Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites

A. Zien[†*], G. Rätsch[‡], S. Mika[‡], B. Schölkopf[§], T. Lengauer[†], K.-R. Müller[‡]

[*] *To whom correspondence should be addressed*

[†] *GMD.SCAI, Schloss Birlinghoven, 53754 Sankt Augustin, Germany*

{zien, lengauer}@gmd.de

[‡] *GMD.FIRST, Kekuléstraße 7, 12489 Berlin, Germany*

{raetsch, mika, klaus}@first.gmd.de

[§] *Microsoft Research, 1 Guildhall Street, Cambridge CB2 3NH, UK*

bsc@microsoft.com

Abstract

Motivation: In order to extract protein sequences from nucleotide sequences, it is an important step to recognize points at which regions start that code for proteins. These points are called translation initiation sites (TIS).

Results: The task of finding TIS can be modeled as a classification problem. We demonstrate the applicability of support vector machines (SVMs) for this task, and show how to incorporate prior biological knowledge by engineering an appropriate kernel function. With the described techniques the recognition performance can be improved by 26% over leading existing approaches. We provide evidence that existing related methods (e.g. ESTScan) could profit from advanced TIS recognition.

Contact: Alexander.Zien@gmd.de, Gunnar.Raetsch@first.gmd.de.

Introduction

Living systems are determined by the proteins that they produce based on their genomes. But only parts of the genomic text in fact code for proteins. These parts are called coding sequence (CDS). Therefore, given a piece of DNA or mRNA sequence, it is a central problem in computational biology to determine whether it contains CDS, and, if so, for which protein it codes.

In principle, both CDS and the encoded protein can be characterized using alignment methods. Programs capable of aligning nucleotide sequences to protein databases include FASTX/FASTY (Pearson *et al.*, 1997), SearchWise (Birney *et al.*, 1996) and BLASTX (Gish & States, 1993). However, this approach is hampered by two severe problems: First, there are several sources of noise making the task more difficult and error-prone than pure protein alignment: (i) The correct strand and reading frame have to be found. (ii) Additional false hits may result from misinterpreting non-coding sequence as CDS. (iii) Sequencing errors may disrupt the correct reading frame. This is a particularly strong problem for low quality sequences like the popular expressed sequence tags (ESTs). Second, approaches based on alignment rely on homologous proteins being known. Thus they cannot be used to find novel genes. Hence, a method to identify CDS in nucleotide sequences is desirable, both in order to ease the task for alignment-based approaches and to find new genes.

Since living cells are able to distinguish between CDS and other nucleotide sequence parts without utilizing any homology information, this should also be possible for computer programs, in principle. In fact, there are algorithms that identify CDS merely relying on properties intrinsic to nucleotide sequences. The most successful programs include GENSCAN (Burge & Karlin, 1997) for genomic DNA and ESTScan (Iseli *et al.*, 1999) for ESTs. ESTs are single-read partial sequences derived from mRNA that are particularly error-prone. ESTScan implements a fifth-order hidden Markov model that simultaneously recognizes CDS by typical oligo-nucleotide frequencies and corrects sequencing errors. It does not incorporate a model of translation initiation site (TIS) sequences, although they mark the beginning of CDS. GENSCAN employs generalized hidden Markov models to capture the structure of an entire genome. It incorporates probabilistic models of DNA signal sequences including TIS, stop codons and splice sites, as well as compositional features and length distributions of different genomic regions. Despite its overall sophistication, GENSCAN uses a relatively crude TIS model: a piece of sequence is assigned a probability for being a TIS, based on the positional relative frequencies of individual nucleotides observed around a true TIS.

There is a number of more elaborate models for TIS. Salzberg extends the positional probabilities (as used by GENSCAN) to first-order Markovian dependencies (Salzberg, 1997). This is essentially a proper probabilistic model of positional di-nucleotides, and leads to a significant increase in recognition performance. There also are methods to explicitly capture correlations between non-adjacent positions near TIS or other signals (Agarwal & Bafna, 1998b), possibly providing insight into the mechanisms of translation initiation. However, since few such correlations can be proved to be significant in TIS sequences, they afford little gain for TIS recognition.

All models discussed so far can be called generative, as they can be used to generate potential TIS sequences with approximately the true probability distribution. Applying such

models, a sequence is considered a TIS if the probability with which the sequence is generated by the model exceeds some threshold. The more closely the true distribution is approximated, the better this approach works. By using so-called discriminative methods, often a superior distinction can be achieved between true TIS and similarly looking pieces of sequence (called *pseudo sites*). These methods aim at learning to discriminate certain objects from others, without explicitly considering probability distributions.

For example, the program `ATGpr` (Salamov *et al.*, 1998) uses a linear discriminant function that combines several statistical measures derived from the sequence. Each of those features is designed to discriminate between true and pseudo-TIS. Learning allows to find a (linear) weighted combination of features that achieves a high level of discrimination on the training set as well as on the test set.

A radically different approach to learning a discriminating function is taken by Pedersen and Nielsen (Pedersen & Nielsen, 1997). They train an artificial neural network (NN) to predict TIS from a fixed-length sequence window around a potential start codon (ATG). The input of the NN consists of a binary encoding of the sequence; no higher-level features are supplied. The intriguing idea is that the NN learns by itself which features derived from the sequence are indicative of a true TIS.

Of the described methods, only `ATGpr` makes use of the ribosome scanning model (Kozak, 1989). According to this model, the translation starts at the first occurrence of a start codon in the mRNA, and thus other start codons further downstream are inactive (pseudo sites). However, it is now known that nucleotides adjacent to the start codon are also relevant for translation initiation, e.g. (Kozak, 1997). The scanning model can be combined with any TIS recognition method, and is confirmed by the resulting improvements of recognition (Agarwal & Bafna, 1998a). The model is orthogonal to TIS signal sequence recognition itself and is limited to complete mRNA sequences, which prohibits application to ESTs. Therefore, we will not consider it in the following.

In this paper, we show that we can outperform established methods for TIS recognition by applying support vector machines (SVMs) (Boser *et al.*, 1992; Vapnik, 1995). Like NNs, SVMs are a discriminative supervised machine learning technology, i.e. they need training with labeled empirical data in order to learn the classification. For the task of TIS recognition, we show that SVMs can be superior to NNs. To achieve this performance gain we use a particularly valuable property of SVMs: the ability to adapt them to the problem at hand by including prior knowledge into the so-called kernel function. Here, we demonstrate how to incorporate basic knowledge of the translation process. The paper is structured as follows: we first give a brief description of the SVM technique, then present experiments and finally discuss results and potential applications.

System and methods

Support Vector Machines

Formally, SVMs, like any other classification method, aim at estimating a classification function $f : \mathcal{X} \rightarrow \{\pm 1\}$ using labelled training data from $\mathcal{X} \times \{\pm 1\}$ such that f will correctly classify unseen examples (test data). In our case, $\mathcal{X}$ will contain simple representations of sequence windows, while ± 1 corresponds to true TIS and pseudo sites, respectively.

In order to be successful, two conditions have to be respected. First, the training data must be an unbiased sample from the same source as the test data. Technically speaking, training and test data have to obey the same underlying probability distribution. This concerns the experimental setup. Second, a measure of the size of the class of functions from which we choose our estimate f, the so-called capacity of the learning machine, has to be sensibly restricted. If the capacity is too small, complex discriminant functions cannot be sufficiently well approximated by any selectable function f – the learning machine is too simple to learn well. On the other hand, too large a capacity bears the risk of losing the ability to learn a function that generalizes well to unseen data. The reason lies in the existence of infinitely many functions that are consistent with the training examples, but disagree on unseen (test) examples. Those functions would perfectly memorize the particular examples used for training, but could not generalize. Picking such a function is called overfitting.

In neural network training, overfitting is avoided by early stopping, regularization or asymptotic model selection (Bishop, 1995; Orr & Müller, 1998). In contrast, the capacity of SVMs is limited according to the statistical theory of learning from small samples (Vapnik, 1995). For learning machines implementing linear decision functions, one way of limiting the capacity is to enforce a large margin of separation between the classes. The margin is the minimal distance of training points to the separation surface. Finding the maximum margin separation can be cast into a convex quadratic programming (QP) problem (Boser *et al.*, 1992). The time complexity of solving such a QP scales approximately between quadratic and cubic in the number of training patterns (see (Schölkopf *et al.*, 1999)).

In order to maximize the generalization power, often it is profitable to misclassify some outlying training data points in order to enlarge the margin between the other training points. This is theoretically founded by statistical learning theory (Vapnik, 1995). This 'neglectful' learning strategy also masters inseparable data (Cortes & Vapnik, 1995; Schölkopf *et al.*, 2000), which frequently occur in real-world applications. See Figure 1 (a) for an example. The trade-off between margin size and number of misclassified training points is controlled by a parameter of the SVM, which can therefore be used to control its capacity. This extension still permits optimization via QP (Cortes & Vapnik, 1995).

It is tempting to think that linear functions can be insufficient to solve complex classification tasks. A little thought reveals that, in fact, this depends on the representation of the data points. Frequently, natural definitions of input space are used that tend to minimize dimensionality and avoid redun-

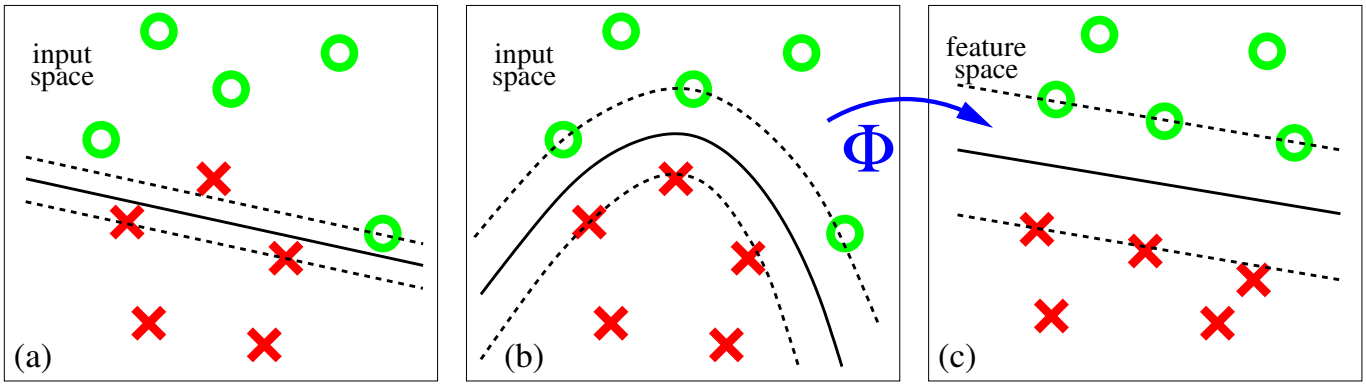

Fig. 1. Three different views on the same dot versus cross separation problem. The data points closest to the separation line are called support vectors. (a) In this example, a linear separation of the input points is not possible without errors. Even the misclassification of one data point permits only a small margin. The resulting linear classification function looks inappropriate for the data. (b) A better separation is permitted by nonlinear surfaces in input space. (c) These nonlinear surfaces correspond to linear surfaces in feature space. Data points are mapped from input space to feature space by the function Φ that is implied by the kernel function k (see main text).

dancy. Then, linearity may easily be too restrictive. However, we are free to define (possibly redundant) features that nonlinearly derive from any number of input space dimensions. Even for complex problems, well chosen features could ideally be related to the respective classification by rather simple means, e.g. by a linear function (cf. Figure 1).

Any linear learning machine can be extended to functions that are nonlinear in input space $\mathcal{X}$ by explicitly transforming the data into a feature space $\mathcal{F}$ using a nonlinear map $\Phi : \mathcal{X} \to \mathcal{F}$ (see Figure 1). SVMs can do so *implicitly*, since all information that we need to supply to the SVM for both training and classification are inner products of pairs of data points $\Phi(x), \Phi(y)$ in feature space $\mathcal{F}$. Thus, we only need to supply a so-called kernel function that computes these inner products. This kernel function k implicitly defines the feature space via

$$k(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle.$$

Not every function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a valid kernel function. The map Φ and the corresponding feature space are guaranteed to exist for functions that satisfy Mercer's condition (see, e.g., (Boser *et al.*, 1992)). Using kernel functions to compute the dot products, we can computationally afford very large (e.g. 10^{10}-dimensional) feature spaces. SVMs can still avoid overfitting thanks to the margin maximization mechanism. Simultaneously, they can learn which of the features implied by k are distinctive for the two classes. Thus, instead of having to design well-suited features by ourselves (which can often be difficult), we can use the SVM to select them from a sufficiently rich feature space. Of course, it will be helpful if the kernel supplies a set of features related to the correct classification. In the following sections, we will show how to boost the process of learning by choosing appropriate kernel functions.

Data sets

Little experience exists in the application of SVMs to biomolecular problems (we only know of work on remote homology detection (Jaakkola *et al.*, 1999) and on gene expression analysis (Brown *et al.*, 2000)). Therefore, we compare the performance of our SVMs to that of the most popular alternative general purpose machine learning technology, neural networks (NNs), trained by NN experts on the same problem domain. In order to do so, we use the NN results and the vertebrate TIS set provided by Pedersen and Nielsen (Pedersen & Nielsen, 1997) as described below. We take care to only replace the learning machinery while retaining the setting: the definition of training and test data sets as well as the definition of input space. We also compare our SVM results to the performance of another successful TIS recognition method, the positional conditional probability method (Salzberg, 1997).

The original sequence set of Pedersen and Nielsen has been assembled from high quality nuclear genomic sequences of a selected set of vertebrates taken from GenBank (Benson *et al.*, 1998). All introns were removed, in analogy to the splicing of mRNA sequences. Only high quality entries with at least 10 nucleotides upstream and 150 downstream of the start codon were selected. In order to avoid over-optimistic performance estimates resulting from biased data samples, the set was thoroughly reduced for redundancy. As a consequence, the results below represent lower limits to the performance to be expected on real world data, which is heavily redundant. The data selection protocol left 3312 sequences (see (Pedersen & Nielsen, 1997)). From the work of other investigators, e.g. (Burge & Karlin, 1997), we expect typical features of TIS to differ for different branches in the evolutionary tree. This implies that the trained classificator will only be valid for mammals, and that retraining on other sequence sets will be necessary for different groups of species.

From the described set of sequences, Pedersen and Nielsen construct the data set for TIS recognition in the following way (personal communication). For each potential start codon (the nucleotide sequence `ATG`) on the forward strand, one data point is generated. This leads to 13503 data points, of which 3312 (24.5%) represent true TIS and the rest (10191 points, 75.5%) represent pseudo sites. We prefer this skewed distri-

bution to a balanced data set, as it is a (crude) approximation to the situation that is expected for real ESTs. Each data point is represented by a sequence window of 200 nucleotides centered around the respective ATG triplet. For triplets near the borders of the available sequence, the positions missing from the 200 nucleotide window are filled with N, the symbol for unknown. Pedersen and Nielsen divide the data into six parts of nearly equal size (≈ 2200 points each) and equal fraction of true TIS. Each part is in turn reserved for testing the classification learned from the other five parts.

Engineering the kernel function for TIS recognition

We define the input space by the same sparse bit-encoding scheme as used by Pedersen and Nielsen (personal communication): each nucleotide is encoded by five bits, exactly one of which is set. The position of the set bit indicates whether the nucleotide is A, C, G or T, or N (for unknown). This leads to an input space of dimension $n = 1000$. Experiments with more compact representations (two-digit encoding of the four nucleotides with an appropriate intermediate state for unknown, data not shown) indicate that the sparse encoding performs best.

Let $\mathbf{x}$ and $\mathbf{y}$ be n-dimensional vectors, representing two inputs. The simple polynomial function

$$k(\mathbf{x}, \mathbf{y}) \quad = \quad \langle \mathbf{x}, \mathbf{y} \rangle^d$$

is a valid kernel that induces $\frac{(n+d-1)!}{d!(n-1)!}$ monomial features of degree d. Precisely, there is one feature $\Phi_{\mathbf{m}}(\mathbf{x})$ of the form

$$\Phi_{\mathbf{m}}(\mathbf{x}) \quad = \quad \sqrt{\frac{d!}{\prod_{i=1}^n \mathbf{m}_i!}} \prod_{i=1}^n \mathbf{x}_i^{\mathbf{m}_i}$$

for every $\mathbf{m} \in \mathbb{N}^n, \sum_{i=1}^n \mathbf{m}_i = d$. This is proved by showing that $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \left(\sum_{i=1}^n \mathbf{x}_i \mathbf{y}_i\right)^d = \langle \mathbf{x}, \mathbf{y} \rangle^d = k(\mathbf{x}, \mathbf{y})$ (Schölkopf, 1997). Note that, using the sparse encoding described above, the dot product $\langle \mathbf{x}, \mathbf{y} \rangle$ simply counts the number of nucleotides that coincide in the two sequences represented by $\mathbf{x}$ and $\mathbf{y}$.

For this kernel, setting the degree d to one leads to a linear separation in input space, since then Φ becomes the identity function and the feature space is identical to the input space. Thus, the features correspond to positional nucleotide incidences, and the SVM learns positional preferences. By setting the degree d to two, we can let the feature space reflect all pairwise correlations of the nucleotide frequencies at any two sequence positions. Mathematically, for an input vector $\mathbf{x} = (x_1, \ldots, x_n)$, all features of the kind $x_i x_j$, $1 \leq i, j \leq n$, are represented. A degree of three corresponds to all correlations of (possibly scattered) triplets, and so on. In order to determine a good value for the degree d, we train with different values of d, using only part of the training data, and validate performance on the reserved training data (cross-validation). We investigate polynomials of first to fifth degree ($d = 1, \ldots, 5$). With this simple polynomial kernel function we already achieve results competitive to those of the NN devised by Pedersen and Nielsen (see Table I).

Table I also shows that the SVM provides a discrimination power superior to that of the generative model of conditional positional probabilities for nucleotides as suggested by Salzberg (Salzberg, 1997). According to this model, TIS score is computed from log ratios of empirically estimated probabilities. Some rare events (positional di-nucleotides) may be observed in the test data but not in the training set. In order to avoid infinite scores arising from the resulting zero probabilities, pseudo counts are introduced. The observed frequency of each dinucleotide at each position is increased by 0.5 before it is used for the estimation of the corresponding conditional probabilities. For the purpose of comparison, we run the method of Salzberg on the same input data as the other methods, i.e. 200 nucleotides from a five letter alphabet. The method works surprisingly well and is competitive to the NN.

Although the SVM with the polynomial kernel already performs better than both established methods, the results can still be improved by modifying the kernel function. We design an improved kernel function by incorporating the basic biological hypothesis that, while certain local correlations are typical for TIS, dependencies between distant positions are of minor importance or do not even exist. We want the feature space to reflect this. Thus, we modify the kernel utilizing a technique that is described in (Schölkopf *et al.*, 1998): At each sequence position, we compare the two sequences locally, within a small window of length $2l+1$ around that position. Again, we count matching nucleotides, this time multiplied with weights $\mathbf{w}$ increasing from the boundaries to the center of the window. The resulting weighted counts are taken to the d_1^{th} power. d_1 reflects the order of local correlations (within the window) that we expect to be of importance.

$$\text{win}_p(\mathbf{x}, \mathbf{y}) = \left(\sum_{j=-l}^{+l} \mathbf{w}_j \ \text{match}_{p+j}(\mathbf{x}, \mathbf{y}) \right)^{d_1}$$

Here, $\text{match}_{p+j}(\mathbf{x}, \mathbf{y})$ is 1 for matching nucleotides at position $p+j$ and 0 otherwise. The window scores computed with win_p are summed over the whole length of the sequence. Correlations between up to d_2 windows are taken into account by raising the resulting sum to the power of d_2.

$$k(\mathbf{x}, \mathbf{y}) = \left(\sum_{p=1}^{l} \text{win}_p(\mathbf{x}, \mathbf{y}) \right)^{d_2}$$

We call this kernel locality-improved. Similar to the polynomial kernel, each window score is a kernel that induces a set of monomial features of degree d_1. The monomials are weighted in order to strengthen the representation of correlations of sequence positions that are close to each other. Only correlations of positions within the window size are represented by the window scores. The second level polynomial induces monomial features of degree $d_1 d_2$ that combine any d_2 intra-window monomials. Thus, distant correlations are taken into account by values $d_2 > 1$. Intuitively, it is clear that this function is a valid kernel function, since it corresponds to the application of a weighted polynomial map with degree d_2 to an intermediate space that is defined as feature space of a weighted polynomial map with degree d_1 on the input space.

algorithm	parameter setting	overall error	specificity	sensitivity	Mathews correlation	mutual information
neural network		15.4%	64.5%	82.4%	62.7%	0.192
Salzberg method		13.8%	73.7%	68.1%	61.9%	0.250
SVM, simple polynomial	d=1	13.2%	75.7%	69.2%	63.9%	0.267
SVM, locality-improved kernel	d_1=4,l=4	11.9%	79.3%	70.0%	66.9%	0.292
SVM, codon-improved kernel	d_1=2,l=3	12.2%	78.7%	69.0%	65.9%	0.283
SVM, Salzberg kernel	d_1=3,l=1	11.4%	76.0%	78.4%	69.6%	0.326

Table I. Comparison of classification errors (measured on the test sets) achieved with different learning algorithms. All results are averages over the six data partitions (see main text). SVMs are trained on 8000 data points with combinations of parameters as described in the text. An optimal set of parameters is selected according to the overall error on the remaining training data ($\approx$ 3300 points); only these are presented. Note that the windows consist of $2l + 1$ nucleotides. The NN results are those achieved by Pedersen and Nielsen ((Pedersen & Nielsen, 1997), personal communication). There, model selection seems to have involved test data, which might lead to slightly over-optimistic performance estimates. Positional conditional preference scores are calculated analogously to Salzberg (Salzberg, 1997), but extended to the same amount of input data also supplied to the other methods. Note that all performance measures shown depend on the value of the classification function threshold. For SVMs, the thresholds are by-products of the training process; for the Salzberg method, 'natural' thresholds are derived from prior probabilities by Bayesian reasoning. Overall error denotes the ratio of false predictions to total predictions. The sensitivity versus specificity trade-off can be controlled by varying the threshold (see Figure 2). The Mathews correlation coefficient is the (Pearson) correlation coefficient of true and predicted labels in an arbitrary numerical representation. The mutual information between true and predicted labels is given in bits and would be 0.804 for a perfect prediction. Note also that all performance measures but sensitivity are sensitive to the relative numbers of TIS to pseudo sites in the data set (here, $\approx$1:3).

Formally, it can be proved that linear combinations of kernels with positive coefficients and positive powers of kernels are valid kernels (Schölkopf *et al.*, 1999; Schölkopf, 1997).

This kernel function poses the problem of how to set a number of parameters (in addition to the general parameter for SVM capacity control described above). We systematically investigated only the case $d_2 = 1$, since some test runs with $d_2 > 1$ yielded inferior performance (data not shown). This is consistent with our intuition that long distance correlations are of minor importance. For each of the remaining parameters, we select a small number of values in an appropriate range. SVMs are trained with all combinations of these values, while excluding a part of the training set as validation set. This validation part is then used to measure the performance of the trained SVM and to select the corresponding parameters. We investigate window sizes $(2l + 1)$ ranging from one to eleven, taking into account some possibly relevant biological numbers. Biological features that we consider important include oligo-nucleotide composition, interactions between neighboring amino acids and the assumed number of nucleotides that the ribosome can have contact with at the same time. For the case of $l = 0$ this kernel reduces to a simple polynomial kernel. We always obtain superior results for larger values ($l > 0$), indicating that local correlations are indeed of special importance. For the degree of local correlations (d_1), we consider values up to five. In Table II the TIS recognition performance of this type of kernel is compared to that of the polynomial kernel for differently sized training sets.

Table II shows that the optimal parameterization of the kernel depends on the training set size. The table also shows that the performance improvements over the polynomial increase for larger training sets. This suggests that carefully designed kernel functions are useful even in presence of a wealth of training data.

In an attempt to further improve performance we try to incorporate another biological hypothesis into the kernel, this time concerning the codon-structure of coding sequence. A codon is a triplet of adjacent nucleotides that codes for one amino acid. By definition the difference between a true TIS and a pseudo site is that downstream of a TIS there is CDS (which shows codon structure), while upstream there is not. CDS and non-coding sequence show statistically different compositions. It is likely that the SVM exploits this difference for classification. We could hope to improve the kernel by reflecting the fact that CDS shifted by three nucleotides still looks like CDS. Therefore, we further modify the locality-improved kernel function to account for this translation-invariance. In addition to counting matching nucleotides on corresponding positions, we also count matches that are shifted by three positions. We call the resulting kernel codon-improved. Except for the modified matching function, it is given by the same expressions as the locality-improved kernel. Again, this function can be shown to be a valid kernel by explicitly deriving the monomial features.

Tables I and II suggest that this modification actually seems to decrease performance. This is disappointing, since a similar modification to the simple polynomial kernel leads to an increase of recognition accuracy (data not shown), which, however, is smaller than the increase by the locality-improvement. We could imagine that the process of learning some relevant features (e.g. strong positional preferences near to the start codon) is distorted by the modification. On the other hand, the other kernels are already capable of learning translation-invariance if they are given enough training data and if this proves advantageous for the classification.

Another direction for modification of the kernel function is suggested by the good performance of the method of Salzberg. In order to integrate his idea into a kernel function, we do not

SVM kernel function	400 data points			1000 data points			4000 data points		
	params	error	Mcc	params	error	MCC	params	error	MCC
simple polynomial	d=2	18.1%	46.0%	d=2	16.0%	54.3%	d=1	13.6%	62.3%
locality-improved	d_1=3,l=2	18.0%	48.1%	d_1=4,l=3	15.9%	53.9%	d_1=3,l=3	12.6%	64.6%
codon-improved	d_1=1,l=2	18.4%	45.7%	d_1=1,l=2	15.6%	55.5%	d_1=1,l=2	13.6%	62.2%
Salzberg kernel	d_1=1,l=1	14.7%	57.1%	d_1=1,l=1	13.6%	61.2%	d_1=3,l=1	11.7%	68.8%

Table II. Performance of kernel functions (measured on the test sets) computed as described in Table I, but trained on differently sized subsets of the training data set. The columns show the optimal parameter settings (params), determined as described in Table I, together with the corresponding overall classification error (error) and Mathews correlation coefficient (Mcc).

calculate the product of the conditional probabilities over the whole sequence, but instead calculate the log odds of the conditional probabilities for each position separately:

$$s_p(\mathbf{x}) = \log \frac{P(\mathbf{x}_p \text{ at pos. } p \text{ in } \mathit{TIS} | \mathbf{x}_{p-1} \text{ at pos. } p-1 \text{ in } \mathit{TIS})}{P(\mathbf{x}_p \text{ at pos. } p \text{ in } \mathit{ANY} | \mathbf{x}_{p-1} \text{ at pos. } p-1 \text{ in } \mathit{ANY})}$$

Here, P denotes estimated probabilities derived from training set counts plus pseudo counts, $\mathbf{x}_p$ is the nucleotide incident at position p in the sequence corresponding to data point $\mathbf{x}$, TIS is the set of training sequences centered around TIS, and ANY is the set of all training sequences (both centered around TIS or pseudo sites). This way, we define a new input space. Each data point is represented by a sequence of log odd scores $s_p(\mathbf{x})$ relating, individually for each position, two probabilities: first, how likely the observed nucleotide at that position derives from a true TIS and second, how likely that nucleotide occurs at the given position relative to any ATG triplet. We then proceed analogously to the locality-improved kernel, replacing the sparse bit representation by the sequence of these scores. As expected, this leads to a further increase in classification performance. The results are shown in Figure I. However, a similar gain could be expected for the neural network if it was trained on the log odd scores as input values.

In conclusion, all three engineered kernel functions clearly outperform the NN as devised by Pedersen and Nielsen by reducing the overall number of misclassifications by about 25% (see Table I). The SVM also beats the performance of positional conditional probabilities, which work surprisingly well when applied to larger windows than suggested by Salzberg. The SVM results show more false positives and fewer false negatives than the NN, corresponding to a higher level of specificity. However, the sensitivity versus specificity trade-off can be controlled by setting the threshold value of the classification function (see Figure 2). From this point of view, again, the SVMs do best.

In order to allow for this performance comparison, all calculations reported above are based on the same representation of input space (encoding a five letter alphabet) as was used for the neural network. We repeated some of the experiments using a more natural representation that distinguishes between four letters only, each representing a real nucleotide. Unknowns are translated into a probability distribution over the four nucleotides which is determined from the sequences. Using this representation, the method of Salzberg performs considerably worse than before: the overall error rate increases to 21.2%

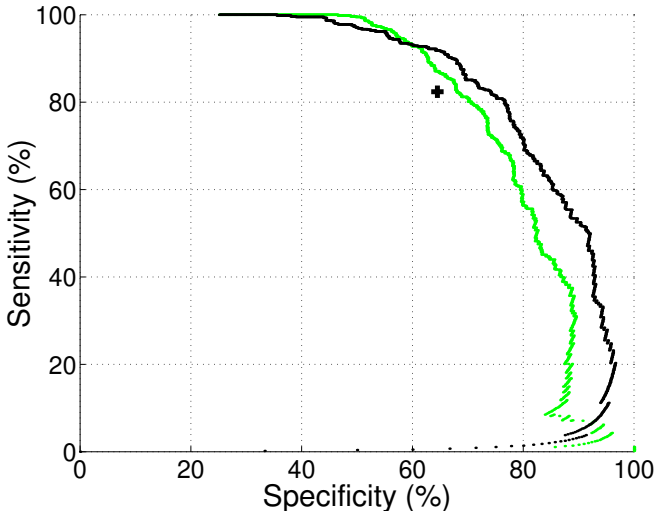

Fig. 2. Specificity versus sensitivity trade-off for the method of Salzberg (gray line) and for the SVM with Salzberg kernel (black line) on the test data for the first data partitions. Specificity is the rate of correctly predicted TIS (true positives) to all predicted TIS; sensitivity is the rate of correctly predicted TIS to all true TIS. The SVM clearly improves sensitivity for a wide range of reasonable values of specificity (60-97%). The performance of the neural network by Pedersen and Nielsen is marked by a black plus symbol (+).

(corresponding to specificity 71.6%, sensitivity 22.7%, Mathews correlation 31.8%, mutual information 0.061). This suggests that the introduction of unknowns is asymmetrical for TIS and pseudo sites, and that the method utilizes this bias for the classification. However, the SVM using the Salzberg kernel performs almost as well on the four letter representation as before: the overall error is 11.7% (Sp 76.6%, Sn 75.4%, Mcc 68.3, MI 0.311). This shows that the algorithm does well on data that is relevant for real world application.

From an application point of view, performance measures may be considered more relevant on a per sequence basis than per ATG, as investigated so far. Thus, we re-evaluate the most successful method, the SVM with the Salzberg kernel, in a manner that is appropriate for complete mRNA sequences: for each of the original 3312 sequences (cf. description of data sets), we predict exactly one TIS. The TIS is predicted to be the highest scoring ATG triplet within the sequence (foreward strand only), regardless of the absolute score value. Since each sequence contains one TIS, we have no true negatives. This

method	number of predictions	correct predictions	false predictions	missed TIS
Salzberg method	3312	2217	1095	1095
SVM, Salzberg kernel	3312	2782	530	530
ESTScan, left ATG	2350	208	2142	3104
ESTScan, right ATG	2350	1614	736	1698
ESTScan, closest ATG	2350	1621	729	1691

Table III. Evaluation of the SVM with the best kernel function on a per-sequence basis on the original set of 3312 sequences. The sequences are supplied in the natural four-letter representation as described in the main text. For comparison purposes, we also show the corresponding results for the original Salzberg method. In both cases, for each sequence the highest scoring ATG codon is predicted to be the TIS. We also apply ESTScan (with default parameters) to all 3312 sequences. For each predicted CDS, an ATG triplet near the supposed start point of the CDS is selected as predicted TIS. The evaluation is shown for three different selection strategies, as indicated in the leftmost column.

renders usual performance measures (including Sn, Sp, Mcc and MI), as used above, inappropriate. Instead, we supply in Table III the raw counts of correct and false predictions. The figures an upper bound for the performance on our target application area, ESTs. In ESTs, in contrast to complete mRNAs, the TIS may not be covered. Thus, the problem of TIS recognition becomes more difficult, since we only know that there is *at most* one TIS contained in the sequence. Therefore, this investigation cannot replace a thorough evaluation on a benchmark that is designed for this purpose and includes ESTs that do not contain a TIS. We intend to develop such a benchmark.

In order to get a first impression how much a program like ESTScan could profit from an advanced TIS recognition module, we applied it to the same set of 3312 mRNA-like sequences. ESTScan aims at identifying CDS within ESTs as accurately as possible. In a slight misuse of the program, we investigate how confidently it predicts the correct TIS (the start position of the CDS) for each sequence. The results are also shown in Table III. On average, the ESTScan misses the true TIS position by 41.6 nucleotides. Both this figure and the table indicate that ESTScan could profit from a TIS recognition module. For genomic sequences and programs like GENSCAN, a similar situation can be expected.

Discussion

TIS recognition can be used to improve reliability and accuracy of amino acid predictions from nucleotide sequences. Two major fields of application are distinguished by their different data types: ESTs and genomic sequences. The data set used in this paper consists of computationally spliced sequences that resemble mRNA and thus is more tuned towards EST data, which is our target area of application. In order to fit our method more closely to this data type and re-evaluate the performance, we are currently building a more realistic data set from real ESTs. Since ESTs cover only a fragment of a real mRNA, we cannot be sure that each EST contains a TIS. Thus, we cannot simply predict the highest scoring ATG of each EST to be a TIS, but we also need some global score threshold below which no TIS is predicted for an EST. This is modeled by the all TIS versus all pseudo site discrimination, while the (possibly easier) task of per-mRNA discrimination would be sufficient for complete mRNAs. The fragmentary nature of ESTs also prohibits the utilization of some additional features for the classification, including the total length of the complete CDS and the number of preceeding start codons.

However, caution is necessary if we use our method within a rigorous probabilistic framework like those of GENSCAN or ESTScan. In some sense, the SVM (as well as Pedersen and Nielsen's NN) exploits the differences of the oligo-nucleotide compositions in CDS and non-coding sequence. These compositional preferences are already incorporated in GENSCAN and ESTScan, leading to probability distribution dependencies that must be taken into account. In order to avoid these dependencies, it is easiest to restrict the sequence window presented to the SVM to the ribosome binding site. In addition, it is desirable that the TIS recognition method computes probability values for potential TIS to be true TIS. Meanwhile, it should be useful to heuristically combine our TIS recognition with GENSCAN or ESTScan output. We plan to devote more work to these issues.

We believe that the kind of kernel functions presented in this paper will prove useful for other bioinformatics problems. There are far too many interesting classification tasks in molecular biology than can be covered here, so we restrict ourselves to three problems. Most obviously, our technique can easily be applied to the recognition of other fixed length DNA signals. These include binding sites of regulatory proteins, that are important elements of promoters, enhancers and silencers. Second, we can imagine that the excellent protein classification performance of the Fisher kernel method developed by Jaakkola and Haussler (Jaakkola *et al.*, 1999) can still be improved by considering local amino acid correlations in a manner similar to our locality-improved kernel or to the Salzberg kernel. Third, we believe that SVMs will prove successful for exploiting the information gathered with DNA chips. Here, kernel functions could be engineered that reflect the structure of expression data as collection of unrelated time series. These are fields of interesting future work.

In summary, we have compared the performance of important methods for sequence classification on a bio-molecular problem of practical relevance. We show that SVMs are competitive to other, more frequently used machine learning methods and show a simple way to include prior knowledge to improve performance. We provide evidence that our advanced

TIS recognition can be of use for other existing programs.

Acknowledgments

This work was supported by the BMBF (TargId, 0311615) and by the DFG (JA 379/9-1,7-1). Part of the present work was done while BS was with GMD.FIRST. We thank A. G. Pedersen and H. Nielsen for e-mail discussions, providing their data sets and sharing unpublished data. We acknowledge M. Schwan for help with the experiments. We thank the referees for helpful comments.

References

Agarwal, P. & Bafna, V. (1998a). The Ribosome Scanning Model for Translation Initiation: Implications for Gene Prediction and Full-Length cDNA Detection. In: *Proc. Sixth International Conference on Intelligent Systems for Molecular Biology*, (Istrail, S., Pevzner, P., & Waterman, M. S., eds) volume 6 pp. 2–7, Menlo Park, California 94025: AAAI Press.

Agarwal, P. & Bafna, V. (1998b). Detecting non-adjoining correlations within signals in DNA. In: *Proc. Second Annual Conference on Research in Computational Molecular Biology*, (Istrail, S., Pevzner, P., & Waterman, M., eds) volume 2 pp. 2–7, New York 10036: The Association for Computing Machinery.

Benson, D. A., Boguski, M. S., Lipman, D. J., Ostell, J., & Ouellette, B. F. F. (1998). GenBank. *Nucleic Acids Res,* **26** (1), 1–7.

Birney, E., Thompson, J. D., & Gibson, T. J. (1996). PairWise and SearchWise: finding the optimal alignment in a simultaneous comparison of a protein profile against all DNA translation frames. *Nucleic Acids Res,* **24** (14), 2730–2739.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Boser, B., Guyon, I., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In: *Proc. COLT*, (Haussler, D., ed) pp. 144–152, ACM Press.

Brown, M. P. S., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C. W., Furey, T. S., Ares, M., & Haussler, D. (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc Natl Acad Sci USA,* **97** (1), 262–267.

Burge, C. & Karlin, S. (1997). Prediction of Complete Gene Structures in Human Genomic DNA. *J Mol Biol,* **268** (1), 78–94.

Cortes, C. & Vapnik, V. (1995). Support vector networks. *M. Learning,* **20**, 273–297.

Gish, W. & States, D. J. (1993). Identification of protein coding regions by database similarity search. *Nature Genetics,* **3** (3), 266–272.

Iseli, C., Jongeneel, C. V., & Bucher, P. (1999). ESTScan: a program for detecting, evaluating, and reconstructing potential coding regions in EST sequences. In: *Proc. Seventh International Conference on Intelligent Systems for Molecular Biology*, (Lengauer, T., Schneider, R., Bork, P., Brutlag, D., Glasgow, J., Mewes, H.-W., & Zimmer, R., eds) pp. 138–148, Menlo Park, California 94025: AAAI Press.

Jaakkola, T., Diekhans, M., & Haussler, D. (1999). Using the Fisher kernel method to detect remote protein homologies. In: *Proc. Seventh International Conference on Intelligent Systems for Molecular Biology*, (Lengauer, T., Schneider, R., Bork, P., Brutlag, D., Glasgow, J., Mewes, H.-W., & Zimmer, R., eds) pp. 149–158, Menlo Park, California 94025: AAAI Press.

Kozak, M. (1989). The scanning model for translation: an update. *J Cell Biol,* **108** (2), 229–241.

Kozak, M. (1997). Recognition of aug and alternative initiator codons is augmented by g in position +4 but is not generally affected by the nucleotides in positions +5 and +6. *EMBO J,* **16** (9), 2482–2492.

Orr, J. & Müller, K.-R., eds (1998). *Neural Networks: Tricks of the Trade*, volume 1524. Springer LNCS.

Pearson, W. R., Wood, T., Zhang, Z., & Miller, W. (1997). Comparison of DNA Sequences with Protein Sequences. *Genomics,* **46** (1), 24–36.

Pedersen, A. G. & Nielsen, H. (1997). Neural Network Prediction of Translation Initiation Sites in Eukaryotes: Perspectives for EST and Genome analysis. In: *Proc. Fifth International Conference on Intelligent Systems for Molecular Biology*, (Gasterland, T., Karp, P., Karplus, K., Ouzounis, C., Sander, C., & Valencia, A., eds) volume 5 pp. 226–233, Menlo Park, California 94025: AAAI Press.

Salamov, A. A., Nishikawa, T., & Swindells, M. B. (1998). Assessing protein coding region integrity in cDNA sequencing projects. *Bioinformatics,* **14** (5), 384–390.

Salzberg, S. L. (1997). A method for identifying splice sites and translational start sites in eukaryotic mRNA. *Comput Appl Biosci,* **13** (4), 365–376.

Schölkopf, B. (1997). *Support Vector Learning*. PhD thesis. Published by: R. Oldenbourg Verlag, Munich, 1997.

Schölkopf, B., Burges, C. J. C., & Smola, A. J., eds (1999). *Advances in Kernel Methods – Support Vector Learning*. MIT Press.

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

Schölkopf, B., Smola, A., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation,* **12**, 1083 – 1121.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag.