

Support Vector Machines for Protein Fold Class Prediction

FLORIAN MARKOWETZ^{1*}, LUTZ EDLER² and MARTIN VINGRON¹

¹ Department of Computational Molecular Biology, Max-Planck-Institute for Molecular Genetics, Ihnestrasse 63-73, D-14195 Berlin, Germany

² Central Unit Biostatistics C060, German Cancer Research Center (DKFZ), Im Neuenheimer Feld 280, D-69120 Heidelberg, Germany

Abstract: Knowledge of the three-dimensional structure of a protein is essential for describing and understanding its function. Today, a large number of known protein sequences faces a small number of identified structures. Thus, the need arises to predict structure from sequence without using time-consuming experimental identification. In this paper the performance of *Support Vector Machines* (SVMs) is compared to Neural Networks and to standard statistical classification methods as Discriminant Analysis and Nearest Neighbor Classification. We show that SVMs can beat the competing methods on a dataset of 268 protein sequences to be classified into a set of 42 fold classes. We discuss misclassification with respect to biological function and similarity. In a second step we examine the performance of SVMs if the embedding is varied from frequencies of single amino acids to frequencies of triplets of amino acids. This work shows that SVMs provide a promising alternative to standard statistical classification and prediction methods in functional genomics.

Keywords: Protein Fold Class Prediction; Support Vector Machines; Statistical Classification Methods; Neural Networks; Confusion Matrix.

1. Introduction

Even the complete knowledge of the genomic sequence is only the first step towards an understanding of how an organism develops and functions. The next key landmark is an overview of the characteristics and activities of the proteins encoded in the genes. The function of a protein largely depends on its structure which itself depends on the protein sequence. Therefore understanding and predicting how protein sequence information translates into three-dimensional protein structure and folding has become one of the most challenging open questions of current molecular biology. Only a small percentage of proteins for which the sequence is known could be explored for their three-dimensional structure by physical methods and there is a large gap between the number of known protein sequences and the number of identified structures (EDLER and GRASSMANN, 1999). Methods of statistical classification can help to bridge this gap. Unlike methods that try to find the 3D structure of a new protein sequence by aligning it to a protein with given structure (e.g., by homology modeling for closely related sequences or threading methods for more distantly related proteins), discriminative methods of machine learning and statistics have been used to recognize the fold class of a protein from its amino acid sequence (GRASSMANN, RECZKO, SUHAI, and EDLER, 1999; EDLER, GRASSMANN and SUHAI, 2001; CAI, LIU, XU, and ZHOU, 2001; DING and DUBCHAK, 2001; LESLIE, ESKIN, and STAFFORD NOBLE, 2002). These previous investigations either concentrated on statistical or on machine learning methods and did not compare both approaches.

*Corresponding author: florian.markowetz@molgen.mpg.de

In this paper, we compare Support Vector Machines to various other methods from machine learning and statistics (section 3.1). For comparison to previous results, we embed the proteins by their dipeptide-frequencies. We investigate the biological background of misclassifications in section 3.2. In a further step we address the question, how the choice of an embedding influences the performance of SVM. Therefore we try SVM with various kernels on different embeddings in section 3.3.

During our investigation of SVMs we also realized that this machine learning method is not well known in the biometrical community although it has a strong theoretical foundation as an optimization procedure applicable to high dimensional data. Therefore we found it timely and justified to draw more attention to this method and to provide a short outline of the basic elements of SVMs in an appendix.

2. Methods and Materials

2.1 Support Vector Machines

SVMs were introduced by Vladimir VAPNIK (1995, 1998). In the last two years very good introductory textbooks were published (CRISTIANINI and SHAW-TAYLOR, 2000; SCHÖLKOPF and SMOLA, 2002). We will give a short overview of the main steps in the construction of SVMs in the appendix, focussing on the task to predict a class label $y \in \{-1, +1\}$ from a pattern vector $x \in \mathbb{R}^d$. As a software we used the MATLAB Support Vector Machine Toolbox (GUNN, 1997), which can be obtained from <http://www.kernel-machines.org>.

Multiple classes. There are many ways to apply SVMs to $n > 2$ classes. Most commonly used is a *one-against-all* approach, also called *winner-takes-it-all*. Every class is separated by a SVM from the pooled datapoints of all other $n - 1$ classes. To test a new point, one calculates the distance to all n hyperplanes and assigns it to the class for which it achieves maximum distance. Because of the high computational cost in our case of 42 classes we decided against using *all-versus-all* or *unique one-versus-all* methods, even if DING and DUBCHAK (2001) show evidence that the generalisation performance can be enhanced by these methods.

Imbalanced datasets. SVM should be adapted to imbalanced datasets, where points of one class are much more numerous than points of the other class (KARAKOULAS and SHAW-TAYLOR, 1999; BROWN et. al., 1997). In this situation the cost of misclassifying a point from the smaller class should be heavier than the cost for errors on the large class. The basic idea is to introduce different error weights C^+ and C^- for the positive and the negative class (see Equation (13) in the appendix), which results in a bias for larger multipliers α_i of the 'critical' class. This induces a decision boundary which is more distant from the smaller class than from the other. As a heuristic to automatically adjust the error weight we chose $C^+ = 1/m^+$ and $C^- = 1/m^-$, where m^+ and m^- are the cardinalities of the two classes.

2.2 The data set of protein sequences

For training and testing we use the data set of GRASSMANN et. al. (1999), originating from the *Database for Expected Fold-classes* (DEF) of RECZKO and BOHR, (1994). In the DEF a sequence of amino acids is assigned a specific fold-class. According to topological similarities of the backbone a set of 38 fold classes was defined (PASCARELLA and ARGOS, 1992) and later

enlarged to 42 classes of tertiary structure, which are called 42-CAT (RECZKO and BOHR, 1994). GRASSMANN et al. (1999) also study the classification of protein sequences into the four so called super secondary classes characterized by the presence and absence of α -helices and β -sheets in the three-dimensional structure of the protein. The SVM methods developed above are applicable to this classification task in the very same way (MARKOWETZ, 2001). In this paper we will report only the results obtained for the much harder classification into the 42 fold classes, where most of the standard statistical methods failed.

We chose the same experimental setting as GRASSMANN et al. (1999). The data set consists of 268 sequences divided into a training set of 143 and a test set of 125 sequences. This division was done randomly, balanced with respect to the prior probabilities of the 42 classes. Typical for protein sequence data, there is no uniform distribution of the classes. A list of all the sequences used can be found at <http://www.dkfz.de/biostatistics/protein/gsm97.html>. This webpage also contains a summary of the performance of all statistical classification methods previously applied to this dataset.

2.3 Data representation

Formally, a protein is a sequence (s_1, \dots, s_q) , where each of the s_i stands for one of the twenty amino acids. To use SVM, these sequences have to be embedded into a more regularly defined *feature space*. A simple embedding is achieved by the *relative frequencies of k -tuples of amino acids*. This results in a feature space of dimension 20^k . CAI et al. (2001) and DING and DUBCHAK (2001) use $k = 1$, i. e. the sequences are embedded into \mathbb{R}^{20} . Both report that the amino acid composition is a very effective way to describe protein sequences. But of course the way a protein folds does not only depend on single amino acids, but on the relationships between neighbors in the sequence (BRANDEN and TOOZE, 1991). Therefore we investigate the performance of classification methods for bigger values of k .

This is similar to the work of LESLIE, ESKIN and STAFFORD NOBLE (2002). They design a kernel function that compares the number of k -length contiguous subsequences in two protein sequences. In our terminology: they embed the protein sequences by the frequencies of k -tuples of amino acids into a 20^k -dimensional vector space and then use a dot product in this space. Our approach differs in two important aspects: first, we use *relative* frequencies to take into account the unequal lengths of the sequences, and second, we not only use dot products on the embedded proteins but also higher degree polynomials and radial basis function kernels.

3. Results

3.1 Comparison of SVM to other classification methods

In (GRASSMANN et al., 1999) several statistical methods are used on the training data embedded by dipeptide frequencies: Feed Forward Neural Networks (NN), Additive Model BRUTO, Projection Pursuit Regression (PPR), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Flexible Discriminant Analysis (FDA) and the k -Nearest-Neighbor Rule (kNN). We compare these methods to SVM in three ways: the error on the training set, the error on the test set and the ten-fold cross-validation error (CV(10)). Classification methods usually break down in very high dimensions ("*curse of dimensionality*"). Therefore, GRASSMANN et al. (1999) preprocessed the data by Principal Component Analysis. Please note, that

Error	kNN	LDA	QDA	NN(0)	NN(5)	NN(9)	PPR	rbf	poly1	poly2	poly3
train	0	0	0	9.8	11.2	2.1	>50	0	0	4.2	1.4
test	33.6	34.4	38.2	39.2	36.8	28.8	>50	23.2	28.8	32	32.8
CV	34.0	30.2	38.4	36.9	38.8	32.5	>50	30.2	29.1	35	34.2

Table 1: Results of classification methods (embedding by dipeptide frequencies). k-Nearest-Neighbor (kNN), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Neural Networks (NN) and Projection Pursuit Regression (PPR) are compared to Support Vector Machines with radial basis kernel (rbf: σ chosen as the mean of the inter-class distances) and polynomial kernel of degree 1,2,3 (poly1, poly2, poly3). The best results on the test set and in cross-validation are emphasized both for the the competing methods and for SVMs.

we do not need to use any dimension reduction in SVM.

The left half of Table 1 shows the results of the classification methods used by GRASSMANN et. al. (1999). The best results are 28.8% test error by a Neural Network with 9 hidden layers and 30.2% cross-validation error by Linear Discriminant Analysis. The right half of Table 1 summarizes the results of support vector classification. The width of the rbf kernel is chosen as the median of the inter-class distances (JAAKKOLA, DIEKHANS and HAUSSLER, 2000). The first observations is that the linear poly1-SVM shows excellent performance. With 28.8% test error it is as good as NN(9) and with 29.1% CV error it is even better than LDA. The rbf-SVM shows the best error rate of 23.2% on the test set, but this success does not show in cross-validation, where rbf-SVM has an error of 30.2%. The poly2-SVM and poly3-SVM exhibit only slight differences on the test set (32% and 32.8%). For the polynomial SVM there are no great differences between the performance on the test set and the behaviour in 10-fold cross validation. In this case the test error seems to be a reliable estimate of generalization ability. rbf-SVM shows a large gap between test error and CV(10). The low test error of 23.2% has to be attributed to the choice of the test set and not to the generalization ability of the Support Vector Machine. The fluctuations in the training error in the case of polynomial SVMs are due to the fact, that the use of kernel mappings in the construction of SVM (see the Appendix for details) results in non-nested modeling.

3.2 Confusion matrices and biological interpretation

In protein fold classification one is usually not only interested in the number of misclassifications but also in the pattern of classification errors. Are all errors uniformly distributed among the 42 classes, or are some classes more likely to be falsely predicted than others? This question is answered by the construction of a so called *Confusion Matrix*. This is a matrix $M = (m_{ij})$ with entries $m_{ij} = \#\{x : x \text{ is predicted as class } i \text{ and truly belongs to class } j\}$.

We built confusion matrices for poly1-SVM and rbf-SVM based on the output of CV(10). The most striking feature in both matrices is the high frequency of misclassifications into class 22 and class 29. As an example we show the confusion matrix of poly1-SVM in Figure 1.

To gain insight into the attractiveness of classes 22 and 29 for misclassifications, we investigate the biological properties of the according proteins. Why are so many sequences falsely predicted as belonging to class 22 or 29? Why are the sequences of class 29 very well recognized by Support Vector Machines, while more than half of the sequences in class 22 are misclassified themselves?

Class 22: TIM Barrels. The barrel structure is distinguished by a core of eight parallel

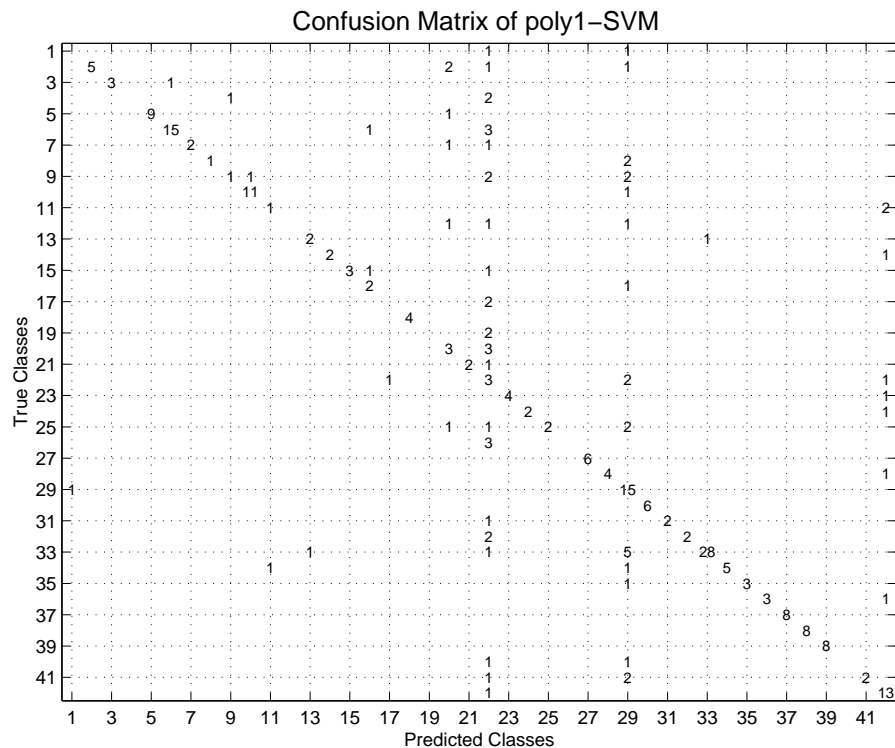


Figure 1: Confusion Matrix of poly1-SVM. Notice the numerous misclassifications in class 22 (31 false positives, 4 false negatives, 3 true positives) and class 29 (23 false positives, 1 false negative, 15 true positives).

β -strands arranged closely together, like staves, into a barrel (SCHEERLINCK et. al., 1992). The β -strands are connected by α -helices, all lying on the outside of this barrel. This structure is called a *TIM barrel*, because it was first observed in the enzyme *triosephosphate isomerase*. The eight-stranded α/β -barrel structure is one of the largest and most regular of all protein domain structures. It has been found in many different proteins with completely different amino acid sequences and different functions. The sequence similiarity is generally poor among these proteins. This suggests that the eightfold α/β -barrel could be a rather nonspecific stable motif that is quite tolerant to sequence variations and onto which different functionalities can be designed. The low sequence similiarity between barrel proteins also gives a reason for the observed misclassifications. Class 22 is very heterogeneous, containing an multitude of diverse sequences. This explains why its elements are often misclassified. In addition, the diversity of barrel sequences explains the high number of false positives: the hyperplanes at the boundary of class 22 are chosen to embrace much volume and this often causes sequences from other classes to fall into this space where they do not belong.

Class 29: Virus Capsid Proteins. Viruses are maybe the simplest form of life. They are constructed from two basic components: a *genomic nucleic acid* molecule, which is surrounded by a protein shell, a *capsid* (and sometimes by an additional *envelope*) (BRANDEN and TOOZE, 1991). No nucleic acid can code for a single protein molecule that is big enough to enclose it. Therefore, the protein shell of a virus is built up from many copies of one or a few polypeptide

chains. Consider as an example the shell of *picorna viruses*. It is a spherical construction of 60 copies each of four polypeptide chains, called VP1 to VP4. The first three chains VP1, VP2 and VP3 are the main building blocks, whereas VP4 is much smaller and can be considered as a continuation of VP2 that has become detached. The three subunits VP1, VP2 and VP3 exhibit no significant amino acid sequence similarity. So here again we encounter a class of proteins belonging to the same folding class but with great differences on the sequence level. As in the case of *TIM barrels*, this explains the extent and diversity of the class, which in turn results in many false positives. In contrast, the same subunit from different *picorna viruses* show a high rate of sequence identity, for example 30% for VP2 from *Mengo virus* and *rhinovirus*. Roughly speaking, the whole class is divided into three clusters according to the three protein subunits. This explains the high number of true positives: every sequence in the test set has nearly related sequences in the training set.

3.3 Influence of neighborhood relations between amino acids

In section 3.1 we fixed the embedding of the protein sequences to dipeptides ($k = 2$) for the comparison between the different classification methods and found that SVM achieved a new record performance on the data set. Our next question is this: how does the performance of SVM depend on the embedding of the data? In other words: How big is the influence of neighborhood relations between amino acids on the performance of SVM? Since we have no data from the competing methods, we can only compare the behaviour of SVM for different choices of the kernel function. Table 2 shows the CV(10) error for $k = 1, 2, 3$. We did not try $k = 4$ because this would result in separating 268 data points in a space of dimension $20^4 = 160\,000$, which we do not think to be reasonable. Also, LESLIE et. al. (2002) report performance decreasing for $k = 4$.

CV(10)	rbf	poly1	poly2	poly3
$k = 1$	29.5	46.6	52.2	59.3
$k = 2$	30.2	29.1	35	34.2
$k = 3$	22.4	23.5	22.8	22.8

Table 2: Comparison of 10-fold Cross Validation error for different embeddings of the protein sequences: amino acid frequencies (1-tuple, \mathbb{R}^{20}), dipeptide frequencies (2-tuple, \mathbb{R}^{400}) and tripeptide frequencies (3-tuple, \mathbb{R}^{8000}).

The combination of a SVM with radial basis kernel and an embedding by 3-tuples achieves almost 8% less cross validation error than LDA with an embedding by dipeptides (as shown in Table 1). If the size of the tuples increases from 1 to 3, more information is contained in the embedding and we expect the error rates to improve. In all polynomial kernel SVM this can clearly be seen, while the rbf-SVM has almost the same CV(10) error for data embedded by single or pairs of amino acids. Generally, the gap between $k = 1$ and the $k = 2$ is bigger than the gap between $k = 2$ and $k = 3$. We conclude that SVM mainly learn the amino acid frequencies, including information about the direct neighborhood improves the results, but further information about more distant neighborhood relations has less impact on the classification. Whether this is a biological property of protein folding or a property of the predictive power of SVM remains open to further research.

4. Discussion

Excellent performance in very high-dimensional spaces. Our results show that SVMs are able to achieve high performance in spaces where many other methods break down. The competing methods could achieve their best results only after reducing the dimensionality by Principal Component Analysis, while SVMs increase generalization performance even in an 8000 dimensional input space.

SVM outperform classical methods. The ability of SVMs to handle high dimensional data leads to classification results surpassing those of the competitors. Even when the sequences were embedded by their dipeptide-frequencies, the error rates were better than those achieved with the classical methods, but the real power of SVMs shows in the embedding by the frequency of 3-tuples of amino acids. While classical methods break down in these high dimensions, SVMs reduce the error in cross validation to roughly 22 – 23%.

Our experiments confirm the results of (CAI et. al., 2001) and (DING and DUBCHAK, 2001) who also report higher accuracy of SVMs as compared to Neural Networks. DING and DUBCHAK (2001) obtain their results on 27 protein folding classes, whereas CAI et. al. (2001) only compare four classes. Because of these different experimental setups, we see the need to identify benchmark datasets for statistical protein fold class prediction to achieve a more thorough comparison of prediction methods.

Data representation. Our results indicate that incorporating neighborhood relations into the embedding of protein sequences ($k > 1$) is more advantageous than just using the amino acid composition ($k = 1$). For large values of k efficient ways to calculate the kernel function have to be found and one always has to bear in mind the immense dimension of the input space for $k > 3$. Further research is needed to evaluate more complex representations of the protein sequences, e. g. by allowing gaps in the embedding. Protein folding is a biological problem, thus every choice of an embedding has to be backed up by sound biological reasoning. We are convinced that the combination of state-of-the-art classification methods with a biologically sensible data representation will help to answer important questions in functional genomics.

Acknowledgement

Part of this work is based on a Diploma Thesis (MARKOWETZ, 2001) at the Institute of Applied Mathematics, University of Heidelberg, and the Department of Theoretical Bioinformatics at the German Cancer Research Center (DKFZ) Heidelberg. The authors thank Enno Mammen (Institute of Applied Mathematics, Heidelberg) for the good cooperation. We are also grateful to two anonymous referees who improved the paper considerably by their questions and comments.

Appendix: Construction of Support Vector Machines

In the following we give a short and comprehensive outline of the basic theory and construction of SVMs. For details we refer to the textbooks cited in section 2.1. In the last few years SVMs proved excellent performance in many real-world applications such as analysis of microarray data, text categorisation, hand-written character recognition, image classification or biological sequence analysis. An almost complete listing of all applications can be found in the publications section of www.kernel-machines.org.

SVMs are a method of *supervised learning*. The statistical task is to predict a class label $y \in \{-1, +1\}$ from the information stored in pattern vectors $x \in \mathbb{R}^d$ after building the classifier on a training set with known labels. SVMs combine two concepts: Optimal margin hyperplanes and kernel mappings.

Optimal Margin Hyperplanes

Consider a set of training examples

$$\mathcal{X} = \{(x_1, y_1), \dots, (x_l, y_l) : x_i \in \mathbb{R}^d, y_i \in \{-1, +1\}, i = 1, \dots, l\}, \quad (1)$$

where the x_i are real d -dimensional pattern vectors and the y_i are dichotomous labels. The set (1) is called *separable by the hyperplane* $H = \{x \in \mathbb{R}^d \mid \langle w, x \rangle + b = 0, w \in \mathbb{R}^d, b \in \mathbb{R}\}$ if there exist both a unit vector w ($\|w\| = 1$) and a constant b such that the inequality

$$y_i(\langle w, x_i \rangle + b) - 1 \geq 0 \quad i = 1, \dots, l. \quad (2)$$

holds, where $\langle \cdot, \cdot \rangle$ denotes the inner product in \mathbb{R}^d . The hyperplane H defined by w and b is called a *separating hyperplane*. The margin $\gamma_i(w, b)$ of a training point x_i is defined as the distance between H and x_i :

$$\gamma_i(w, b) = y_i(\langle w, x_i \rangle + b). \quad (3)$$

The margin $\gamma_{\mathcal{S}}(w, b)$ of a set of vectors $\mathcal{S} = \{x_1, \dots, x_n\}$ is defined as the minimum distance from H to the vectors in \mathcal{S} :

$$\gamma_{\mathcal{S}}(w, b) = \min_{x_i \in \mathcal{S}} \gamma_i(w, b). \quad (4)$$

The *Optimal Margin Hyperplane* is the separating hyperplane achieving maximal margin of separation on the training set \mathcal{X} . It is the solution of the optimization problem

$$\begin{aligned} & \text{maximize} && \gamma_{\mathcal{X}}(w, b) \\ & \text{subject to} && \gamma_{\mathcal{X}}(w, b) > 0 \quad \text{and} \quad \|w\|^2 = 1. \end{aligned} \quad (5)$$

Equivalently (SCHÖLKOPF and SMOLA, 2002, p.196) for $i = 1, \dots, l$

$$\begin{aligned} & \text{minimize} && \frac{1}{2} \|w\|^2 \\ & \text{subject to} && y_i(\langle w, x_i \rangle + b) - 1 \geq 0. \end{aligned} \quad (6)$$

This optimization problem can be solved by finding the saddle point of the primal Lagrangian

$$L_P(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^l \alpha_i [y_i(\langle w, x_i \rangle + b) - 1], \quad (7)$$

where $\alpha_i \geq 0$ are the Lagrange multipliers. The Lagrangian $L_P(w, b, \alpha)$ has to be minimized with respect to the primal variables w and b and maximized with respect to the dual variables α_i . The *dual problem* is to find multipliers α_i which solve the problem

$$\begin{aligned} \text{maximize} \quad L_D(\alpha) &= \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle, \\ \text{subject to} \quad \alpha_i &\geq 0 \quad \forall i, \quad \text{and} \quad \sum_{i=1}^l \alpha_i y_i = 0. \end{aligned} \quad (8)$$

The construction of the Optimal Margin Hyperplane amounts to maximizing L_D with respect to the α_i , subject to (2) and positivity of the α_i . Denote the solution by α_i^* , then we obtain

$$w^* = \sum_{i=1}^l \alpha_i^* y_i x_i. \quad (9)$$

The solution (w^*, b^*) of (6) fulfills the *Kuhn-Tucker complementarity condition*

$$\alpha_i (y_i (\langle w^*, x_i \rangle + b^*) - 1) = 0 \quad \forall i. \quad (10)$$

Notice that for a given training point x_i either the corresponding Lagrange multiplier α_i equals zero, or x_i lies on one of the *margin hyperplanes*

$$H_1 = \{x : \langle w^*, x \rangle + b^* = +1\} \quad \text{or} \quad H_2 = \{x : \langle w^*, x \rangle + b^* = -1\},$$

containing the training points with the minimal distance to the Optimal Margin Hyperplane. The vectors on H_1 or H_2 with $\alpha_i > 0$ are called *Support Vectors* (SV). The complementarity condition (10) is used to compute the offset b^* : choose any i for which $\alpha_i \neq 0$; then (2) becomes an equality from which b^* can be computed. The predicted label of a new test point x is the output of

$$f(x) = \text{sign}(\langle w^*, x \rangle + b^*) = \text{sign} \left(\sum_{i=1}^{\#SV} \alpha_i y_i \langle x_i^{SV}, x \rangle + b^* \right), \quad (11)$$

where $\text{sign}(\cdot)$ denotes the sign function with values in $\{+1, -1\}$. Next we show how a separating hyperplane can be adapted to the case of linear non-separable datasets. Therefore, the separability constraints (2) are relaxed by introducing misclassification penalties ξ_i ($i = 1, \dots, l$):

$$y_i (\langle w, x_i \rangle + b) - 1 + \xi_i \geq 0 \quad i = 1, \dots, l. \quad (12)$$

If $\xi_i \geq 1$ then x_i is misclassified; if $0 < \xi_i < 1$, then x_i is classified correctly, but lies inside the margin; and if $\xi_i = 0$, then x_i is classified correctly and lies outside the margin or on the margin boundary. $\sum_{i=1}^l \xi_i$ is an upper bound on the number of training errors. The minimization problem is changed to

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i^k \\ \text{subject to} \quad & y_i (\langle w, x_i \rangle + b) - 1 + \xi_i \geq 0 \quad \text{and} \quad \xi_i \geq 0, \end{aligned} \quad (13)$$

where the *error weight* C has to be chosen e. g. by cross-validation. For $k = 1$, the primal Lagrangian for this problem becomes

$$L_P(w, b, \xi, \alpha, \beta) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i - \sum_{i=1}^l \alpha_i (y_i (\langle w, x_i \rangle + b) - 1 + \xi_i) - \sum_{i=1}^l \beta_i \xi_i \quad (14)$$

with $\alpha_i \geq 0$ and $\beta_i \geq 0$. The β_i are the Lagrange multipliers introduced to enforce $\xi_i \geq 0$. This results in the following dual formulation:

$$\begin{aligned} \text{maximize} \quad & L_D(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C \quad \forall i \quad \text{and} \quad \sum_{i=1}^l \alpha_i y_i = 0. \end{aligned} \quad (15)$$

The only difference to (8) is that the Lagrange multipliers have an upper bound of C . Non-zero ξ_i only occur for $\alpha_i = C$. Pattern vectors x_i for which $0 < \alpha_i < C$ lie on one of the two margin hyperplanes H_1 or H_2 .

Kernel mappings

The optimal separating hyperplane obtained by solving the margin optimization problem (5) is a simple special case of a SVM. We will now show how this concept can be enhanced to more complex nonlinear classifiers. By the use of *kernel functions* the pattern vectors $x \in \mathbb{R}^d$ are mapped to a high dimensional space \mathcal{H} and separated there by a linear classifier. This results in a classifier nonlinear in \mathbb{R}^d .

Kernel functions. Given a mapping $\Phi : \mathbb{R}^d \rightarrow \mathcal{H}$ from input space \mathbb{R}^d to an (inner product) feature space \mathcal{H} , the function $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is called a kernel function, iff for all $x, z \in \mathbb{R}^d$

$$k(x, z) = \langle \Phi(x), \Phi(z) \rangle_{\mathcal{H}}. \quad (16)$$

The kernel function behaves like an inner product in \mathcal{H} , but can be evaluated as a function in \mathbb{R}^d . Choosing a kernel-function will implicitly define a mapping Φ . Most commonly used are

$$\begin{aligned} \text{polynomial kernels} \quad & k(x, z) = (\langle x, z \rangle + 1)^p \quad \text{and} \quad (17) \\ \text{radial basis function kernels} \quad & k(x, z) = \exp(-\|x - z\|^2 / 2\sigma^2), \quad (18) \end{aligned}$$

where the parameters $p \geq 1$ (degree of polynomial) and $\sigma > 0$ (width of rbf) are to be chosen by the user. All kernel functions have to fulfill *Mercer theorem* (see CRISTIANINI and SHAWE-TAYLOR, 2000, p33).

A *Support Vector Machine* then is the Optimal Margin Hyperplane combined with the kernel-induced mapping to a high dimensional feature space. This can be easily incorporated into (15) by substituting the inner product $\langle \cdot, \cdot \rangle$ with the kernel function $k(\cdot, \cdot)$: to construct a SVM we have to solve the optimization problem

$$\begin{aligned} \text{maximize} \quad & L_D(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C \quad \forall i \quad \text{and} \quad \sum_{i=1}^l \alpha_i y_i = 0. \end{aligned} \quad (19)$$

References

- [1] BRANDEN, C. and TOOZE, J., 1991: *Introduction to Protein Structure*, Garland Publishing, New York.
- [2] BROWN, M., NOBLE GRUNDY, W. , LIN, D. , CRISTIANINI, N. , SUGNET, C. , FUREY, T., ARES, M., and HAUSSLER, D., 1997: Knowledge-based analysis of microarray gene expression data by using support vector machines, *PNAS*, **97**, 262-267.
- [3] CAI, Y.-D., LIU, X.-J., XU, X.-B., and ZHOU, G.-P., 2001: Support Vector Machines for predicting protein structural class, *BMC Bioinformatics*, **2**.
- [4] CRISTIANINI, N. and SHAWE-TAYLOR, J., 2000: *An Introduction to Support Vector Machines*, Cambridge University Press, Cambridge, UK.
- [5] DING, C., and DUBCHAK, I., 2001: Multi-class protein fold recognition using support vector machines and neural networks, *Bioinformatics*, **17**, 349-358.
- [6] EDLER, L. and GRASSMANN, J., 1999: Protein Fold Class Prediction is a new field for statistical classification and regression, In: F. Seillier-Moisewitsch (ed.): *Statistics in Molecular Biology and Genetics, IMS Lecture Notes – Monograph Series*, **33**, 288–313.
- [7] EDLER, L., GRASSMANN, J., and SUHAI, S., 2001: Role and Results of Statistical Methods in Protein Fold Class Prediction, *Mathematical and Computer Modelling*, **33**, 4201–4217.
- [8] GRASSMANN, J., REZKO, M., SUHAI, S., and EDLER, L., 1999: Protein Fold Class Prediction – New Methods of Statistical Classification In: T. Lengauer (ed.): *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology (ISMB)*, AAAI Press, Menlo Park CA, 106–112.
- [9] GUNN, S. R., 1997: Support Vector Machines for Classification and Regression. *Technical Report*, Image Speech and Intelligent Systems Research Group, University of Southampton.
- [10] JAAKKOLA, T., AND DIEKHANS, M., and HAUSSLER, D., 2000: A discriminative framework for detecting remote protein homologies, *Journal of Computational Biology*, **7**, 95–114.
- [11] KARAKOULAS, G. and SHAWE-TAYLOR, J., 1999: Optimizing classifiers for imbalanced training sets, In: M. Kearns, S. Solla and D. Cohn (eds.): *Advances in Neural Information Processing Systems 11*, The MIT Press, Cambridge, MA.
- [12] LESLIE, C., ESKIN, E., and STAFFORD NOBLE, W., 2002: The Spectrum Kernel: A String Kernel for SVM Protein Classification, In: *Proc. Pacific Symposium on Biocomputing 7*, 566-575.
- [13] MARKOWETZ, F., 2001: *Support Vector Machines in Bioinformatics*, Master’s thesis, Mathematical Department of Ruprecht-Karls-University Heidelberg.
- [14] PASCARELLA, S., and ARGOS, P., 1992: A databank merging related protein structures and sequences, *Protein Engineering*, **5**, 121–137.
- [15] REZKO and BOHR, 1994: The DEF data base of sequence based protein fold class predictions, *Nucleic Acids Research*, **22**, 3616–3619.

- [16] SCHEERLINCK, J.-P.Y., LASTERS, I., CLAESSENS, M., DE MAEYER, M., PIO, F., DELHAISE, P., and WODAK, S., 1992: Recurrent $\alpha\beta$ Loop Structures in TIM Barrel Motifs Show a Distinct Pattern of Structural Features, *Proteins: Structure, Function, and Genetics*, **12**, 299–313.
- [17] SCHÖLKOPF, B. and SMOLA, A. 2002, *Learning with Kernels*, MIT Press, Cambridge, MA.
- [18] VAPNIK, V., 1995: *The Nature of Statistical Learning Theory*, Springer, New York.
- [19] VAPNIK, V., 1998: *Statistical Learning Theory*, Wiley, New York.

FLORIAN MARKOWETZ and MARTIN VINGRON
Max-Planck-Institute for Molecular Genetics
Computational Molecular Biology
Innestrasse 63-73
D-14195 Berlin, Germany
Phone: +49 (30) 8413-1352; Fax: -1152
E-mail: florian.markowetz@molgen.mpg.de; vingron@molgen.mpg.de

LUTZ EDLER
German Cancer Research Center (DKFZ)
Im Neuenheimer Feld 280
D-69120 Heidelberg, Germany
Phone: +49 (6221) 42-2392; Fax: -2397
E-mail: edler@dkfz.de