., Jonassen, I., Ukkonen, E., and Vilo, J. Discovering patterns and subfamilies in biosequences. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pages 34–43, 1996.

[3] 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.

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

[5] 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.

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

[7] Dempster, A., Laird, N., and Rubin, D. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39 (Series B), 1–38, 1977.

[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] 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.

[12] 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.

[13] 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.

[14] 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.

[15] Lawrence, C. E. and Reilly, A. A. An expectation-maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins: Structure, Function, and Genetics*, 7:41–51, 1990.

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

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

[18] Mackay, D. J. C. A practical Bayesian framework for backprop networks. *Neural Computation*, 4(3):448–472, 1992.

[19] 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.

[20] Neal, R. M. *Bayesian Learning for Neural Networks.* Lecture Notes in Statistics, volume 118, Springer-Verlag, New York, 1996.

[21] Opitz, D. W. and Shavlik, J. W. Connectionist theory refinement: Genetically searching the space of network topologies. *Journal of Artificial Intelligence Research*, 6:177–209, 1997.

[22] 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.

[23] 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.

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

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

[26] Salzberg, S. A method for identifying splice sites and translational start sites in eukaryotic mRNA. *Computer Applications in the Biosciences*, 13(4):365–376, 1997.

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

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

[29] 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.

[30] 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.

[31] Wang, J. T. L., Rozen, S., Shapiro, B. A., Shasha, D., Wang, Z., and Yin, M. New techniques for DNA sequence classification. *Journal of Computational Biology*, 6(2):209–218, 1999.

[32] 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.

[33] Wu, C. H., Berry, M., Fung, Y. S., and McLarty, J. Neural networks for full-scale protein sequence classification: Sequence encoding with singular value decomposition. *Machine Learning*, 21:177–193, 1995.

[34] Xu, Y., Mural, R. J., Einstein, J. R., Shah, M. B., and Uberbacher, E. C. Grail: A multi-agent neural network system for gene identification. *Proceedings of the IEEE*, 84(10):1544–1551, 1996.

[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.