BIOINFORMATICS

Vol. 19 no. 15 2003, pages 1964–1971
DOI: 10.1093/bioinformatics/btg265

Sequence alignment kernel for recognition of promoter regions

Leo Gordon[1,*], Alexey Ya. Chervonenkis[1,2], Alex J. Gammerman[1], Ilham A. Shahmuradov[1] and Victor V. Solovyev[3]

[1]Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK, [2]Institute of Control Science, Profsoyuznaja 65, Moscow, Russia and [3]Softberry Inc., 116 Radio Circle, Suite 400, Mount Kisco, NY, 10549, USA

Received on December 6, 2002; revised on March 2, 2003; accepted on April 29, 2003

ABSTRACT

In this paper we propose a new method for recognition of prokaryotic promoter regions with startpoints of transcription. The method is based on Sequence Alignment Kernel, a function reflecting the quantitative measure of match between two sequences. This kernel function is further used in Dual SVM, which performs the recognition.

Several recognition methods have been trained and tested on *positive data set*, consisting of 669 σ^{70}-promoter regions with known transcription startpoints of *Escherichia coli* and two *negative data sets* of 709 examples each, taken from *coding* and *non-coding* regions of the same genome. The results show that our method performs well and achieves 16.5% average error rate on *positive* & *coding negative* data and 18.6% average error rate on *positive* & *non-coding negative* data.

Availability: The demo version of our method is accessible from our website http://mendel.cs.rhul.ac.uk/

Contact: leo@cs.rhul.ac.uk

INTRODUCTION

Promoter region is an area on the chromosome which determines where the transcription of a particular gene(s) should be initiated and on what conditions. In prokaryotic organisms the *promoter region* occupies several hundred base pairs upstream of *Transcription Start Site* (*TSS*) and a smaller area downstream of TSS, and may serve for transcription of a single gene as well as for a group of genes (an operon).

During the last five years many prokaryotic genomes have been sequenced, including that of *Escherichia coli* (Blattner *et al*., 1997). The gene content of these genomes was mostly computationally recognized (which is usually in good agreement with later lab experiments). However, the promoter regions and TSS are still undetermined in most cases and the software able to accurately predict promoters in sequenced genomes is not yet available in public domain. *Promoter recognition*, the computational task of finding the promoter regions on a DNA sequence, is very important for defining the transcription units responsible for specific pathways (because gene prediction alone cannot provide the solution) and for analysis of gene/operon regulation.

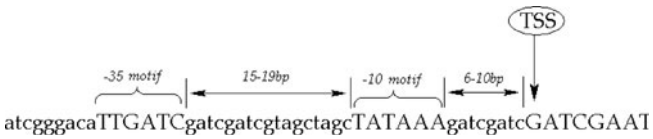

Fig. 1. Prokaryotic σ^{70} promoter region with TSS, '−10' and '−35' binding motifs and two spacers.

Experimental studies of prokaryotic promoter regions show that they contain a certain set of *binding motifs*—relatively short chunks of DNA to which the RNA polymerase and special regulatory proteins can bind in order to initiate and control the transcription (De Haseth *et al*., 1998). The majority of prokaryotic promoter recognition approaches use the *binding motif* based model of *promoter region*, which emphasizes the importance of *binding motifs* and essentially neglects the data between them (*spacers*). Figure 1 shows such a model for prokaryotic σ^{70} promoter region with two *binding motifs* and two *spacers*.

Because certain variation in the lengths of the *spacers* and informational content of each *binding motif* is allowed,[4] the search for the 'right' placement of *binding motifs* is not completely trivial.

Different general approximate matching techniques, as well as the ones, specifically designed for this task, are used for *binding motif* search (Crochemore and Sagot, 2002, and references therein). Various approaches based on weight matrices (Staden, 1984; Harley and Reynolds, 1987; Mulligan and McClure, 1986), neural networks (Lukashin *et al*., 1989; Demeler and Zhou, 1991; O'Neill, 1991, 1992; Horton and Kanehisa, 1992; Mahadevan and Ghosh,

*To whom correspondence should be addressed.

[4]For example, so-called -10-box of σ^{70} bacterial promoter may look like TATAAT, CATAAT, TATAAA, etc.

1994; Pedersen and Engelbrecht, 1995), generalized portrait (Alexandrov and Mironov, 1990), hidden markov models (Pedersen *et al*., 1996), genetic algorithms (Bailey and Hart, http://citeseer.nj.nec.com/172804.html), syntactic recognition algorithms (Rosenblueth *et al*., 1996; Leung *et al*., 2001) and automatic motif discovery techniques (Hertz and Stormo, 1996; Bailey and Elkan, 1994, 1995; Tompa, 1999; Liu *et al*., http://bioprospector.stanford.edu/; Kent, http://www.cse.ucsc.edu/~kent/improbizer/improbizer.html) have been proposed to serve this purpose. There are also separate approaches that use the motifs found as their input to some 'second level' recognizers, such as hidden Markov models, neural networks or support vector machines, for further prediction (Eddy, 1998, http://hmmer.wustl.edu/; Grundy *et al*., 1997; Pavlidis *et al*., 2001).

When such approaches are specifically applied to prokaryotic promoter recognition, the results reported are quite high. However, most of the tests were performed on relatively small data sets, available at that time. To increase the amount of positive examples a risky procedure of 'permuting the bases on non-critical positions' was proposed and used by O'Neill (1992). As Horton and Kanehisa (1992) have noted when comparing different methods, the prediction accuracy drops as the amount of input data grows.

Unfortunately, negative examples are an even more serious issue, as there is no available data on which regions of DNA are *proven not to contain* promoters. So the experiments were done either on randomly generated sequences (usually matching the relative distribution of bases to that of real promoters) or on sequences randomly chosen from other genomes. Obviously, such choice of data could give additional hints to the recognizers.

Today we have much more available data on *E.coli*—669 σ^{70} promoters[5] from two partially overlapping databases (Salgado *et al*., 2000, http://www.cifn.unam.mx/Computational_Genomics/regulondb/; Hershberg *et al*., 2001, http://bioinfo.md.huji.ac.il/marg/promec/. Our preliminary analysis of the data has shown much higher variation of all the elements of the *binding motif* model than it was in the data sets used by other authors, so the expected prediction accuracy of applying these methods to the full data set is smaller.

We propose an alternative approach—not to break up the promoter regions into 'important' and 'unimportant' parts, but to compare them as whole entities. In this paper we make use of recently developed Sequence Alignment Kernel (Watkins, 2000; Surkov *et al*., 2001) to define the measure of similarity between two promoter regions. This measure is further used in SVM algorithm (Vapnik, 1998), which performs the training and recognition.

Our method is preferable in cases when we have a sufficient number of known promoter regions, but might not know anything about their composition.

One of the findings of this approach is that in spite of generality the developed kernel has outperformed in accuracy of the prediction several other known approaches, which suggests that the information 'between the boxes' might also be important for recognition.

PROBLEM STATEMENT

We will treat genome as a string S composed of *letters* {A,C,G,T}. On it some specific positions called TSS are given. We assume that, given a position p, a region $S_{p-U} \cdots S_{p+D}$ contains enough information to distinguish whether p is a TSS or not. We will call such a region a (potential) *promoter region*.

We are given a *training set* composed of *positive examples* ('true' *promoter regions*) and *negative examples* ('false' *promoter regions*[6]). Our goal is, given an arbitrary *potential promoter region* to be able to find out whether it is 'true' or 'false' *promoter region*.

ALGORITHM: SEQUENCE ALIGNMENT KERNEL

Our method is based on building the *kernel function* $K(R, Q)$ as a quantitative measure of similarity between two sequences R and Q. Such a function should be suitable for classification by Dual SVM (Vapnik, 1998) or any other kernel-based classification method.

Suppose we are given a matrix $\mathrm{Swap}(x, y)$ which defines the score corresponding to a single point mutation of letter x into letter y or vice versa (the matrix is symmetric). We are also given a vector $\mathrm{Gap}(x)$ which defines the score corresponding to a single point deletion or insertion of letter x.

One of the schemes for simultaneous generation of two sequences over a given alphabet was proposed by Watkins (2000). The generative model may emit either two letters (one into each sequence), only one letter into the first sequence (which corresponds to a gap into the second one), or only one letter into the second sequence (which corresponds to a gap into the first one). The model is completely defined by the probabilities for each pair it may emit. For any two non-empty sequences there are several ways (or paths) to generate them using this model. For every such path the corresponding probability is the product of probabilities along the path. The total probability $P(x, y)$ that the sequences x and y will be generated by the model is the sum of probabilities of all the paths that lead to generating the given pair. Watkins (2000)

[5]There are still not enough examples from other classes of *E.coli* promoters like σ^{38}, σ^{54}, etc.—for statistical analysis.

[6]Theoretically, they do not have to belong to the genome and can be any strings of $U + D$ *letters* from {A, C, G, T}.

has proven that the function $P(x, y)$ is symmetric and positively definite, and so may be used as a kernel for SVM and other kernel-based algorithms.

If we take the Swap(x, y) matrix and Gap(x) vector to be the logarithms of the probabilities from Watkins' model, then the classical Global Alignment algorithm by Needleman–Wunsch (Needleman and Wunsch, 1970) can be regarded as a method to calculate the probability of the *most probable path* to generate the two sequences. However, it has not been proven that the alignment score it provides is non-negatively definite [one of the necessary conditions for a kernel function to be valid, see Vapnik (1998)].

Straightforward summation of all paths' probabilities would need exponential time. We would also need to add together a big number of very small values, which might suffer from floating point arithmetic underflow. Following the dynamic programming ideas used in Global Alignment, an algorithm was proposed (Surkov *et al.*, 2001) for fast and precise calculation of the kernel function $P(x, y)$.

The algorithm

Suppose we are given two sequences to align, $Q = \text{'}ACCT\text{'}$ and $R = \text{'}ACGTC\text{'}$. Let us write them along the two dimensions of an empty matrix (Fig. 2).

In each cell $p_{i,j}$ of the matrix we will be keeping the probability that $Q_{1\cdots j}$ aligns with $R_{1\cdots i}$. It is convenient to start the calculations from the bottom left corner, which is initialized with the value of 1. Then, we fill all the other cells using the recursive formula:

$$
\begin{aligned}
&p_{0,0} = 1,\\
&p_{i,0} = p_{0,j} = 0, \quad \text{for } i > 0 \text{ and } j > 0,\\
&p_{i,j} \leftarrow \text{Swap}(R_i, Q_j) \cdot p_{i-1,j-1}\\
&\qquad + \text{Gap}(Q_j) \cdot p_{i,j-1}\\
&\qquad + \text{Gap}(R_i) \cdot p_{i-1,j},
\end{aligned}
$$

where the Swap(x, y) matrix and the Gap(x) vector of probabilities are given as parameters to the algorithm. The kernel value we are looking for is the probability $\mathcal{K} = p_{|R|,|Q|}$ in the top right corner of the matrix.

Note, that to calculate values on any 'backslash' diagonals of the p matrix ($i + j = D$) we only need to know the values on the two preceding diagonals: $i + j = D - 1$ and $i + j = D - 2$. This property is used to speed up the calculations. If we 'turn' the matrix p by 45°, the diagonals become rows and columns, yielding O[max$(|R|, |Q|)^2$] time complexity and O[max$(|R|, |Q|)$] space complexity.

Allowing for affine gaps

A more complicated version of this algorithm accounts for 'affine gaps' (Gotoh, 1982)—it means that in a run of gaps, the one starting the run may be given a different probability than the gaps extending the run. This is attained by using a more sophisticated computational scheme, a three-layer matrix (Fig. 3, for a close-up see Fig. 4) instead of one layer described above.

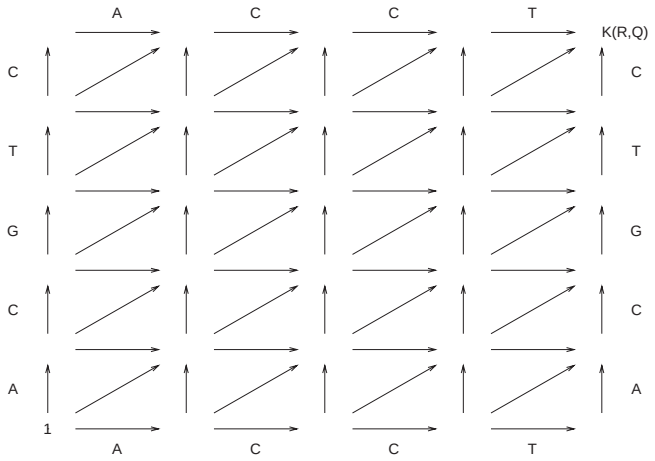

Fig. 2. 'Flat' version of sequence alignment.

On the *d-level* only diagonal transitions are allowed, which correspond to substitutions. *H-level* is used for 'horizontal' transitions only, it accounts for gaps in Q. *V-level* is used for 'vertical' transitions only, it accounts for gaps in R. The end result is the probability $\mathcal{K} = p^d_{|R|,|Q|}$. The core idea of this stratification is to multiply each transition from the main, *d-layer* to any of the two 'gap layers' by an additional probability coefficient, StartGap. If StartGap $= 1$, there is no difference with the original scheme. But if StartGap < 1, it is equivalent to paying an additional penalty in order to start the gap.

$$
\begin{aligned}
&p^{\mathrm{d}}_{0,0} = 1,\\
&p^{\mathrm{d}}_{0,j} = p^{\mathrm{d}}_{i,0} = 0, \quad \text{for } i > 0 \text{ and } j > 0,\\
&p^{\mathrm{h}}_{0,j} = p^{\mathrm{h}}_{i,0} = p^{\mathrm{v}}_{0,j} = p^{\mathrm{v}}_{i,0} = 0, \quad \text{for } i \geqslant 0 \text{ and } j \geqslant 0,\\
&p^{\mathrm{h}}_{i,j} \leftarrow \text{Gap}(Q_j) \cdot (p^{\mathrm{h}}_{i,j-1} + p^{\mathrm{d}}_{i,j-1} \cdot \text{StartGap}),\\
&p^{\mathrm{v}}_{i,j} \leftarrow \text{Gap}(R_i) \cdot (p^{\mathrm{v}}_{i-1,j} + p^{\mathrm{d}}_{i-1,j} \cdot \text{StartGap}),\\
&p^{\mathrm{d}}_{i,j} \leftarrow \text{Swap}(R_i, Q_j) \cdot (p^{\mathrm{h}}_{i-1,j-1} + p^{\mathrm{v}}_{i-1,j-1} + p^{\mathrm{d}}_{i-1,j-1}).
\end{aligned}
$$

The stratification has a slow-down impact on the performance,[7] but gives more flexibility to the system.

In addition to what is described above, we propose the following:

1. Take the root of power $|R| + |Q|$ from each kernel value—this gives us a close approximation to the similarity measure 'per symbol' (or rather 'per gap'), and makes the kernel values relatively independent of sequence lengths $|R|$ and $|Q|$.

[7] In fact, both the execution time and the memory needs increase in a constant number of times—this does not affect the complexity of the algorithm.

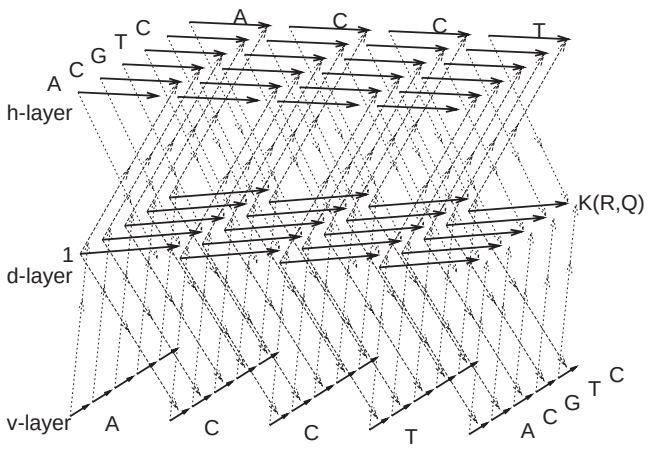

Fig. 3. '3D' version of sequence alignment, which accounts for affine gaps.

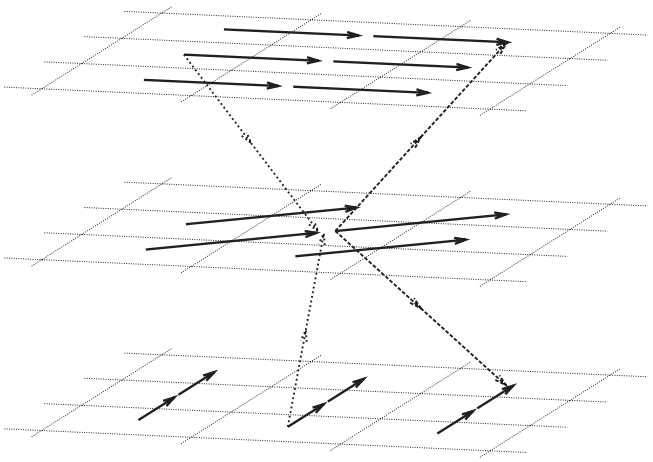

Fig. 4. A local fragment of the '3D' version of sequence alignment.

2. After the whole matrix of kernel values K has been found, normalize the matrix:

$$K'(X,Y) = \frac{K(X,Y)}{\sqrt{K(X,X)\cdot K(Y,Y)}}.$$

   Because a kernel corresponds to dot product in some imaginary feature space between two vectors, by this normalization we get rid of the 'lengths' of the vectors, so that only the cosine of the angle between them remains.

3. Then take each element to some fixed power $\alpha > 1$; this gives us one more convenient parameter to tune.[8] The need for this arises when the vectors are 'too much different' from each other, so that after the normalization we get ones on the diagonal, $K'(X,X) = 1$, and nearly zeros everywhere else.

[8]We are unaware of a theoretical proof that this operation is valid for kernel functions in general, but in our experiments involving generating hundreds of kernel matrices every single one was positively definite.

Table 1. Optimal Swap(x, y) matrix and Gap(x) vector used for σ^{70} prokaryotic promoter recognition

Swap(x, y)	A	C	G	T
A	1.0000	0.1738	0.3679	0.1738
C	0.1738	1.0000	0.1738	0.3679
G	0.3679	0.1738	1.0000	0.1738
T	0.1738	0.3679	0.1738	1.0000
Gap(x)	0.1054	0.1054	0.1054	0.1054

To test this method on the problem of prokaryotic promoter recognition, the following parameter values were found to be optimal: StartGap $= 0.05, \alpha = 1.3$. We obtained Swap(x, y) matrix and Gap(x) vector by exponentiation *Sankoff-76 transition/transversion* score matrix (Sankoff *et al*., 1976), which happened to work well applied to our problem (see Table 1).

COMPARISON OF DIFFERENT METHODS

Sequence Alignment Kernel was used in conjunction with Dual Support Vector Machine (Vapnik, 1998). This method was tested among several different promoter region predicting methods.

The data

As the primary source of data we used the sequenced genome of *E.coli* (strain K-12, substrain MG1655) Blattner *et al*. (1997). A list of 669 experimentally confirmed σ^{70} promoters with known TSS positions was put together from RegulonDB [Salgado *et al*. (2000); http://www.cifn.unam.mx/Computational_Genomics/regulondb/ and PromEC (Hershberg *et al*., 2001); http://bioinfo.md.huji.ac.il/marg/promec/] databases. Then *promoter regions* $[TSS-60\cdots TSS+19]$ were taken as the *positive examples*.

As there is no experimentally confirmed negative data (i.e. the positions that are confirmed *not* to be TSS), we had to take the risk and choose the *negative examples* randomly from the same chromosome. Approximately 81% of known TSS are located in the intergenic non-coding regions and 19% in the coding regions. So two different *negative example* sets were prepared:

(a) *coding negative example set* containing 709 subsequences, 80 letters each, from the coding regions (genes) and

(b) *non-coding negative example set* containing 709 subsequences, 80 letters each, from the non-coding regions (intergenic spacers).

The hypothetical non-TSS in both sets of examples is located in the 61st position, so the examples have the same format as the positive ones: $[nonTSS-60\cdots nonTSS+19]$.

The same data points were used when testing all the other methods.

Other methods

Here we briefly[9] list the methods that we used to compare with our Sequence Alignment Kernel based method. In all methods except the first two we used SVM in the last stage, simple or kernel-based.

- BLAST-based Nearest Neighbours method (Altschul *et al*., 1997) is simply a Nearest Neighbours classifier where the distance is defined by pairwise BLASTn E-value. On each iteration the training set is converted into BLAST-compatible database, then each test example is looked up in the database to find the nearest match.
- Boxes + threshold method (Staden, 1984; Harley and Reynolds, 1987) is probably the best known technique for automatic motif discovery. Here a hypothesis is made that σ^{70} promoter regions contain but two important *binding motifs* on relatively fixed positions (see Fig. 1). Every potential promoter area is converted into four numerical features (matching the scores for the '−10' and '−35' *binding motifs* and likelihoods of the two distances, from TSS to '−10' motif and '−10' motif to '−35' motif). Then the four are added together, forming the 'general' likelihood. The optimal threshold value for it is found on the training set, and then used to classify the test set examples.
- Boxes + SVM method is very similar to the previous one, but the four likelihoods are not added. They are used as four independent features in the standard SVM routine. Simply speaking, SVM finds the best combination of those features, of which the sum is but one, so it is easily explainable, why SVM-based methods generally perform better.
- Boxes and regulatory sites + SVM method (Bailey and Elkan, 1995) is based on the previous method, where three additional features are generated to each example from CRP, IHF and LexA *regulatory sites*.
  The resulting 4 + 3 features were used in the standard SVM routine.
- Zone likelihood + SVM method (Oppon and Hide, 1998) uses the hypothesis that *promoter region* has a distribution of *oligonucleotides* (short substrings of a given length) which is different from the overall distribution in the DNA. Unlike Oppon and Hide (1998), we suspected that different *zones* inside the *promoter region* have different distributions. We have found five zones, whose local distributions well predicted the promoter regions.[10]
  The score was computed for every such zone and the resulting five features were used in the standard SVM routine.
- Locality-improved kernel + SVM method (Schölkopf *et al*., 1998) was originally applied to *Translation Initiation Sites* prediction (Zien *et al*., 2000, locality-improved kernel), but as it suits relatively general purpose of classifying DNA regions, we applied it to prediction of *promoter regions*.[11]
  This method directly yielded the kernel function which was used in the kernel-based SVM routine.

Criteria and results of comparison

In every experiment the *positive examples* and the *negative examples* were mixed together, 1/2 of them were randomly chosen to serve as *training* and the other 1/2—as *test examples*. Then the recognition program was executed. After each execution four numbers were calculated:

- TP, true positives, #{correctly recognized positives},
- TN, true negatives, #{correctly recognized negatives},
- FN, false negatives, #{positives recognized as negatives},
- FP, false positives, #{negatives recognized as positives}.

For every method, 100 executions were performed. Then the average TP, TN, FN and FP were found and the following relative measures were calculated:

- FN% = FN/(TP + FN) × 100%,
- FP% = FP/(TN + FP) × 100%,
- AE% (average error) = (FN + FP)/(TP + TN + FP + FN) × 100%.
- Sn (sensitivity or recall) = TP/(TP + FN),
- Sp (specificity or precision) = TP/(TP + FP),
- CC (correlation coefficient)=

$$\frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP}) \times (\mathrm{TN}+\mathrm{FN}) \times (\mathrm{TP}+\mathrm{FN}) \times (\mathrm{TN}+\mathrm{FP})}}.$$

The Table 2 shows that Sequence Alignment Kernel based method outperformed other tested methods on average error and false negatives. This result is important, since it is clear that Sequence Alignment Kernel does not use any prior information as to what chunks of the *promoter region* are important and what are not. Obviously, some information 'between the boxes' is also important and needs more attention.[12]

[9]For full description see (Gordon, 2002).

[10]Relative to the TSS they are: (i) $[-71 \cdots -61]$ in mononucleotides, (ii) $[-60 \cdots -41]$ in mononucleotides, (iii) $[-40 \cdots -35]$ in pentanucleotides, (iv) $[-17 \cdots -8]$ in pentanucleotides, (v) exact position of TSS in mononucleotides.

[11]The parameters that were found to give the best prediction are: $l = 2, d_1 = 4, d_2 = 3$.

[12]It is interesting to note that, although boxes & regulatory sites method is quite close to Sequence Alignment Kernel when tested on the *positive examples* and *coding negative example set*, on the set composed of *positive examples* and *non-coding negative example set* it seriously falls behind. We think it happens because the regulatory sites are well spread in the non-coding regions, so they affect both positive and negative examples alike.

Table 2. Different methods compared, results averaged over 100 executions

Method	AE%	FN%	FP%	Sn	Sp	CC
Sequence Alignment	**16.5**	**18.5**	14.6	**0.82**	0.84	**0.67**
Kernel + SVM	*18.6*	*19.0*	18.2	*0.81*	0.81	*0.63*
Boxes + SVM	19.1	23.6	14.8	0.76	0.83	0.62
	20.5	25.6	15.7	0.74	0.82	0.59
Boxes + threshold	19.5	24.4	14.8	0.76	0.83	0.61
	21.0	28.4	14.0	0.72	0.83	0.58
Zone likelihood + SVM	21.0	32.2	**10.4**	0.68	0.86	0.59
	22.5	33.1	12.5	0.67	0.84	0.56
Locality-improved	19.3	24.9	14.1	0.75	0.83	0.62
kernel + SVM	23.5	38.8	*9.1*	0.61	*0.86*	0.55
Boxes & regulatory	16.8	22.7	11.3	0.77	**0.87**	**0.67**
sites + SVM	30.3	25.7	34.6	0.74	0.67	0.40
Blast-based Nearest	34.6	40.9	28.7	0.59	0.66	0.31
Neighbours	35.4	40.9	30.2	0.59	0.64	0.29

Upper rows: negative data is taken from coding regions (bold shows best values). lower rows: negative data is taken from non-coding regions (italics show best values).

DISCUSSION

We developed a new SVM-based approach using Sequence Alignment kernel, which achieves better prediction accuracy than that of other SVM-based methods. But it is also important to compare our methods with other non-SVM methods reported in the literature.

O'Neill and Chiafari (1989) used consensus-based approach on 47 known *E.coli* promoters, dividing them into three classes with 16, 17 and 18 bases separating −35 and −10 regions. Overall 77% were correctly identified, but the level of false positives was very high. Later, this work was continued (O'Neill, 1991, 1992) and the following prediction accuracies were achieved: 78–100% for 16 bp spacing, 97% for 17 bp spacing and 79% for 18 bp spacing.

Another paper (Demeler and Zhou, 1991) describes 'neural network optimization for *E.coli* promoter prediction methods'. A neural network was trained on a set of 80 known *E.coli* promoter sequences and a different number of random sequences. The prediction accuracy of the resulting weight matrix was tested against a separate set of 30 known promoter sequences and 1500 random sequences with equal composition of A, C, G and T bases. Accuracies of 100% on promoters and 98.4% on the random sequences were achieved with optimal parameters. However, these figures could have been very much affected by the choice of data. First, both training and test test were very small. Second, because of the way the negative examples were generated, the promoter search protocol is likely to be highly sensitive to the average A/T ratio of the input due to A/T relative richness of promoters (Mulligan and McClure, 1986; O'Neill and Chiafari, 1989).

Horton and Kanehisa (1992) reported 'perceptron type neural network for prediction of *E.coli* σ^{70} promoters'. Moreover, they reconstructed five previously reported methods and compared the quality of prediction of these methods and their own approach on the same data sets. Although prediction accuracy in previous reports (Demeler and Zhou, 1991; O'Neill, 1991, 1992) was very high, training and testing of perceptron type neural network in the same data gave comparable results. 'The difference in prediction rates with these different data sets seems to be explainable to a large extent by the differences in information content of the combined training and test sets' (Horton and Kanehisa, 1992): both of previously used data sets were essentially subsets of the one used by Horton and Kanehisa. In particular, O'Neill's data only contained promoters with 17 bp spacing.

Later, Mahadevan and Ghosh (1994) reported on using neural networks trained on 106 promoters and random sequences which were 60% A/T rich and tested on 126 promoters for recognition of *E.coli* promoters of all spacing classes (15–21 bp). This network showed 98% accuracy on promoters and 90.2% accuracy on non-promoters (tested on 500 randomly generated sequences).

At last, Leung *et al*. (2001) presented 'basic gene grammars and DNA-ChartParser for language processing of *E.coli* promoter DNA sequences approach'. The method was tested on 300 *E.coli* promoters and 300 non-promoter random sequences. Four experiments, performed using different 'grammar rules', have yielded the best prediction of 76% accuracy with 82% specificity and 69% sensitivity.

It should be noted that the set of known *E.coli* σ^{70} promoters that we used both for training and test is the largest and includes the previously used ones. Horton and Kanehisa (1992) have noted the drop in prediction accuracy with larger training and/or test data sets. Taking into consideration the tendency observed, our sequence alignment kernel gives quite comparable results. Moreover, in some of the experiments mentioned above random sequences were used as negative examples. We believe it is not quite fair, as they may differ too much from the actual genomic sequences. The negative examples we are using are not only from the same genome—they are from non-coding regions, to avoid the distributional 'hints' of the coding areas.

The future research would include reconstruction of other published methods, then training and testing them on our expanded data set. There is also a direction we would like to undertake to assess the 'trustworthiness' of our predictions, by using *confidence* and *credibility* measures for each individual prediction (Gammerman and Vovk, 2002).

ACKNOWLEDGEMENTS

The authors are grateful to Heladia Salgado, member of RegulonDB (Salgado *et al*., 2000 http://www.cifn.unam.mx/Computational_Genomics/regulondb/) database support group and Jose Carlos Gonzalez of Universidad Politecnica de Madrid (UPM) for useful discussions and

Ruti Hershberg, member of PromEC (Hershberg *et al*., 2001, http://bioinfo.md.huji.ac.il/marg/promec/) database support group for the data for the experiments and other useful information. We also wish to thank Spanish Ministerio de Educacion, Culltura y Deporte and School of Telecommunications, UPM, for their support through grant no. SAB2001-0057. We also wish to thank the anonymous referees for their valuable comments that helped us to improve the paper.

This work is supported by BBSRC grant no.111/BIO14428, 'Pattern recognition techniques for gene identification in plant genomic sequences' and EPSRC grant GR/M14937, 'Predictive complexity: recursion-theoretic variants'.

REFERENCES

Alexandrov,N. and Mironov,A. (1990) Application of a new method of pattern recognition in DNA sequence analysis: a study of *E.coli* promoters. *Nucleic Acids Res.*, **18**, 1847–1852.

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

Bailey,T. and Elkan,C. (1994) Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In *The Second International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, pp. 28–36.

Bailey,T.L. and Elkan,C. (1995) Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Mach. Learning*, **21**, 51–80.

Bailey,T. and Hart,W. Learning consensus patterns in unaligned DNA sequences using a genetic algorithm. http://citeseer.nf.nec.com/172804.html

Blattner,F., Plunkett,G., Bloch,C., Perna,N., Burland,V., Riley,M., Collado-Vides,J., Glasner,J., Rode,C., Mayhew,G. *et al*. (1997) The complete genome sequence of *Escherichia coli* k-12. *Science*, **277**, 1453–1462. Genbank/EMBL/DDBJ record: U00096.

Crochemore,M. and Sagot,M.-F. (2002) Motifs in sequences: localization and extraction. In Konopka,A. *et al*. (eds.), *Handbook of Computational Chemistry*, Marcel Dekker, New York (in press).

De Haseth,P., Zupancic,M. and Record,M. (1998) RNA polymerase–promoter interactions: the comings and goings of RNA polymerase. *J. Bacteriol.*, **180**, 3019–3025.

Demeler,B. and Zhou,G. (1991) Neural network optimization for *E.coli* promoter prediction. *Nucleic Acids Res.*, **19**, 1593–1599.

Eddy,S. (1998) Profile hidden markov models. *Bioinformatics*, **14**, 755–763.

Gammerman,A. and Vovk,V. (2002) Prediction algorithms and confidence measures based on algorithmic randomness theory. *Theoreth. Comput. Sci.*, **287**, 209–217.

Gordon,L. (2002) Using kernel methods in prokaryotic promoter recognition. Technical Report CLRC-TR-02-10, Dept. of Computer Science, Royal Holloway, University of London.

Gotoh,O. (1982) An improved algorithm for matching biological sequences. *J. Mol. Biol.*, **162**, 705–708.

Grundy,W. N., Bailey,T.L., Elkan,C.P. and Baker,M.E. (1997) Meta-meme: motif-based hidden markov models of biological sequences. *Compu. Appl. Biosci.*, **13**, 397–406.

Harley,C. and Reynolds,R. (1987) Analysis of *E.coli* promoter sequences. *Nucleic Acids Res.*, **15**, 2343–2361.

Hershberg,R., Bejerano,G., Santos-Zavaleta,A. and Margalit,H. (2001) Promec: an updated database of *Escherichia coli* mRNA promoters with experimentally identified transcriptional start sites. *Nucleic Acids Res.*, **29**, 277.

Hertz,G. and Stormo,G. (1996) *Escherichia coli* promoter sequences: analysis and prediction. *Methods Enzymol.*, **273**, 30–42.

Horton,P. and Kanehisa,M. (1992) An assesment of neural network and statistical approaches for prediction of *E.coli* promoter sites. *Nucleic Acids Res.*, **20**, 4331–4338.

Kent,J. Improbizer motif discovery program with web interface.

Leung,S.-W., Mellish,C. and Robertson,D. (2001) Basic gene grammars and dna-chartparser for language processing of *Escherichia coli* promoter dna sequences. *Bioinformatics*, **17**, 226–236.

Liu,X., Brutlag,D. and Liu,J. (2001) Bioprospector: discovering conserved DNA motifs in upstream regulatory regions of coexpressed genes.

Lukashin,A., Anshelevich,V., Amirikyan,B., Gragerov,A. and Frank-Kamenitskii,M. (1989) Neural network models for promoter recognition. *J. Biomol. Struct. Dyn.*, **6**, 1123–1133.

Mahadevan,I. and Ghosh,I. (1994) Analysis of *E.coli* promoter structures using neural networks. *Nucleic Acids Res.*, **22**, 2158–2165.

Mulligan,M. and McClure,W. (1986) Analysis of the occurence of promoter-sites in DNA. *Nucleic Acids Res.*, **14**, 109–126.

Needleman,S. and Wunsch,C. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 433–453.

O'Neill,M. (1991) Training back-propagation neural networks to define and detect DNA-binding sites. *Nucleic Acids Res.*, **19**, 313–318.

O'Neill,M. (1992) *Escherichia coli* promoters: neural networks develop distinct descriptions in learning to search for promoters of different spacing classes. *Nucleic Acids Res.*, **20**, 3471–3477.

O'Neill,M. and Chiafari,F. (1989) *Escherichia coli* promoters. II. A spacing-class dependent promoter search protocol. *J. Biol. Chem.*, **264**, 5531–5534.

Oppon,J. and Hide,W. (1998) A statistical model for prokaryotic promoter prediction. In *The Ninth Workshop on Genome Informatics*, pp. 271–273. Poster.

Pavlidis,P., Furey,T., Liberto,M., Haussler,D. and Grundy,W. (2001) Promoter region-based classification of genes. In *The Pacific Symposium on Biocomputing*.

Pedersen,A., Baldi,P., Brunak,S. and Chauvin,Y. (1996) Characterization of prokaryotic and eukaryotic promoters using hidden markov models. In *Proceedings of the 1996 Conference on Intelligent Systems for Molecular Biology*, pp. 182–191.

Pedersen,A. and Engelbrecht,J. (1995) Investigations of *Escherichia coli* promoter sequences with artificial neural networks: new signals discovered upstream of the transcriptional startpoint. In *Proceedings, Third International Conference on Intelligent Systems for Molecular Biology*, pp. 292–299.

Rosenblueth,D., Thieffry,D., Huerta,A., Salgado,A. and Collado-Vides. (1996) Syntactic recognition of regulatory regions in *Escherichia coli*. *Comput. Appl. Biol.*, **12**, 415–422.

Salgado,H., Santos-Zavaleta,A., Gama-Castro,S., Millan-Zarate,D., Blattner,F. and Collado-Vides,J. (2000) Regulondb (version 3.0): transcriptional regulation and operon organization in *Escherichia coli* K-12. *Nucleic Acids Res.*, **28**, 65–67.

Sankoff,D., Cedergren,R. and Lapalme,G. (1976) Frequency of insertion/deletion, transversion and transition in the evolution of 5S ribosomal RNA. *J. Mol. Evolution*, **7**, 133–149.

Schölkopf,B., Simard,P., Smola,A. and Vapnik,V. (1998) Prior knowledge in support vector kernels. In Jordan,M. Kearns,M. and Solla,S. (eds), *Advances in Neural Information Processing Systems 10*, MIT Press, Cambridge, MA, pp. 640–646.

Staden,R. (1984) Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Res.*, **12**, 505–519.

Surkov,D., Chervonenkis,A. and Gammerman,A. (2001) Kernel for protein sequences classification. Technical Report CLRC-TR-01-08, Computer Learning Research Centre, Dept. of Computer Science, Royal Holloway, University of London.

Tompa,M. (1999) An exact method for finding short motifs in sequences, with application to the ribosome binding site problem. In *Seventh International Conference on Intelligent Systems for Molecular Biology*, pp. 262–271.

Vapnik,V.N. (1998) *Statistical Learning Theory*. Wiley, New York.

Watkins,C. (2000) Dynamic alignment kernels. In Smola,A. Bartlett,P. Schölkopf,B. and Schuurmans,D. (eds.), *Advances in Large Margin Classifiers*, MIT Press, Cambridge, MA, pp. 39–50.

Zien,A., Rätsch,G., Mika,S., Schölkopf,B., Lengauer,T. and Müller,K.-R. (2000) Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, **16**, 799–807.