BIOINFORMATICS

Vol. 1 no. 1 2002
Pages 1–8

Modeling Splicing Sites with Pairwise Correlations

Masanori Arita, Koji Tsuda and Kiyoshi Asai

Computational Biology Research Center (CBRC), National Institute of Advanced Industrial Science and Technology (AIST), 2-41-6 Aomi, Koto-ku, Tokyo, 135-0064, Japan

ABSTRACT
Motivation: A new method for finding subtle patterns in sequences is introduced. It approximates the multiple correlations among residuals with pair-wise correlations, with the learning cost $O(m^2n)$ where n is the number of training sequences, each of length m. The method suits to model splicing sites in human DNA, which are reported to have higher-order dependencies.
Results: By computational experiments, the prediction accuracy of our model was shown to surpass that of previously reported Markov models for the prediction of acceptor sites in human.
Availability: The C++ source code is available on request from the authors.
Contact: m-arita@aist.go.jp

INTRODUCTION

The prediction of splicing sites poses a challenge of machine learning from primary nucleotide sequences. The prediction not only contributes to gaining biological insight into splicing, but improves the accuracy of a gene finder, which is a gateway for virtually any type of sequence analysis.

Two types of splicing junctions are commonly called an acceptor (3' splice site) and a donor (5' splice site). Donor sites, with GT, correspond to the beginning of introns, while acceptor sites are their terminals with AG (GT-AG rule)[†]. The present knowledge on these sites in vertebrates is the rough consensus patterns: 5'–AG↓GTAAGT–3' for donors, and 5'–YYYYYYNCAG↓–3' for acceptors (Mount, 1982) [‡]. Inspired by this apparent consensus, many researchers attempted to predict splicing sites using weight matrices (Staden, 1986; Shapiro & Senapathy, 1987), neural networks (Brunak *et al.*, 1991), and Markov models (Zhang & Marr, 1993; Salzberg, 1997).

It is noteworthy that the linear-chain first-order Markov model shows a high prediction accuracy (Salzberg, 1997). C. Burge, who investigated the dependence structure of splicing sites, made the following observations. (For detail, readers are referred to his review (Burge, 1998).)

- In acceptor sites, residuals depend on pyrimidines at adjacent positions.
- In donor sites, almost three fourth of all residual pairs exhibit significant dependence.
- A training set of several hundreds is not enough to estimate the transition parameters of higher-order Markov models.

From these observations, it is concluded that a first-order Markov model is not the best method, but rather a clever compromise to model splicing sites. As will be shown, both donors and acceptors have complex dependencies among all residuals, therefore, models that can consider distant correlations should be better for learning these sites.

The dilemma we face is that, despite the significant dependencies among most residual pairs in splicing sites, we cannot introduce a complex model without a large, high-quality data set. This impediment forced researchers to estimate correlations among only limited residual pairs.

One solution, proposed by Burge, is the Maximal Dependence Decomposition (MDD) method, which is basically a decision tree bifurcating at the most influential residuals. Since branching occurs only when statistically significant residuals are detected, this method can suppress the increase of model parameters, compared to higher-order Markov models, although it considers higher-order dependencies. Indeed, the MDD model showed an improvement over previous models, and its accuracy had been unrivaled until the appearance of the Bayes network model (Cai *et al.*, 2000). The Bayes model is obtained by (1) computing the correlations between all residuals, (2) finding the maximum spanning tree by linking positions of high correlations, and (3) computing the conditional probability for each linked position. The number of parameters is the same as in the linear-chain Markov model.

In this paper, we propose the use of Bahadur expansion (Bahadur, 1961), a representation of the probability

[†] We ignore the few exceptional splicing sites that do not obey this rule.
[‡] ↓, Y, and N denote the cleavage cite, a pyrimidine base T or C, and any of four bases, respectively.

distribution in multiple orders of correlations. When truncated at the second order, it becomes the estimate of the original probability distribution requiring no more computation than first-order Markov models. Still, it is a more complex model than the Markov models, because it parameterizes all pair-wise correlations.

In computational experiments, its prediction accuracy was shown to surpass that of first-order Markov models, including the Bayes network variant, for the prediction of acceptor sites showing more complex correlations than donor sites in our data set. The learning cost of our method is $O(m^2 n)$, where n is the number of examples, each of length m. Our model is trained much faster than support vector machines (SVMs), which require a quadratic programming problem of $n \times n$ coefficient matrix.

SYSTEMS AND METHODS

Maximum Entropy Modeling

Let $\boldsymbol{x} = (x_1, \cdots, x_m),\ x_i \in \{0, 1, 2, \cdots, |S-1|\}$ denote a sequence of length m over the alphabet S. In the case of a DNA sequence, the cardinality of S is 4, where its residuals A,C,G,T are coded into integers 0,1,2,3, respectively. The purpose of probabilistic modeling is to obtain the underlying probability distribution of sequences $p^*(\boldsymbol{x})$ from a finite number of examples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ (Durbin *et al.*, 1998). A simplistic model is to construct the empirical distribution defined as

$$\hat{p}(\boldsymbol{x}) \overset{def}{=} \frac{1}{N}\sum_{k=1}^{N} I(\boldsymbol{x} = \boldsymbol{x}_k), \tag{1}$$

where I denotes the indicator function whose value is 1 if the equation holds and is 0 otherwise. Except when the number of samples is close to infinity, however, $\hat{p}$ gives a very poor estimate of p^*. Typically, $\hat{p}$ gives a very sparse distribution whose value is 0 for most sequences. Therefore, $\hat{p}$ is usually smoothed to obtain a better estimate. In terms of information theory, this smoothing amounts to maximizing the entropy of distribution.[§] To obtain a good estimate, it is important to determine how far it is maximized. Pushing it to its limit, for example, it ends up with a truly randomized distribution with maximum entropy. For this purpose, one must add constraints on $\hat{p}$ to prevent the entropy from growing too large. Typically, these constraints are derived from training samples. This probabilistic modeling technique is called "Maximum Entropy Modeling" (MEM) (Cover & Thomas, 1991).

The moment constraints (i.e. mean and correlation) are often used in MEM. Let μ_{si} and $\hat{C}_{si,tj}$ denote the mean and correlation of N samples:

$$\mu_{si} \overset{def}{=} \frac{1}{N}\sum_{k=1}^{N} I(x_{ki} = s),$$
$$\hat{C}_{si,tj} \overset{def}{=} \frac{1}{N}\sum_{k=1}^{N} I(x_{ki} = s) I(x_{kj} = t),$$

where s and t are alphabet letters and i and j denote their sequence positions. When the moments are constrained to coincide with those of training samples, MEM is formulated as follows:

Find p *maximizing* $-\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p(\boldsymbol{x})$
such that

$$\sum_{\boldsymbol{x}} I(x_i = s) p(\boldsymbol{x}) = \mu_{si},\ \textit{and}$$
$$\sum_{\boldsymbol{x}} I(x_i = s) I(x_j = t) p(\boldsymbol{x}) = \hat{C}_{si,tj},$$

where $\sum_{\boldsymbol{x}}$ *denotes the sum over every possible* $\boldsymbol{x}$.

It is known that the optimal p belongs to the parametric family (Cover & Thomas, 1991)

$$p(\boldsymbol{x}) = \exp\Big(-\varphi + \sum_{i=1}^{m}\sum_{s=0}^{S-1} \theta_{is} I(x_i = s) + \sum_{i=1}^{m}\sum_{j=i+1}^{m}\sum_{s=0}^{S-1}\sum_{t=0}^{S-1} w_{si,tj} I(x_i = s) I(x_j = t)\Big) \tag{2}$$

where φ is a normalization constant determined so that $\sum_{\boldsymbol{x}} p(\boldsymbol{x}) = 1$. When parameters θ and w are determined so that the mean and correlation satisfy the constraints, $p(\boldsymbol{x})$ gives the maximum entropy distribution.

This parametric model of distribution is known as Boltzmann machines in the neural network community (Hertz *et al.*, 1991). However, the exact determination of θ (in other words, "training of the Boltzmann machine") takes exponential time with respect to the sequence length m (Hertz *et al.*, 1991). Therefore, approximation is necessary to obtain a suboptimal but acceptable solution.

Bahadur Expansion

For approximated training of Boltzmann machines, we adopt the Bahadur expansion of probability distribution (Bahadur, 1961; Humphreys & Titterington, 1999; Losee, 1994). The aim of Bahadur expansion is to expand the logarithm of the empirical distribution $\hat{p}(\boldsymbol{x})$ into multiple orders of correlations as in the Fourier transformation. When this expansion is truncated at the second

[§] The entropy of distribution p is defined as $-\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p(\boldsymbol{x})$.

order, a suboptimally trained Boltzmann machine is obtained that gives good results in practice. In this section, the expansion is briefly introduced. For theoretical details, readers are referred to other literature (Humphreys & Titterington, 1999).

For simplicity, let us assume that a sequence including A,C,G,T is readily coded into a binary sequence. By the sparse-coding scheme, A,C,G,T are coded as 1000, 0100, 0010, and 0001, respectively. Let $x_i \in \{0,1\}$ $(i = 1, \cdots, m)$ denote this sequence of binary random variables. Let $\mu_i = p(x_i = 1)$, or the expected value of $x_i = 1$. Let us define

$$p_{[1]}(\boldsymbol{x}) \overset{def}{=} \prod_{i=1}^{m} (\mu_i)^{x_i}(1-\mu_i)^{(1-x_i)}.$$

Consider a vector space of real-valued functions where the inner product is defined as

$$\langle f, g\rangle \overset{def}{=} \sum_{\boldsymbol{x}\in X} f(\boldsymbol{x})g(\boldsymbol{x})p_{[1]}(\boldsymbol{x}) \tag{3}$$

Then, the set of functions

$$\begin{aligned}\mathcal{G} = \{1; z_1, z_2, \ldots, z_m; z_1 z_2, \ldots, z_{m-1}z_m;\\ \cdots; z_1 z_2 \cdots z_m\}\end{aligned}$$

forms orthonormal bases of this vector space, or Bahadur space (Bahadur, 1961), where

$$z_i = (x_i - \mu_i)/\sqrt{\mu_i(1-\mu_i)}.$$

This tells that *any* function f can be uniquely represented as

$$f(\boldsymbol{x}) = \sum_{g\in\mathcal{G}} \langle f, g\rangle\, g(\boldsymbol{x}). \tag{4}$$

Let us choose $f(\boldsymbol{x})$ to be $\log(p(\boldsymbol{x})/p_{[1]}(\boldsymbol{x}))$, or its equivalent

$$\log p(\boldsymbol{x}) \overset{def}{=} \log p_{[1]}(\boldsymbol{x}) + f(\boldsymbol{x}). \tag{5}$$

Let $\mathcal{I}$ denote the set of all subsets $\{i_1, \ldots, i_l\}$ $(0 < l \leq m)$ of indices $\{1, \ldots, m\}$. Note that the null set ϕ is included in $\mathcal{I}$. For $I \in \mathcal{I}$, define

$$z_I(\boldsymbol{x}) \overset{def}{=} z_{i_1}\cdots z_{i_l},$$

where $z_\phi(\boldsymbol{x}) = 1$. Note that each z_I denotes a function in $\mathcal{G}$. Let r_I be the coefficient for z_I when f is expanded using (4).

$$r_I \overset{def}{=} \langle f, z_I\rangle = \sum_{x\in X} f(\boldsymbol{x})z_I(\boldsymbol{x})p_{[1]}(\boldsymbol{x}).$$

Then, from (5) we have

$$\begin{aligned}\log p(\boldsymbol{x}) &= \log p_{[1]}(\boldsymbol{x}) + \sum_{I\in\mathcal{I}} r_I z_I(\boldsymbol{x})\\ &\simeq \log p_{[1]}(\boldsymbol{x}) + r_\phi + \sum_{i=1}^{n} r_i z_{\{i\}}(\boldsymbol{x})\\ &\quad + \sum_{i<j} r_{ij} z_{\{ij\}}(\boldsymbol{x}).\end{aligned} \tag{6}$$

Each coefficient in (6) represents the followings:

$$\begin{aligned}r_\phi &= \sum_{\boldsymbol{x}} p_{[1]}(\boldsymbol{x})f(x),\\ r_i &= \sum_{\boldsymbol{x}} z_i p_{[1]}(\boldsymbol{x})f(x),\\ r_{ij} &= \sum_{\boldsymbol{x}} z_i z_j p_{[1]}(\boldsymbol{x})f(x),\end{aligned}$$

where $f(x)$ is defined in (5). Note that the coefficient r_i is the contribution of position i to the distribution, and r_{ij} captures distant pairwise correlations between position i and j.

These coefficients include the sum over every possible $\boldsymbol{x}$, but assuming that $\hat{p}(\boldsymbol{x}) > 0$, r_ϕ is efficiently computed as

$$r_\phi = \frac{1}{N}\sum_{k=1}^{N} \frac{p_{[1]}(\boldsymbol{x}_k)}{\hat{p}(\boldsymbol{x}_k)}\{\log \hat{p}(\boldsymbol{x}_k) - \log p_{[1]}(\boldsymbol{x}_k)\} \tag{7}$$

The other coefficients can be computed similarly.

Let us define $\log p_{[2]}(\boldsymbol{x})$ as the approximated expression of (6). In general, $p_{[2]}$ is not a probability distribution, that is, $\sum_{\boldsymbol{x}} p_{[2]}(\boldsymbol{x}) \neq 1$. When normalized, however, $p_{[2]}$ belongs to the parametric family of Boltzmann machines in (2).

In summary, $p_{[2]}$, the truncated Bahadur expansion, is an approximation of Boltzmann machine learning, which would otherwise take exponential time for its exact learning. Since it does not give the exact maximum entropy solution, the remaining question is its accuracy, and we will demonstrate its result in the following experiments.

Computational Costs

For learning with the Bahadur expansion, three coefficients r_ϕ, r_i, r_{ij} in (6) are computed. The computational cost is $O(m^2 n)$, where m and n are the sequence length and the number of examples, respectively. Note that the cost for computing a log-likelihood $\log p(\boldsymbol{x})$ for each training example is $O(m^2)$. The costs for other methods are as follows: The linear-chain Markov model takes

Table 1. Three data sets of splicing sites

	true donors	false donors	true acceptors	false acceptors
Human (Genie96) (Reese *et al.*, 1997)	1324	4922	1324	5553
Human (Asai *et al.*, 1998)	2314	4628	2377	4754
C.elegans (Kent & Zahler, 2000)	74505	177061	74455	122154

Table 2. Learning results from 1324 positive and 4922 negative donors (Reese *et al.*, 1997). The scores are averaged over 100 trials. The window size is 15 bases (including GT), beginning from the -5 position of the exon-intron boundary.

	Bahadur		Bayes network		chain Markov	
Training FN (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)
0	0.0	31.89	0.0	33.76	0.0	29.4
1	0.8332	16.77	0.7491	19.8	0.7491	15.26
2.5	2.104	11.6	2.247	13.05	2.247	8.905
5	4.603	8.7	4.495	9.22	4.495	5.739
10	9.602	5.293	9.738	5.222	9.738	3.395
20	19.6	2.598	19.48	2.552	19.48	2.02
25	24.58	1.97	24.53	2.104	24.53	1.58
30	29.51	1.61	29.24	1.578	29.24	1.278

$O(mn)$ for learning and $O(m)$ for testing (i.e. computing a log-likelihood). Also, the Bayes network model takes $O(m^2n)$ for learning and $O(m)$ for testing.

Thus, the learning cost of our method is the same as that for the Bayes network model, except for the quadratic cost for testing. This quadratic cost is inevitable as long as all pair-wise correlations are considered, meaning that our computational cost is optimal. In an ordinary situation, this quadratic term can be tolerated, since m is relatively small (e.g. $m = 15$ in our experiment). When the number of examples n is very large, $O(n)$ term dominates the computation. Note that other complicated learning methods may impose a much larger cost. For example, in SVM (Zien *et al.*, 2000), a quadratic programming problem of $n \times n$ coefficient matrix must be solved for learning.

IMPLEMENTATION

Data set

The performance of the three models for donor and acceptor sites was tested on three data sets: (Reese *et al.*, 1997), (Asai *et al.*, 1998), and (Kent & Zahler, 2000), obtained from the www (Table1).

For each set, data were randomly partitioned into training data (90%) and test data (10%). Then, the log-likelihood ratio $\log P_{pos}(\boldsymbol{x}) - \log P_{neg}(\boldsymbol{x})$ was computed in the three different models: the truncated Bahadur expansion, the Bayes network, and the linear-chain Markov model.

Results

Prediction accuracies of these methods are shown in tables from 2 to 7, formatted in the same style as that used by Cai *et al* (Cai *et al.*, 2000). Tables show the percentage of false-positives and false-negatives in the test sequences. Test data were classified by their log-likelihood computed in each model, according to the threshold determined by the ratio of false-negatives in the training data.

In two data sets (Tables 2∼5) the Bahadur model showed better separation between positive and negative acceptors than did the Bayes network and the linear-chain Markov models. The results of the two models were similar. For donors, however, the Bahadur model provided no advantage.

To further investigate this result, the absolute parameter values $|w_{i,j}|$ were plotted for the data set by Asai *et al.* (Figure 1). Here, $w_{i,j}$ was derived from r_{ij} in (6) as

$$w_{i,j} = r_{ij}\sqrt{\mu_i(1-\mu_i)}\sqrt{\mu_j(1-\mu_j)}.$$

The value $|w_{i,j}|$ illustrates the contribution of $I(x_i = 1)I(x_j = 1)$ in the obtained probability distribution.

Table 3. Learning results from 1324 positive and 5553 negative acceptors (Reese *et al.*, 1997). The scores are averaged over 100 trials. The window size is 15 bases (including AG), beginning from the -10 position of the intron-exon boundary.

	Bahadur		Bayes network		chain Markov	
Training FN (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)
0	0	36.74	0.0	47.99	0.0	48.19
1	0.7491	24.15	0.7491	35.63	0.7491	35.32
2.5	2.247	16.57	2.247	27.79	2.247	26.48
5	4.495	12.41	4.495	21.54	4.495	21.15
10	9.738	8.055	9.738	14.84	9.738	14.47
20	19.48	4.266	19.48	8.123	19.48	8.02
25	24.53	3.253	24.52	6.551	24.52	6.267
30	29.24	2.57	29.24	5.447	29.24	5.131

Table 4. Learning results from 2314 positive and 4754 negative donors (Asai *et al.*, 1998). The scores are averaged over 100 trials. The window size is 15 bases (including GT), beginning from the -5 position of the exon-intron boundary.

	Bahadur		Bayes network		chain Markov	
Training FN (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)
0	0.0	58.88	0.0	44.12	0.0	43.37
1	0.8332	39.91	0.8562	18.94	0.8562	17.69
2.5	2.104	28.39	2.141	11.35	2.141	9.66
5	4.603	20.09	4.709	7.154	4.709	5.35
10	9.602	12.51	9.846	3.643	9.846	3.353
20	19.6	6.463	19.69	1.62	19.69	1.616
25	24.58	5.016	24.56	1.198	24.56	1.126
30	29.51	4.064	29.45	0.8346	29.45	0.7982

In the figures, dark dots show high contribution to the probability distribution. They indicate that distant pairs have high contribution both in donor and accepter sites, and donor sites have a stronger correlation in their diagonal direction. This tendency contributed to the better performance of the linear-chain Markov model for donors. For acceptors, on the other hand, the Bahadur method scored better because it arrested their distant correlations.

For the large data set (Tables 6∼7), the Bahadur model provided no advantage.

DISCUSSION AND CONCLUSION

Splicing Site Prediction

In our experiment on Human data, the false-positive rates of acceptors were worse than those of donors. Although this relative difference agreed with already reported result (Cai *et al.*, 2000), the error rates were higher than the report, where most false-positive rates for the same experiment were less than 1%.

Since the error rates were reduced for the large data set on C. elegans, the high error rates are probably due to our data selection. One clear reason is that, for both splicing sites, only 15 base-pairs around the GT-AG signal were input to the learning models to compare their performance. This size may be too short especially for acceptors. It is reported in (Reese *et al.*, 1997) that acceptors were better learned from 80 base-pairs using a neural network. Such a long region is not applicable for the Bahadur model, however, for its number of parameters to be learned becomes $\binom{80}{2}$. Determination of an appropriate window size for learning is important for a practical application of our model.

Expansion Strategy

In this paper, the Bahadur expansion was not fully exercised because it can expand *any* real-valued function f as in (5). The function f was defined so that its truncated

Table 5. Learning results from 2139 positive and 4278 negative acceptors (Asai *et al.*, 1998). The scores are averaged over 100 trials. The window size is 15 bases (including AG), beginning from the -10 position of the intron-exon boundary.

	Bahadur		Bayes network		chain Markov	
Training FN (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)
0	0	55.71	0.0	60.94	0	61.62
1	0.8332	36.6	0.8332	39.83	0.8332	40.91
2.5	2.104	24.7	2.104	29.13	2.104	29.4
5	4.603	17.07	4.603	20.19	4.603	20.48
10	9.602	9.999	9.602	12.73	9.602	13.3
20	19.6	5.474	19.6	7.105	19.6	7.682
25	24.58	4.26	24.58	5.538	24.58	6.105
30	29.51	3.266	29.51	4.58	29.51	4.93

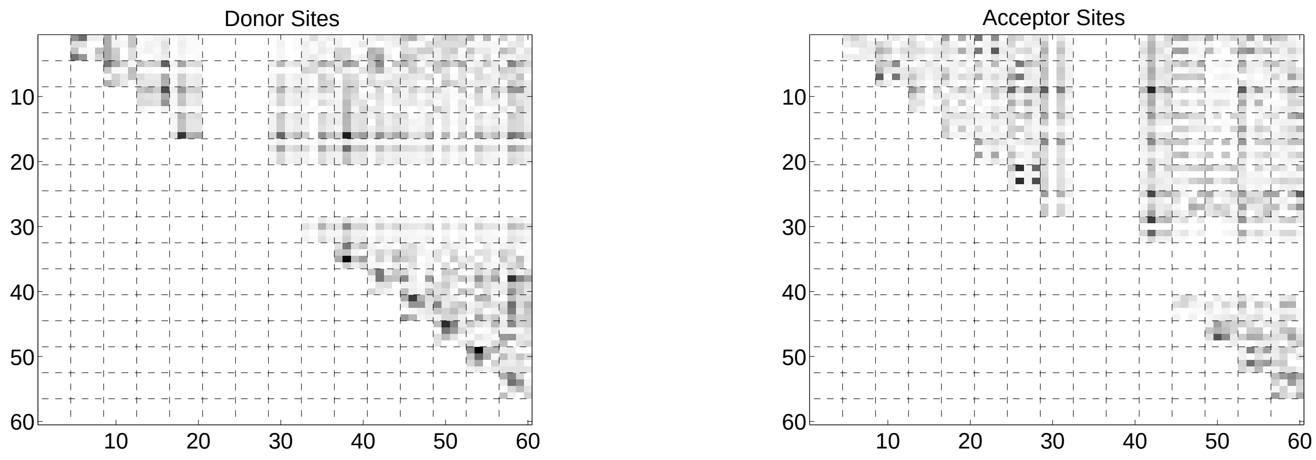

Fig. 1. The absolute parameter value $|w_{i=s,j=t}|$ which illustrates the contribution of $I(x_i = s)I(x_j = t)$ $(s,t \in \{A,C,G,T\})$. The X, Y-axis correspond to the indices t, j and s, i as $x = 4(j-1)+t+1$ and $y = 4(i-1)+s+1$, respectively. The dotted line shows the boundary between residuals, and the blank part (range $20 \sim 30$ in donors and $30 \sim 40$ in acceptors) corresponds to the fixed residuals GT or AG.

form corresponds to the learning of Boltzmann machines, but other choices are possible (Bahadur, 1961).

For classification problems, the following definition is another possibility:

$$f(\boldsymbol{x}) \overset{def}{=} \begin{cases} 1 & \text{(positive sample } \boldsymbol{x}\text{)} \\ -1 & \text{(negative sample } \boldsymbol{x}\text{).} \end{cases}$$

This definition results in a simpler expansion:

$$f(\boldsymbol{x}) \simeq p_{[1]}(\boldsymbol{x})(\sum_{\boldsymbol{x}\in\mathcal{P}} z_i z_j - \sum_{\boldsymbol{x}\in\mathcal{N}} z_i z_j),$$

where $\mathcal{P}$ and $\mathcal{N}$ denote positive and negative samples, respectively. Its prediction rates were sometimes better than the original expansion 6 (data not shown), but for this formula, we could not assign any theoretical meaning, such as the Boltzmann machines.

Theoretical Drawbacks

First, there is no performance guarantee on the approximation error of the truncated Bahadur expansion. Our experiments showed that its result is better than first-order Markov models in case of complex correlations, but this empirical conclusion on its performance needs theoretical confirmation. This is an important open problem.

The second theoretical problem concerns the validity of computing the log-likelihood ratio of the positive and negative models using the truncated Bahadur expansion, because the truncated result is not a probability distribution.

Table 6. Learning results from 74505 positive and 177061 negative donors (Kent & Zahler, 2000). The scores are averaged over 10 trials. The window size is 15 bases (including GT), beginning from the -5 position of the exon-intron boundary.

	Bahadur		Bayes network		chain Markov	
Training FN (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)
0	0	86.49	0.0	93.87	0	94.05
1	0.7649	44.84	0.9834	49.5	0.9834	48.25
2.5	1.912	29.88	2.472	31.24	2.472	30.32
5	4.207	18.56	4.944	17.89	4.944	17.22
10	8.796	9.485	9.9	8.192	9.9	7.967
20	17.98	3.76	19.8	2.808	19.8	2.801
25	22.56	2.52	24.75	1.93	24.75	1.904
30	27.15	1.8	29.7	1.397	29.7	1.402

Table 7. Learning results from 74455 positive and 122154 negative acceptors (Kent & Zahler, 2000). The scores are averaged over 10 trials. The window size is 15 bases (including AG), beginning from the -10 position of the intron-exon boundary.

	Bahadur		Bayes network		chain Markov	
Training FN (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)	True test FN (%)	False test FP (%)
0	0	82.1	0.0	85.66	0	87.73
1	0.7649	17.41	0.9841	17.32	0.9841	16.73
2.5	1.912	10.38	2.473	10.21	2.473	10.25
5	4.207	6.585	4.947	6.307	4.947	6.554
10	8.796	3.776	9.894	3.68	9.894	3.844
20	17.98	1.928	19.8	1.927	19.8	2.005
25	22.56	1.557	24.75	1.505	24.75	1.564
30	27.15	1.279	29.7	1.226	29.7	1.274

The solution to these problems are prerequisites for engineering a good function for expansion, as well as for evaluating the classified results.

Conclusion

We introduced a new approximation method, using Bahadur expansion, for representing multiple correlations on sequences. The expansion can be applied to any real-valued function; this paper showed one example whose truncation corresponds to the approximated learning of the Boltzmann machine. Although its theoretical background is incomplete, our method is sound; it models probability distribution among all residuals. In the computational experiments on splicing sites, the Bahadur model predicted acceptors better than did Markov models because acceptors exhibited multiple correlations among residuals in our data set.

In general, if the training data are known to have higher order correlations, there is little reason to use Markov models, for they presume correlations only between adjacent positions. If enough data are provided, therefore, our model is more suitable than Markov models for biological sequences with complex correlations. Our strategy is similar to that of Agarwal and Bafna (Agarwal & Bafna, 1998) in this respect, but the implementation is as easy as for Markov models.

Since our model does not directly suggest a corresponding biological mechanism, locating major (or influential) correlations out of learned results is the important next step toward the elucidation of the hidden mechanism(s) behind splicing. Application to subtle signals other than splicing sites also adds another prospect to the Bahadur method.

Acknowledgments We thank O. Gotoh, T. Tanaka, and S. Akaho for valuable discussions.

REFERENCES

Agarwal, P. & Bafna, V. (1998). Detecting non-adjoining correlations within signals in DNA. In *Proceedings of the 2nd RECOMB Conference*. ACM Press, pp. 2–8.

Asai, K., Itou, K., Ueno, Y. & Yada, T. (1998). Recognition of human genes by stochastic parsing. In *Pacific Symposium on Biocomputing (PSB98)*. World Scientific, Hawaii, pp. 228–239. http://www.genedecoder.org/.

Bahadur, R. R. (1961). A representation of the joint distribution of responses to n dichotomous items. In Solomon, H., (ed.) *Studies in Item Analysis and Prediction*. Stanford University Press, pp. 158–168.

Brunak, S., Engelbrecht, J. & Knudsen, S. (1991). Prediction of human mRNA donor and acceptor sites from the DNA sequences. *Journal of Molecular Biology*, **220**, 49–65.

Burge, C. B. (1998). Modeling dependencies in pre-mRNA splicing signals. In Salzberg, S. L., Searls, D. B. & Kasif, S., (eds.) *Computational Methods in Molecular Biology*. Elsevier, pp. 129–164.

Cai, D., Delcher, A., Kao, B. & Kasif, S. (2000). Modeling splice sites with Bayes networks. *Bioinformatics*, **16**, 152–158.

Cover, T. & Thomas, J. (1991). Elements of Information Theory. John Wiley & Sons.

Durbin, R., Eddy, S., Krogh, A. & Mitchison, G. (1998). Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press.

Hertz, J., Krogh, A. & Parmer, R. (1991). Introduction to the Theory of Neural Computation. Addison-Wesley.

Humphreys, K. & Titterington, D. (1999). The exploration of new methods for learning in binary Boltzmann machines. In Heckerman, D. & Whittaker, J., (eds.) *Artificial Intelligence and Statistics 99*. Morgan Kaufmann, San Francisco, pp. 209–214.

Kent, W. J. & Zahler, A. M. (2000). The intronerator: exploring introns and alternative splicing in c. elegans. *Nucleic Acids Research*, **28**, 91–93. http://www.cse.ucsc.edu/ kent/intronerator/.

Losee, R. (1994). Term dependence: Truncating the Bahadur Lazarsfeld expansion. *Information Processing & Management*, **30**, 293–303.

Mount, S. (1982). A catalogue of splice junction sequences. *Nucleic Acids Research*, **10**, 459–472.

Reese, M. G., Eeckman, F. H., Kulp, D. & Haussler, D. (1997). Improved splice site detection in genie. *Journal of Computational Biology*, **4**, 311–323. http://www.fruitfly.org/seq_tools/datasets/Human/.

Salzberg, S. (1997). A method for identifying splice sites and translational start sites in eukaryotic mRNA. *Comput. Appl. Biol. Sci.*, **13**, 365–376.

Shapiro, M. & Senapathy, P. (1987). RNA splice junctions of different classes of eucaryotes: Sequence statistics and functional implications in gene expression. *Nucl. Acids. Res.*, **15**, 7155–7174.

Staden, R. (1986). The current status and portability of our sequence handling software. *Nucleic Acids Research*, **14**, 217–231.

Zhang, M. & Marr, T. (1993). A weight array method for splicing signal analysis. *Computer Applications in the Biosciences*, **9**, 499–509.

Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T. & Müller, K.-R. (2000). Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites. *BioInformatics*, **16**, 799–807.