International Journal of Neural Systems
© World Scientific Publishing Company

EVOLUTIONARY OPTIMIZATION OF SEQUENCE KERNELS FOR DETECTION OF BACTERIAL GENE STARTS

BRITTA MERSCH
Abteilung Molekulare Biophysik
German Cancer Research Center, 69120 Heidelberg, Germany
E-mail: b.mersch@dkfz.de

TOBIAS GLASMACHERS
Institut für Neuroinformatik
Ruhr-Universität Bochum, 44780 Bochum, Germany
E-mail: tobias.glasmachers@neuroinformatik.rub.de

PETER MEINICKE
Institut für Mikrobiologie und Genetik, Abteilung für Bioinformatik
Georg-August-Universität Göttingen, 37077 Göttingen, Germany
E-mail: pmeinic@gwdg.de

CHRISTIAN IGEL
Institut für Neuroinformatik
Ruhr-Universität Bochum, 44780 Bochum, Germany
E-mail: christian.igel@neuroinformatik.rub.de

Received (to be inserted
Revised by Publisher)

Oligo kernels for biological sequence classification have a high discriminative power. A new parameterization for the K-mer oligo kernel is presented, where all oligomers of length K are weighted individually. The task specific choice of these parameters increases the classification performance and reveals information about discriminative features. For adapting the multiple kernel parameters based on cross-validation the covariance matrix adaptation evolution strategy is proposed. It is applied to optimize the trimer oligo kernels for the detection of bacterial gene starts. The resulting kernels lead to higher classification rates, and the adapted parameters reveal the importance of particular triplets for classification, for example of those occurring in the Shine-Dalgarno Sequence.

1. Introduction

Kernel-based learning algorithms have been successfully applied to a variety of sequence classification tasks within the field of bioinformatics.[45] Recently, *oligo kernels* were proposed for the analysis of biological sequences,[34] where the term oligo refers to oligomers, short single stranded DNA/RNA fragments. Oligo kernels compare sequences by looking for matching fragments. They allow for gradually controlling the level of position-dependency of the representation, that is, how important the exact position of an oligomer is. Besides good performance[34,35,25], decision functions based on oligo kernels are easy to interpret and to visualize and can therefore be used to infer characteristic sequence features.[34,35]

In the standard oligo kernel, all oligomers are weighted equally. Thus, all oligomers are considered to have the same importance for classification. In general this assumption is not reasonable. In this study, we therefore propose the weighted oligo kernel considering all oligomers of length K (K-mers), in which the relative importance of all K-mers can be controlled individually. A task specific choice of the weighting parameters can potentially increase the classification performance. Moreover, appropriate weights for a particular classification task may reveal sequence characteristics with high discriminative power and biological importance.

The question arises how to adjust the weighting parameters for the K-mers for a given task. In practice, appropriate hyperparameter combinations for kernel-based methods are usually determined by grid-search. This means that the hyperparameters are varied with a fixed step size through a wide range of values and the performance of every combination is assessed using some performance measure. Because of the computational complexity, grid-search is only suitable for the adjustment of very few parameters. Hence, it is not applicable for the adjustment of the 4^K parameters of the weighted oligo kernel. Perhaps the most elaborated systematic technique for choosing multiple hyperparameters are gradient descent methods.[3,4,15,17,25,28,49] If applicable, these methods are highly efficient. However, they have significant drawbacks. In particular, the score function for assessing the performance of the hyperparameters (or at least an accurate approximation of this function) has to be differentiable with respect to all hyperparameters. This excludes reasonable measures such as the standard cross-validation error. Further, the considered space of kernels has to have an appropriate differentiable structure. Depending on the performance measure, gradient-based algorithms for kernel optimization have to be combined with a multi-start strategy to deal with convergence to undesired local extrema.

Here a method for hyperparameter selection is employed that does not suffer from the limitations described above and is less prone to get stuck in undesired local optima. We bring forward to use the covariance matrix adaptation evolution strategy (CMA-ES[21]), an adaptive variable-metric algorithm for efficient direct real-valued optimization, to search for appropriate hyperparameter vectors.[13,26,23]

As an application of our approach to kernel optimization we consider the prediction of bacterial gene starts in genomic sequences. Although exact localization of gene starts is crucial for correct annotation of bacterial genomes, it is difficult to achieve with conventional gene finders, which are usually restricted to the identification of long coding regions. The prediction of gene starts therefore provides a biologically relevant signal detection task, well-suited for the evaluation of our kernel optimization scheme.

We therefore apply the CMA-ES to the tuning of weighted oligo kernels for detecting prokaryotic translation initiation sites, that is, for classifying potential gene starts in bacterial RNA. The performance measure for the hyperparameter optimization is based on the mean classification rate of five-fold cross-validation.

In the following, we first introduce the oligo kernel and our new parameterization. Then the adaptation of kernel parameters using evolutionary optimization methods is described. After that we present the experiments demonstrating the performance of the kernel and the optimization of the hyperparameters.

2. Oligo Kernels

The basic idea of kernel methods for classification is to map the input data, here biological sequences, to a feature space endowed with a dot product. Then the data is processed using a learning algorithm in which all operations in feature space can be expressed by dot products. The trick is to compute these inner products efficiently in input space using a kernel function.[44] Here the feature space can be described in terms of *oligo functions*.[34] These functions encode occurrences of oligomers in sequences with an adjustable degree of positional uncertainty. This is in contrast to existing methods, which provide either position-dependent[8] or completely position-independent representations[31]. For an alphabet $\mathcal{A}$ and a sequence $\mathbf{s}$, which contains K-mer $\omega \in \mathcal{A}^K$ at positions $S^{\mathbf{s}}_{\omega} = \{p_1, p_2, \dots\}$, the oligo function is given by

$$\mu^{\mathbf{s}}_{\omega}(t) = \sum_{p \in S^{\mathbf{s}}_{\omega}} \exp\left(-\frac{1}{2\sigma^2}(t-p)^2\right) \qquad (1)$$

for $t \in \mathbb{R}$. The smoothing parameter σ adjusts the width of the Gaussians centered on the observed oligomer positions and determines the degree of position-dependency of the function-based feature space representation. While small values for σ imply peaky functions, large values imply flatter functions. For a sequence $\mathbf{s}$ the occurrences of all K-mers contained in $\mathcal{A}^K = \{\omega_1, \omega_2, \ldots, \omega_m\}$ can be represented by a vector of m oligo functions. This yields the final feature space representation $\Phi(\mathbf{s}) = [\mu^{\mathbf{s}}_{\omega_1}, \mu^{\mathbf{s}}_{\omega_2}, \ldots, \mu^{\mathbf{s}}_{\omega_m}]^{\mathrm{T}}$ of that sequence. The feature space objects are vector-valued functions. This can be stressed using the notation

$$\boldsymbol{\phi}_{\mathbf{s}}(t) = [\mu^{\mathbf{s}}_{\omega_1}(t), \mu^{\mathbf{s}}_{\omega_2}(t), \ldots, \mu^{\mathbf{s}}_{\omega_m}(t)]^{\mathrm{T}} \ . \tag{2}$$

This representation is well-suited for the interpretation of discriminant functions and visualization.[34] To make it practical for learning, we construct a kernel function to compute the dot product in the feature space efficiently. The inner product of two sequence representations $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$, corresponding to the oligo kernel $k(\mathbf{s}_i, \mathbf{s}_j)$, can be defined as

$$\begin{aligned}\langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle &= \int \boldsymbol{\phi}_i(t) \cdot \boldsymbol{\phi}_j(t) \mathrm{d}t \\ &\propto \sum_{\omega \in \mathcal{A}^K} \sum_{p \in S^i_\omega} \sum_{q \in S^j_\omega} \exp\left(-\frac{1}{4\sigma^2}(p-q)^2\right) \\ &= k(\mathbf{s}_i, \mathbf{s}_j) \end{aligned} \tag{3}$$

writing $\boldsymbol{\phi}_i$ for $\boldsymbol{\phi}_{\mathbf{s}_i}$. The feature space representations of two sequences may have different norms. In order to improve comparability between sequences of different lengths, we compute the normalized oligo kernel

$$\tilde{k}(\mathbf{s}_i, \mathbf{s}_j) = \frac{k(\mathbf{s}_i, \mathbf{s}_j)}{\sqrt{k(\mathbf{s}_i, \mathbf{s}_i) k(\mathbf{s}_j, \mathbf{s}_j)}} \ . \tag{4}$$

From the above definition of the oligo kernel, the effect of the smoothing parameter σ becomes obvious. For the limiting case $\sigma \to 0$ with no positional uncertainty, only oligomers which occur at the same positions in both sequences contribute to the sum. In general it is not appropriate to represent oligomer occurrences without positional uncertainty. This would imply zero similarity between two sequences if no K-mer appears at *exactly* the same position in both sequences. For $\sigma \to \infty$ position-dependency of the kernel completely vanishes. In this case, all terms of oligomers occurring in both sequences contribute equally to the sum, regardless of their distance and the oligo kernel becomes identical to the spectrum kernel.[31]

2.1. *Weighted Oligo Kernel*

So far, the different K-mers are weighted equally in the K-mer oligo kernel. However, some K-mers may be more discriminative than others. Therefore, we introduce new parameters w_i, $i = 1, \ldots, 4^K$, for their weighting and define

$$k_{\text{weighted}}(\mathbf{s}_i, \mathbf{s}_j) = \sum_{\omega \in \mathcal{A}^K} \exp(w_i) \sum_{p \in S^{\mathbf{s}_i}_\omega} \sum_{q \in S^{\mathbf{s}_j}_\omega} \exp\left(-\frac{1}{4\sigma^2}(p-q)^2\right) \ . \tag{5}$$

The normalized weighted oligo kernel $\tilde{k}_{\text{weighted}}$ is then given by

$$\tilde{k}_{\text{weighted}} = \frac{k_{\text{weighted}}(\mathbf{s}_i, \mathbf{s}_j)}{\sqrt{k_{\text{weighted}}(\mathbf{s}_i, \mathbf{s}_i) k_{\text{weighted}}(\mathbf{s}_j, \mathbf{s}_j)}} \ . \tag{6}$$

The parameterization ensures a valid oligo kernel for $w_1, \ldots, w_{4^K}, \sigma \in \mathbb{R}$. This makes unconstrained optimization methods directly applicable to the $1 + 4^K$ kernel parameters.

2.2. *Combined Oligo Kernel*

Meinicke et al. already showed that it is beneficial to employ combinations of oligo kernels that consider oligomers of different lengths.[34] The κ-*combined oligo kernel*

$$\tilde{k}_{\kappa\text{-combined}}(\mathbf{s}_1, \mathbf{s}_2) = \frac{1}{\kappa} \sum_{i=1}^{\kappa} \tilde{k}_i(\mathbf{s}_1, \mathbf{s}_2) \tag{7}$$

was introduced, where the subscript i indicates that the normalized oligo kernel $\tilde{k}_i$ is defined on the oligomers of length i. The level of position-dependency can be controlled for each oligomer length individually using κ parameters $\sigma_1, \ldots, \sigma_\kappa$.

The learning machines using $\tilde{k}_{6\text{-combined}}$ performed better than the machines using single oligo kernels $\tilde{k}_i(\mathbf{s}_1, \mathbf{s}_2)$ with $i = 1, \ldots, 6$.[34,35]

Igel et al. optimized the parameters of the combined oligo kernel $\tilde{k}_{6\text{-combined}}$ using gradient-based optimization of the kernel-target alignment.[25] Roughly speaking, the kernel-target alignment is large if the similarity measure induced by the kernel is large for input patterns of the same class and small for patterns from different classes.[7] Choosing a kernel with large kernel-target alignment aims at choosing a feature space in which class membership and neighborhood of patterns are related. The alignment can be optimized very efficiently, because it only depends on the kernel function and the training data (and not on the learning machine) and is well suited for gradient-based adaptation. However, the search space of potential kernels has to be carefully restricted, because otherwise the kernel-target alignment can lead to overfitting the training data. This could be observed in the study by Igel et al., where optimizing the 6-combined oligo kernel significantly improved the classification results of SVMs, while the authors note that adaptation of the weighted oligo kernel with $K = 3$, having ≈ 10 times more degrees of freedom than $\tilde{k}_{\text{6-combined}}$, led to overfitting.

3. Evolutionary Model Selection

Evolutionary algorithms are iterative, direct, randomized optimization methods inspired by principles of neo-Darwinian evolution theory. They have proven to be suitable for hyperparameter and feature selection for kernel-based learning algorithms.[9,13,14,24,26,27,36,37,42,46,23] Here we apply a real-valued evolutionary algorithm to adapt the parameters of weighted oligo kernels.

Evolution strategies (ES[1]) are one of the main branches of evolutionary algorithms. Here the highly efficient covariance matrix adaptation ES (CMA-ES[19,21]) for real-valued optimization is applied, which learns and employs a variable metric by means of a covariance matrix for the search distribution. The CMA-ES has successfully been applied to tune Gaussian kernels for SVMs considering a cross-validation error as optimization criterion.[13,26] Visualization of the objective function for a benchmark problem depicts an error surface that shows a global trend superimposed by local minima, and ES are usually a good choice for such kind of problems, which are difficult for purely gradient-based algorithms.[13]

In the CMA-ES, a set of μ individuals forming the parent population is maintained. Each individual has a genotype that encodes a candidate solution for the optimization problem at hand, here a real-valued vector containing the hyperparameter combination of the kernel parameters to be optimized. The fitness of an individual is equal to the objective function value—here the five-fold cross-validation error—at the point in the search space it represents. In each iteration of the algorithm, $\lambda > \mu$ new individuals, the offspring, are generated by partially stochastic variations of parent individuals. The fitness of the offspring is computed and the μ best of the offspring form the next parent population. This loop of variation and selection is repeated until a termination criterion is met. The object variables are altered by weighted intermediate recombination and Gaussian mutation. That is, an offspring is created by computing the weighted center of mass of the parents to which a realization of a normally distributed random vector with zero mean and covariance matrix that is updated online using the covariance matrix adaptation method (CMA). The key idea of the CMA is to alter the mutation distribution in a deterministic way such that the probability to reproduce steps in the search space that led to the actual population (i.e., produced offspring that were selected) is increased. This enables the algorithm to detect correlations between object variables and to become invariant under transformations of the search space. The search path of the population over the past generations is taken into account, where the influence of previous steps decays exponentially. The CMA does not only adjust the mutation strengths in m directions, but also detects correlations between object variables. The CMA-ES is invariant under order-preserving transformations of the fitness function and in particular against rotation and translation of the search space—apart from the initialization. The CMA-ES algorithm is described in detail in the appendix.

4. Detection of Bacterial Gene Starts

We apply 1-norm soft margin support vector machines (SVMs[5]) with 3-mer weighted oligo kernels to the detection of prokaryotic translation initiation sites.[18] We first introduce the problem and then the locality improved kernel, as well as simple Markov chain models[30], which we consider for comparison. Then the experimental setup is described. Finally, the results are presented.

4.1. *Problem Description*

To extract protein-encoding sequences from nucleotide sequences is an important task in bioinformatics. For this purpose it is necessary to detect locations at which coding regions start. These locations are called translation initiation sites (TIS). In most cases a TIS contains the start codon ATG or rarely GTG or TTG. The start codon marks the position at which the translation starts. The codon ATG codes for the amino acid methionine, and not every ATG triplet is a start codon. Therefore, it must be decided whether a particular candidate triplet corresponds to a start codon or not. This classification problem can be solved automatically using machine learning techniques, in which the neighborhood of nucleotides observed around potential TISs is used as input pattern to a classifier.[22,25,32,34,35,38,40,43,48,50]

Here we have to distinguish between eukaryotes and prokaryotes, that is, between organisms in which the genetic material is organized into membrane-bound nuclei and organisms without a cell nucleus, like bacteria. In contrast to prediction of eukaryotic TIS there is no biological justification for using a general learning machine across different species for prediction of prokaryotic TIS. For this reason, learning of prokaryotic TISs is always restricted to a limited amount of species-specific examples and model selection methods have to cope with small data sets.

As in previous studies, we tested our approach on *E. coli* genes from the EcoGene database.[41] Only those entries with biochemically verified N-terminus were considered and the neighboring nucleotides were looked up in the GenBank file U00096.gbk.[2] From the 732 positive examples we created associated negative examples. For the negative examples we extracted sequences centered around a codon from the set {ATG, GTG, TTG}. Such a sequence is used as a negative example if the codon is in-frame with one of the correct start sites used as a positive case, its distance from a real TIS is less than 80 nucleotides, and no in-frame stop codon occurs in between. This procedure generates a difficult benchmark data set, because the potential TISs in the neighborhood of the real start codon are the most difficult candidates in TIS discrimination. We created 1248 negative examples. The length of each sequence is 50 nucleotides, with 32 located upstream and 15 downstream with respect to the potential start codon.

To minimize random effects, we generated 50 different partitionings of the data into training and test sets. Each training set contained 400 sequences plus the associated negatives, the corresponding test set 332 sequences plus the associated negatives.

4.2. *Alternative Classification Methods*

We compare the classification performance of our approach with the results achieved with SVMs using the locality improved kernel and simple Markov models.

4.2.1. *Locality Improved Kernel*

The locality improved kernel counts matching nucleotides and considers local correlations within local windows of length $2l+1$.[45,50] For two sequences $\mathbf{s}_i$, $\mathbf{s}_j$ of length L the locality improved kernel computes

$$k_{\text{locality}}(\mathbf{s}_i,\mathbf{s}_j) = \sum_{p=1}^{L}\left(\sum_{t=\max(1,p-l)}^{\min(L,p+l)} v_{t+l-p}\cdot \text{match}_t(\mathbf{s}_i,\mathbf{s}_j)\right)^d \quad . \qquad (8)$$

The function $\text{match}_t(\mathbf{s}_i,\mathbf{s}_j)$ is equal to one if $\mathbf{s}_i$ and $\mathbf{s}_j$ have the same nucleotide at position t and zero otherwise. The weights v_t allow to emphasize regions of the window which are of special importance. In our experiments they are fixed to $v_t = 0.5 - 0.4|l-t|/l$. The hyperparameter d determines the order to which local correlations are considered.

4.2.2. *Markov Model*

As a baseline we consider simple inhomogeneous Markov chain models, also referred as weight array matrix models.[30,40] Given a Markov chain M of order n over an alphabet $\mathcal{A}$ for stings of a fixed length l the likelihood of a sequence $\mathbf{s} = s_1 s_2 \dots s_l$ is given by

$$P^M(\mathbf{s}) = P_1^M(s_1)\cdot P_2^M(s_s|s_1)\dots P_n^M(s_n|s_1,\dots,s_{n-1}) \cdot \prod_{i=n+1}^{l} P_i^M(s_i|s_{i-n},\dots,s_{i-1}) \quad . \qquad (9)$$

The conditional probabilities are estimated form the frequencies in the training data plus a pseudo count c_{pseudo}.[30] A sequence $\mathbf{s}$ is classified based on the sign of $\ln P^{M^+}(s) - \ln P^{M^-}(s)$, where M^+ and M^- denote the Markov models build from positive and negative examples in the training data, respectively.

Our simple Markov model has only two hyperparameters, its order n and the value of the pseudo count c_{pseudo}, which serves as a regularization parameter.

4.3. *Experiments*

In our experiments, we optimize weighted trimer oligo kernels with adjustable σ and 64 weighting parameters as well as the combined oligo kernel $\tilde{k}_{6\text{-combined}}$ with 6 parameters. For comparison the locality improved kernels with hyperparameters l and d, and simple Markov chain models with parameters n and c_{pseudo} were considered.

As kernel-based learning machines we considered 1-norm soft margin SVMs. Therefore we had to adjust an additional hyperparameter, namely the parameter $C \in \mathbb{R}^+$ controlling the regularization of the SVM. For an introduction to SVMs we refer to the standard literature.[5,6,10,44,45] The quadratic optimization problem corresponding to the SVM training is solved efficiently using sequential minimal optimization (SMO[39]) using second order information.[11,16]

For each of the 50 partitionings into training and test data and each classification method independent optimizations of the hyperparameters were conducted.

The parameters l, d, and C of the SVM with locality improved kernel were optimized using three-dimensional grid-search. After determining an interval of parameters leading to well generalizing classifiers, the grid-search varied $l, d \in \{1, \dots, 6\}^{50}$ and $C \in \{j/500 \,|\, j = 1, ..., 10\}$. This amounts to 360 grid points.

For the Markov chain model, we use our results obtained in a previous study[25], where the two parameters $n \in \{0, \dots, 5\}$ (order) and $c_{\text{pseudo}} \in \{j/5 \,|\, 1 \leq j \leq 10\}$ were optimized by grid-search.

The $1 + 4^3 + 1 = 66$ parameters of SVMs with weighted trimer oligo kernels were optimized using the CMA-ES. Objective vectors $\mathbf{x} \in \mathbb{R}$ are mapped to the non-negative hyperparameters according to

$$(w_1, \dots, w_{64}, \sigma, C)^{\mathrm{T}} = (\exp(x_1), \dots, \exp(x_{64}), |x_{65}|, |x_{66}|)^{\mathrm{T}} \ . \quad (10)$$

All evolutionary optimizations started from $x_1 = \dots = x_{64} = 0$ and $x_{65} = x_{66} = 1$ (i.e., the initial triplet weights were one). For each of the 50 partitionings an independent optimization trial was started. The offspring population size was $\lambda = 16$ (e.g., a default choice for this dimensionality[19]) and each trial lasted 200 generations.

In case of the combined oligo kernel $\tilde{k}_{6\text{-combined}}$ we employed the CMA-ES with $\lambda = 9$ and set the maximum number of generations to 100.

The optimization criterion in the grid-searches and the evolutionary optimization was the five-fold cross-validation error based on the classification error. The training data set was partitioned into five disjoint subsets. For each of the subsets, the classifier was trained using the union of the four other sets and a test error was computed on the left-out subset. The final cross-validation error is the average of the five test errors. The partitioning of the data is schematically shown in Fig. 1.

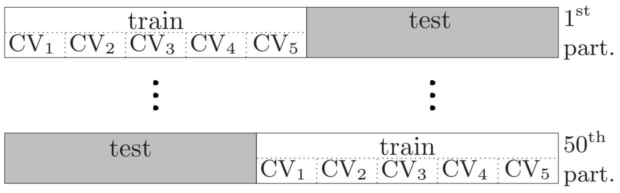

Fig. 1. The experimental results refer to 50 partitionings of the available data into training and test sets. To compute the cross validation error, every training data set is again split into five subsets (CV_1,...,CV_5).

The computational complexity of training and model selection of course varied between the different classifiers in our application. The training of the simple Markov models was much faster compared to the SVMs. The time to determine a final SVM solution was basically proportional to the number of hyperparameter combinations that were tested. For the three parameters of the SVM with locality improved kernel we spent 360 evaluations. To adjust the seven parameters of the SVM with combined kernel we allowed $9 \cdot 100 = 900$ training processes. For the SVM with weighted trimer oligo kernels having 66 parameters we spent $16 \cdot 200 = 3200$ evaluations. As we were not interested in the time needed for training and model selection but only in the quality of the resulting classifiers, we did neither tune the grid-size

An upper bound on the maximum number of generations can be viewed as some kind of regularization of the model selection procedure. Using the cross-validation test error does not prevent the evolutionary model selection from overfitting to the data available nor selected a tight upper bound on the number of generations.

It has to be stressed that 200 generations were more than sufficient for optimizing the trimer oligo kernel. This indicates the good scaling behavior of the CMA-ES.

4.4. *Results*

We first interpret the outcome of the optimization of the parameters of the weighted oligo kernel. Then we compare the classification performance of the oligo kernels, the locality improved kernel and the Markov chain model.

Table 1. Optimized smoothing parameter and regularization parameter for the 64 weight oligo kernel.

	sigma	C
mean	0.81	0.96
25% quantile	0.60	0.71
median	0.69	0.91
75% quantile	0.82	1.11

The results of the optimization of the smoothing parameter σ and the regularization parameter C of the SVM are shown in Table 1. The optimized values of σ are rather small, that is, the position of the triplets is very important.

To analyze the relevance of particular oligomers, the 64 triplets were sorted according to the mean of the corresponding evolved weighting parameters. The weight values indeed vary, see Fig. 2. The triplets on the first 10 ranks are given in Table 1. Besides the start codon ATG also GAG, AGG, and GGA are among the triplets with the largest weight values. These triplets are known to be associated with the so-called Shine-Dalgarno Sequence.[29,47] This sequence is of importance for translation initiation sites because it corresponds to the ribosome binding site.

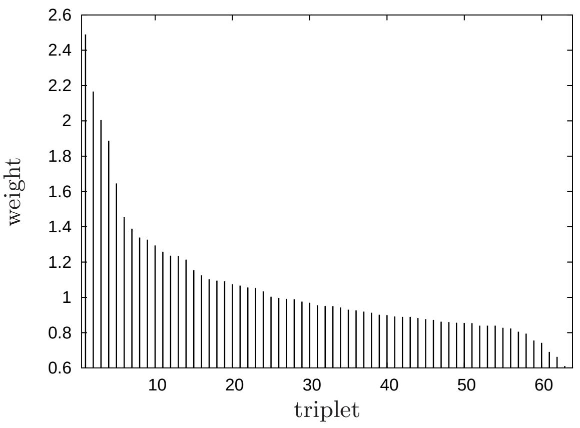

Fig. 2. Means of the weight values of the 64 triplets after optimization sorted by value.

The mean of the weights for the potential start codons were 2.49 for ATG, 1.12 for TTG (rank 16), and 0.82 for GTG (rank 56). That is, the presence of ATG appears to be a relevant feature, whereas GTG and TTG are not as important as ATG. In all positive as well as negative sequence patterns there is a potential start codon at the positions 33–35. Still, the frequency of ATG at this position is considerably higher in positive than in negative examples. The initiation codon of more than 90 % of prokaryotic genes is ATG.[18] The rule of thumb "a pattern is positive if the start codon is ATG and negative otherwise", which would lead to a classification accuracy of about 72% when applied to our data, can be implemented with the evolved kernel weights. However, more sophisticated features based on the triplets with large weights in Table 1 can overrule the presence or ab-

for model building. However, we did not investigate such effects in this study.

sence of ATG.

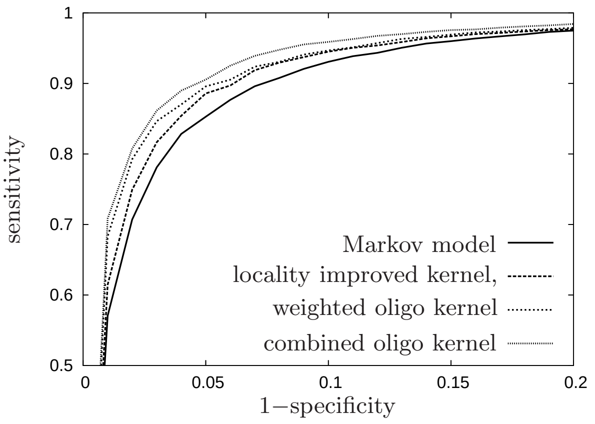

Fig. 3. Median ROC curves of the adapted classifiers based on 50 trials. Here the simple Markov performed worst, the combined oligo kernel best.

The classification results are given in Table 2. The table shows mean values as well as 25%, 50% and 75% quantiles over the 50 partitions of the classification error (accuracy), sensitivity, specificity, and Matthews correlation coefficient.[33] Sensitivity is defined by $\frac{TP}{(TP+FN)}$, specificity by $\frac{TN}{(TN+FP)}$ and Matthews correlation coefficient by

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \,. \tag{11}$$

Here TP, TN, FP, and FN denote the true positives, true negatives, false positives, and false negatives, respectively.

The SVMs with oligo kernels optimized by the CMA-ES gave the best results. The combined oligo kernel achieved an average accuracy of 93.3 %, the weighted oligo kernel follows with a classification performance of 92.9%. Thus, while the trimer oligo kernel provided interesting results about the relevance of particular triplets, it did not outperform the 6-combined oligo kernel.

The means of the locality improved kernel parameters adjusted by grid-search were 2 for l, 4.92 for d, and 0.00684 for C. That is, the nucleotides were only compared within a small window. This is in accordance with the results for σ in the oligo kernels. The median of the classification performance reached by the locality improved kernel is 92.53%. In our scenario, the highly adapted weighted oligo kernel is significantly better than the locality improved kernel.[†]For the Markov model the parameters n and c_{pseudo} were optimized to 1.32 and 0.712 (mean over the 50 trials), respectively. For the Markov model the median of the classification rate is 91.42%, and the weighted oligo kernel is significantly better than the Markov model[†].

Our results are also supported by the ROC (receiver operating characteristic) curves of the optimized classifiers shown in Fig. 3, which were generated by simply shifting the classification threshold.[25] Each curve in the plot corresponds to the median of the 50 trials (similar to the attainment surfaces[12]).

Considering previous results achieved with SVMs and oligo kernels, it is possible to compare the CMA-ES applied to the cross-validation error with the gradient-based optimization of the kernel-target alignment.[25] When applied to the combined oligo kernel, cross-validation and CMA-ES yielded competitive results. While the gradient-based optimization of the kernel-target alignment is computationally more efficient, the evolutionary optimization is more general in the sense that it is not restricted to search spaces having differentiable structure. The cross-validation error proved to be the more robust performance measure compared to the kernel-target alignment. We did not observe overfitting in the evolutionary optimization of the very flexible trimer weighted oligo kernel, whereas the optimization of the kernel-target alignment resulted in overfitting in this example.[25]

5. Conclusion

A task specific choice of the kernel can significantly improve kernel-based machine learning. Often a parameterized family of kernel functions is considered so that the kernel adaptation reduces to real-valued optimization. Still, the adaptation of complex kernels requires powerful optimization methods that can adapt multiple parameters efficiently. When the considered space of kernel functions lacks a differentiable structure or the model selection criterion is non-differentiable, a direct search method is needed. The covariance matrix adaptation evolution strategy

[†]As the data sets in the different experiments of the cross-validation procedure are not fully independent of each other, the preconditions for standard statistical test are not met. Still we think that the test statistics give a good idea about the quality of the results. A Wilcoxon rank-sum test would indicate that the differences are highly statistically significant ($p < 0.001$).

(CMA-ES) is such a powerful, direct algorithm for real-valued hyperparameter selection.

In biological sequence analysis, the CMA-ES allows for a more task specific adaptation of sequence kernels. Because multiple parameters can be adapted, it is possible to adjust new weighting variables in the oligo kernel to control the influence of every oligomer individually. Further, the cross-validation error can directly be optimized (i.e., without smoothing).

We demonstrated the discriminative power of the oligo kernel and the benefits of the evolutionary model selection approach by applying them to prediction of prokaryotic translation initiation sites (TISs). The weighted oligo kernel leads to improved results compared to locality improved kernel, which was optimized by grid-search, as well as to simple Markov models. Furthermore, it is possible to reveal biologically relevant information from analyzing the evolved weighting parameters. For the prediction of bacterial gene starts, for example, triplets referring to the Shine-Dalgarno Sequence are of particular importance for discrimination.

Acknowledgments

The authors thank Nico Pfeifer for his work on the original implementation of the oligo kernel and the TIS classification.

Christian Igel acknowledges support from the German Federal Ministry of Education and Research within the Bernstein group "The grounding of higher brain function in dynamic neural fields".

Appendix A

Details of the CMA-ES

In the following, we present the CMA-ES with weighted recombination and rank-μ-update, $(\mu_{\mathbf{w}}, \lambda)$-CMA-ES.[20] The object parameters $\mathbf{x}_k^{(g+1)}$ of offspring $k = 1, \ldots, \lambda$ created in generation $g + 1$ are given by

$$\mathbf{x}_k^{(g+1)} = \langle \mathbf{x} \rangle_{\mathbf{v}}^{(g)} + \sigma^{(g)} \mathbf{B}^{(g)} \mathbf{D}^{(g)} \mathbf{z}_k^{(g)} \quad , \tag{A.1}$$

where the $\mathbf{z_k^{(g)}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ are independent realizations of an n-dimensional normally distributed random vector with zero mean and covariance matrix equal to the identity matrix $\mathbf{I}$ and

$$\langle \mathbf{x} \rangle_{\mathbf{v}}^{(g)} = \sum_{i=1}^{\mu} v_i \mathbf{x}_{i:\lambda}^{(g)} \tag{A.2}$$

is the weighted mean of the selected individuals with $\sum_{i=1}^{\mu} v_i = 1$ and $v_i > 0$ for $i = 1, \ldots, \mu$. The index $i{:}\lambda$ denotes the i-th best individual. We use superlinear weighted recombination and set $v_i = \ln(\mu + 1) - \ln(i)$. The covariance matrix $\mathbf{C}^{(g)}$ of the random vectors

$$\mathbf{B}^{(g)} \mathbf{D}^{(g)} \mathbf{z}_k^{(g)} \sim \mathrm{N}(\mathbf{0}, \mathbf{C}^{(g)}) \tag{A.3}$$

is a symmetric positive $n \times n$ matrix with

$$\mathbf{C}^{(g)} = \mathbf{B}^{(g)} \mathbf{D}^{(g)} \left(\mathbf{B}^{(g)} \mathbf{D}^{(g)} \right)^{\mathrm{T}} \quad . \tag{A.4}$$

The columns of the orthogonal $n \times n$ matrix $\mathbf{B}^{(g)}$ are the normalized eigenvectors of $\mathbf{C}^{(g)}$ and $\mathbf{D}^{(g)}$ is a $n \times n$ diagonal matrix with the square roots of the corresponding eigenvalues.

In the CMA-ES, rank-based (μ, λ)-selection is used. That is, the μ best of the λ offspring form the next parent population. After selection, the strategy parameters, both the matrix $\mathbf{C}^{(g)}$ and the step size $\sigma^{(g)}$, are updated. The update of the matrix $\mathbf{C}^{(g)}$ is governed by

$$\begin{aligned} \mathbf{p}^{(g+1)} &= (1 - c_c) \cdot \mathbf{p}^{(g)} \\ &\quad + \sqrt{c_c(2 - c_c)} \frac{\sqrt{\mu_{\mathrm{eff}}}}{\sigma^{(g)}} \left(\langle \mathbf{x} \rangle_{\mathbf{v}}^{(g+1)} - \langle \mathbf{x} \rangle_{\mathbf{v}}^{(g)} \right) \quad , \end{aligned} \tag{A.5}$$

$$\begin{aligned} \mathbf{C}^{(g+1)} &= (1 - c_{\mathrm{cov}}) \cdot \mathbf{C}^{(g)} \\ &\quad + c_{\mathrm{cov}} \cdot \left(\frac{1}{\mu_{\mathrm{cov}}} \mathbf{U}_1^{(g+1)} + \left(1 - \frac{1}{\mu_{\mathrm{cov}}} \right) \mathbf{U}_\mu^{(g+1)} \right) \quad , \end{aligned} \tag{A.6}$$

where

$$\mathbf{U}_1^{(g+1)} = \mathbf{p}^{(g+1)} \left(\mathbf{p}^{(g+1)} \right)^{\mathrm{T}} \quad , \tag{A.7}$$

$$\begin{aligned} \mathbf{U}_\mu^{(g+1)} &= \sum_{i=1}^{\mu} \frac{v_i}{\sigma^{(g)2}} \\ &\quad \cdot \left(\mathbf{x}_{i:\lambda}^{(g+1)} - \langle \mathbf{x} \rangle_{\mathbf{v}}^{(g)} \right) \left(\mathbf{x}_{i:\lambda}^{(g+1)} - \langle \mathbf{x} \rangle_{\mathbf{v}}^{(g)} \right)^{\mathrm{T}} \quad . \end{aligned} \tag{A.8}$$

The vector $\mathbf{p}^{(g+1)} \in \mathbb{R}^n$ is the evolution path, a weighted sum of the centers of the population over the generations starting from $\mathbf{p}^{(0)} = \mathbf{0}$. The factor $\sqrt{\mu_{\mathrm{eff}}}$ compensates for the loss of variance due to computing the center of mass. The parameter

$c_c \in]0,1]$ controls the time horizon of the adaptation of $\mathbf{p}$. The constant $\sqrt{c_c(2-c_c)}$ normalizes the variance of $\mathbf{p}$ viewed as a random variable, because $1^2 = (1-c_c)^2 + (\sqrt{c_c(2-c_c)})^2$. The parameter $c_{\mathrm{cov}} \in [0,1[$ controls the update of $\mathbf{C}^{(g)}$. The vector $\mathbf{p}$ does not only represent the last (adaptive) step of the parent population, but a time average over all previous adaptive steps. The influence of previous steps decays exponentially, where the decay rate is controlled by c_{cov}.

The update rule (A.6) for the covariance matrix shifts $\mathbf{C}^{(g)}$ towards the $n \times n$ matrices $\mathbf{U}_1^{(g+1)}$ and $\mathbf{U}_\mu^{(g+1)}$. The relative importance of the two matrices is controlled by the parameter μ_{cov}. The matrix $\mathbf{U}_1^{(g+1)}$ has rank one, and the normal distribution with zero mean and covariance matrix $\mathbf{U}_1^{(g+1)}$ is the normal distribution with zero mean that makes the mutation $\mathbf{p}^{(g+1)}$ most likely. That is, the shift towards $\mathbf{U}_1^{(g+1)}$ exploits the information gathered in the evolution path over several generations. The matrix $\mathbf{U}_\mu^{(g+1)}$ results from the weighted sum over all selected offspring. It has (almost surely) rank $\min(\mu, n)$. That is, the shift towards $\mathbf{U}_1^{(g+1)}$ exploits the information from selection in the last generation. This is particularly useful in case of large populations.

The adaptation of the global step-size parameter σ is done separately on a shorter timescale (a single parameter can be estimated based on less samples than the complete covariance matrix). We keep track of a second evolution path $\mathbf{p}_\sigma$ without the scaling by $\mathbf{D}$:

$$\begin{aligned}\mathbf{p}_\sigma^{(g+1)} &= (1-c_\sigma)\cdot\mathbf{p}_\sigma^{(g)} \\ &\quad + \sqrt{c_\sigma(2-c_\sigma)}\cdot\mathbf{B}^{(g)}{\mathbf{D}^{(g)}}^{-1}{\mathbf{B}^{(g)}}^{\mathrm{T}} \\ &\quad \cdot\frac{\sqrt{\mu_{\mathrm{eff}}}}{\sigma^{(g)}}\left(\langle\mathbf{x}\rangle_{\mathbf{v}}^{(g+1)} - \langle\mathbf{x}\rangle_{\mathbf{v}}^{(g)}\right) \ , \qquad \text{(A.9)}\end{aligned}$$

$$\sigma^{(g+1)} = \sigma^{(g)}\cdot\exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^{(g+1)}\| - \hat{\chi}_n}{\hat{\chi}_n}\right)\right) \ , \qquad \text{(A.10)}$$

where $\hat{\chi}_n$ is the expected length of a n-dimensional, normally distributed random vector with covariance matrix $\mathbf{I}$. It is approximated by $\sqrt{n}(1-\frac{1}{4n}+\frac{1}{21n^2})$. The damping parameter d_σ decouples the adaptation rate from the strength of the variation. The parameter $c_\sigma \in]0,1]$ controls the update of $\mathbf{p}_\sigma$.

If there were no selection (i.e., if the new parents were selected from the offspring uniformly at random), the evolution path $\mathbf{p}_\sigma$ would be a weighted sum of independently normally distributed random variables starting form $\mathbf{p}_\sigma^{(0)} = \mathbf{0}$. Because of the normalization, its expected length would tend to $\hat{\chi}_n$ for growing g. Hence, the update rule basically increases the global step size if the evolution path $\mathbf{p}_\sigma$ is larger than expected under uniform random selection and decreases the step size in the opposite case. If the path $\mathbf{p}_\sigma$ is shorter than $\hat{\chi}_n$ then the steps that led to selected individuals canceled out each other more strongly than expected (i.e., they tended to be anticorrelated) or the selected steps were smaller than expected. Thus, the step size σ should be decreased. Many successive steps in the same direction, which do not cancel out in $\mathbf{p}_\sigma$ and could have been realized by a single long step, lead to an evolution path $\mathbf{p}_\sigma$ that is larger than expected and the step size should be increased.

The standard parameters[20] of the CMA-ES using the rank-μ-update and superlinear weighted recombination are

$$v_i = \ln(\mu+1) - \ln i \ , \qquad \text{(A.11)}$$

$$c_\sigma = \frac{10}{n+20} \ , \qquad \text{(A.12)}$$

$$d_\sigma = \max\left(1, \frac{3\mu_{\mathrm{eff}}}{n+10})\right) + c_\sigma \ , \qquad \text{(A.13)}$$

$$c_c = \frac{4}{4+n} \ , \qquad \text{(A.14)}$$

$$\mu_{\mathrm{cov}} = \mu_{\mathrm{eff}} \ , \qquad \text{(A.15)}$$

$$\begin{aligned}c_{\mathrm{cov}} &= \frac{1}{\mu_{\mathrm{cov}}}\frac{2}{(n+\sqrt{2})^2} \\ &\quad + \left(1-\frac{1}{\mu_{\mathrm{cov}}}\right)\min\left(1, \frac{2\mu_{\mathrm{eff}}-1}{(n+2)^2+\mu_{\mathrm{eff}}}\right) \ . \qquad \text{(A.16)}\end{aligned}$$

The population sizes are chosen according to the following heuristics[20]

$$\lambda = 4 + \lfloor 3\ln n\rfloor \ , \qquad \text{(A.17)}$$

$$\mu = \lfloor \lambda/2 \rfloor \ . \qquad \text{(A.18)}$$

These default values are used throughout this study.

Table 1. The 3-mers of major importance for classification

oligomer	ATG	AGG	GGA	GAG	GGC	GGT	ACT	TCG	TGG	GAT
weight	2.490	2.166	2.004	1.888	1.646	1.455	1.389	1.339	1.327	1.295

Table 2. Performance of the 64 weight oligo kernel SVM, the locality improved kernel SVM and the Markov chain model.

model	accuracy	specificity	sensitivity	correlation
SVM, 64 weight oligo kernel	92.92 %	95.73 %	88.13 %	84.13 %
25% quantile	92.37 %	95.06 %	86.79 %	82.65 %
median	92.90 %	95.64 %	87.84 %	84.07 %
75% quantile	93.46 %	96.47 %	89.19 %	85.43 %
SVM, 6-combined oligo kernel	93.28 %	95.73 %	89.12 %	85.04 %
25%-quantile	93.25 %	95.83 %	89.34 %	84.90 %
median	92.82 %	95.16 %	87.99 %	83.92 %
75%-quantile	93.90 %	96.20 %	90.69 %	86.67 %
SVM, locality improved kernel	92.54 %	95.15 %	88.10 %	83.47 %
25% quantile	92.02 %	94.55 %	87.09 %	82.50 %
median	92.53 %	95.03 %	88.14 %	83.46 %
75% quantile	93.03 %	95.85 %	89.19 %	84.61 %
Markov chain model	91.51 %	92.01 %	90.64 %	82.69 %
25% quantile	90.86 %	90.96 %	89.49 %	81.45 %
median	91.42 %	91.88 %	90.69 %	82.91 %
75% quantile	91.94 %	93.01 %	91.89 %	83.68 %

References

1. H.-G. Beyer and H.-P. Schwefel. Evolution strategies: A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.
2. F. R. Blattner et al. The complete genome sequence of Escherichia coli K-12. *Science*, 277:1453–1462, 1997.
3. O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1):131–159, 2002.
4. K.-M. Chung, W.-C. Kao, C.-L. Sun, and C.-J. Lin. Radius margin bounds for support vector machines with RBF kernel. *Neural Computation*, 15(11):2643–2681, 2003.
5. C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
6. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.
7. N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2001. MIT Press.
8. S. Degroeve, B. D. Beats, Y. V. de Peer, and P. Rouzé. Feature subset selection for splice site prediction. *Bioinformatics*, 18 Suppl 2:75–83, 2002.
9. D. R. Eads et al. Genetic algorithms and support vector machines for time series classification. In B. Bosacchi, D. B. Fogel, and J. C. Bezdek, editors, *Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation V.*, volume 4787 of *Proceedings of the SPIE*, pages 74–85, 2002.
10. T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000.
11. R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using the second order information for training support vector machines. *Journal of Machine Learning Research*, 6:1889–1918, 2005.
12. C. M. Fonseca and P. J. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature (PPSN IV)*, pages 584–593, 1996.
13. F. Friedrichs and C. Igel. Evolutionary tuning of multiple SVM parameters. *Neurocomputing*, 64(C):107–117, 2005.
14. H. Fröhlich, O. Chapelle, and B. Schölkopf. Feature selection for support vector machines using genetic algorithms. *International Journal on Artificial Intelligence Tools*, 13(4):791–800, 2004.
15. T. Glasmachers and C. Igel. Gradient-based adaptation of general gaussian kernels. *Neural Computation*, 17(10):2099–2105, 2005.
16. T. Glasmachers and C. Igel. Maximum-gain working set selection for support vector machines. *Journal of Machine Learning Research*, 7:1437–1466, 2006.
17. C. Gold and P. Sollich. Model selection for support vector machine classification. *Neurocomputing*, 55(1-2):221–249, 2003.
18. C. O. Gualerzi and C. L. Pon. Initiation of mRNA translation in procaryotes. *Biochemistry*, 29(25):5881–5889, 1990.
19. N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao, E. Burke, J. A. Lozano, J. Smith, J. J. Merelo-Guervós, J. A. Bullinaria, J. Rowe, P. T. A. Kabán, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature (PPSN VIII)*, volume 3242 of *LNCS*, pages 282–291. Springer-Verlag, 2004.
20. N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao et al., editors, *Parallel Problem Solving from Nature (PPSN VIII)*, volume 3242 of *LNCS*. Springer-Verlag, 2004.
21. N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.
22. A. G. Hatzigeorgiou. Translation initiation start prediction in human cDNAs with high accuracy. *Bioinformatics*, 18:343–350, 2002.
23. C. Igel. Evolutionary kernel learning. In C. Sammut, editor, *Encyclopedia of Machine Learning*. Springer-Verlag. To appear Novemeber 2007.
24. C. Igel. Multi-objective model selection for support vector machines. In C. A. C. Coello, E. Zitzler, and A. H. Aguirre, editors, *Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, volume 3410 of *LNAI*, pages 534–546. Springer-Verlag, 2005.
25. C. Igel, T. Glasmachers, B. Mersch, N. Pfeifer, and P. Meinicke. Gradient-based optimization of kernel-target alignment for sequence kernels applied to bacterial gene start detection. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(2):216–226, 2007.
26. C. Igel, S. Wiegand, and F. Friedrichs. Evolutionary optimization of neural systems: The use of strategy adaptation. In M. G. de Bruin, D. H. Mache, and J. Szabados, editors, *Trends and Applications in Constructive Approximation*, volume 151 of *International Series of Numerical Mathematics*, pages 103–123. Birkhäuser Verlag, 2005.
27. K. Jong, E. Marchiori, and A. van der Vaart. Analysis of proteomic pattern data for cancer detection. In G. R. Raidl et al., editors, *Applications of Evolutionary Computing*, number 3005 in LNCS, pages 41–51. Springer-Verlag, 2004.
28. S. S. Keerthi. Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks*, 13(5):1225–1229, 2002.
29. M. Kozak. Initiation of translation in prokaryotes and

eukaryotes. *Gene*, 234:187–208, 1999.

30. A. Krogh. An introduction to hidden Markov models for biological sequences. In S. L. Salzberg, D. B. Searls, and S. Kasif, editors, *Computational Methods in Molecular Biology*, chapter 4, pages 45–63. Elsevier, 1998.
31. C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for SVM protein classification. In R. B. Altman, A. K. Dunker, L. Hunter, H. Lauerdale, and T. E. Klein, editors, *Proceedings of the Pacific Symposium on Biocomputing*, pages 564–575. World Scientific, 2002.
32. H. Li and T. Jiang. A class of edit kernels for SVMs to predict translation initiation sites in eukaryotic mRNAs. *Journal of Computational Biology*, 12(6):702–718, 2005.
33. B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) – Protein Structure*, 405(2):442–451, 1975.
34. P. Meinicke, M. Tech, B. Morgenstern, and R. Merkl. Oligo kernels for datamining on biological sequences: A case study on prokaryotic translation initiation sites. *BMC Bioinformatics*, 5:169, 2004.
35. B. Mersch, T. Glasmachers, P. Meinicke, and C. Igel. Evolutionary optimization of sequence kernels for detection of bacterial gene starts. In Kollias et al., editors, *International Conference on Artificial Neural Networks (ICANN 2006)*, number 4132 in LNCS, pages 827–836. Springer-Verlag, 2006.
36. M. T. Miller, A. K. Jerebko, J. D. Malley, and R. M. Summers. Feature selection for computer-aided polyp detection using genetic algorithms. In A. V. Clough and A. A. Amini, editors, *Medical Imaging 2003: Physiology and Function: Methods, Systems, and Applications*, volume 5031 of *Proceedings of the SPIE*, pages 102–110, 2003.
37. S. Pang and N. Kasabov. Inductive vs. transductive inference, global vs. local models: SVM, TSVM, and SVMT for gene expression classification problems. In *International Joint Conference on Neual Networks (IJCNN)*, volume 2, pages 1197–1202. IEEE Press, 2004.
38. A. G. Pedersen and H. Nielsen. Neural network prediction of translation initiation sites in eukaryotes: Perspectives for est and genome analysis. In *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology*, pages 226–233. AAAI Press, 1997.
39. J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 12, pages 185–208. MIT Press, 1999.
40. J. C. Rajapakse and L. S. Ho. Markov encoding for detecting signals in genomic sequences. *ACM Transactions on Computational Biology and Bioinformatics*, 2(2):131–142, 2006.
41. K. E. Rudd. Ecogene: a genome sequence database for *Escherichia coli* K-12. *Nucleic Acids Research*, 28:60–64, 2000.
42. T. P. Runarsson and S. Sigurdsson. Asynchronous parallel evolutionary model selection for support vector machines. *Neural Information Processing – Letters and Reviews*, 3(3):59–68, 2004.
43. S. L. Salzberg. A method for identifying splice sites and translational start sites in eukaryotic mRNA. *Computer Applications in the Biosciences*, 13:365–376, 1997.
44. B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.
45. B. Schölkopf, K. Tsuda, and J.-P. Vert, editors. *Kernel Methods in Computational Biology*. Computational Molecular Biology. MIT Press, 2004.
46. S. Y. M. Shi, P. N. Suganthan, and K. Deb. Multiclass protein fold recognition using multi-objective evolutionary algorithms. In *IEEE Symposium on Computational Intelligence in Bioinformatics an d Computational Biology*, pages 61–66, 2004.
47. J. Shine and L. Dalgarno. The 3'-terminal sequence of *Escherichia coli* 16S ribosomal RNA: Complementarity to nonsense triplets and ribosome binding sites. *PNAS*, 71(4):1342–1346, 1974.
48. M. Tech and P. Meinicke. An unsupervised classification scheme for improving predictions of prokaryotic TIS. *BMC Bioinformatics*, 7:121, 2006.
49. H. Xiong, M. N. S. Swamy, and M. O. Ahmad. Optimizing the kernel in the empirical feature space. *IEEE Transactions on Neural Networks*, 16(2):460–474, 2005.
50. A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K. R. Müller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, 2000.