Kernel-based Machine Learning on Sequence Data from Proteomics and Immunomics

Dissertation

der Fakultät für Informations- und Kognitionswissenschaften
der Eberhard-Karls-Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von

M.Sc. Nico Pfeifer

aus Hannover

Tübingen
2009

Tag der mündlichen Qualifikation:	22.07.2009
Dekan:	Prof. Dr. Oliver Kohlbacher
1. Berichterstatter:	Prof. Dr. Oliver Kohlbacher
2. Berichterstatter:	Prof. Dr. Knut Reinert

Zusammenfassung

Ein großes Anwendungsgebiet für Maschinelle Lernverfahren ist die Biologie. Hierbei reichen die Anwendungen von der Vorhersage von Genen über die Vorhersage der Aktivität von Wirkstoffen bis hin zur Vorhersage der dreidimensionalen Struktur eines Proteins. Im Rahmen dieser Dissertation wurden kernbasierte Lernverfahren entwickelt in den Bereichen der Proteomik und der Immunomik. Alle Anwendungen haben hierbei das Ziel, bestimmte Eigenschaften von Teilen von Proteinen, so genannten Peptiden, vorherzusagen, welche in vielen biologischen Prozessen eine wichtige Rolle spielen.

Im ersten Teil der Dissertation stellen wir einen neuen Kern vor, der zusammen mit einer Support-Vektor-Maschine benutzt werden kann, um das chromatographische Verhalten von Peptiden in Umkehrphasen-Flüssigchromatographie und starker Anionenaustauschchromatographie vorherzusagen. Der Prädiktor für die Flüssigchromatographie wird daraufhin verwendet, um einen p-Wert basierten Filter für Peptididentifikationen in der Proteomik zu entwickeln. Der Filter beruht auf der Idee, dass das vorhergesagte Retentionsverhalten ähnlich zum gemessenen Verhalten sein sollte. Ist dies nicht der Fall, so ist das ein Indiz dafür, dass die identifizierte Peptidsequenz falsch ist. Hierdurch können falsch identifizierte Peptide herausgefiltert werden. Dies kann zum einen dazu verwendet werden, um die Qualität der Identifikationen zu verbessern. Zum anderen können mehr Identifikationen erhalten werden, indem auch nicht ganz sichere Identifikationen betrachtet werden, da der Filter viele falsche Identifikationen herausfiltern und somit einen guten Qualitätsgrad garantieren kann.
Im darauffolgenden Abschnitt zeigen wir, dass dieses Verfahren auch für zweidimensionale Trennverfahren verallgemeinert werden kann, was zu einem weiteren Anstieg an Peptididentifikationen bei ähnlicher Qualität führt. Außerdem zeigen wir am Beispiel des Organismus *Sorangium cellulosum*, dass das Verfahren sehr gut für die Verbesserung der Messungen von ganzen Proteomen geeignet ist. Für diese Anwendung können wir zeigen, dass wir bei ähnlicher Präzision ca. 25% mehr Spektren identifizieren können.
Der nächste Abschnitt zeigt, dass der neue Kern auch zur Vorhersage proteotypischer Peptide geeignet ist. Dies sind Peptide, die mit massenspektrometriebasierten Verfahren gemessen werden können und Proteine eindeutig identifizieren. Zusätzlich kann die gelernte Diskriminante sehr gut dafür verwendet werden um festzustellen, welche Aminosäuren an welchen Positionen die Wahrscheinlichkeit eines Peptids erhöht proteotypisch zu sein.
Die Fähigkeit eines Peptids eine Immunantwort auszulösen hängt von seiner Bindeaffinität zu einem speziellen Rezeptor des Immunsystems ab, welcher

MHC Rezeptor genannt wird. Es gibt verschiedene Varianten dieses Rezeptors, die in zwei Klassen eingeteilt werden können. Wir präsentieren einen kernbasierter Ansatz um die Bindeaffinität von Peptiden zu MHC Klasse II Rezeptoren präzise vorherzusagen. Außerdem zeigen wir, wie Prädiktoren für bestimmte Varianten dieses Rezeptors gebaut werden können, obwohl für sie keine experimentellen Daten verfügbar sind. Hierzu werden experimentelle Daten von anderen Varianten des Rezeptors verwendet. Durch dieses Verfahren können wir für gut zwei Drittel aller MHC Klasse II Rezeptoren Prädiktoren erstellen im Gegensatz zu ca. 6%, für die vorher Prädiktoren existierten.

Abstract

Biology is a large application area for machine learning techniques. Applications range from gene start prediction over prediction of drug activity to the prediction of the three-dimensional structure of proteins. This thesis deals with kernel-based machine learning in proteomics and immunomics applications. In all applications, we are interested in predicting properties of peptides, which are parts of proteins. These peptides play an important role in many biological systems.

In the first part, we introduce a new kernel which can be used together with a support vector machine for predicting chromatographic separation of peptides in reversed-phase liquid chromatography and strong anion exchange solid-phase extraction. The predictor for reversed-phase liquid chromatography can be used to build a p-value-based filter for identifications in proteomics. The filter is based on the idea that if the measured and the predicted behavior differ significantly, the identified sequence is probably wrong. In this way, we can filter out false identifications. First, this is useful for increasing the precision of identifications. Second, one can lower mass spectrometric scoring thresholds and filter out false identifications to get a significant increase in the number of correctly identified spectra at comparable precision. We also show in the following section that we can extend our method to predict retention times in two-dimensional chromatographic separations, which leads to a further increase in the number of correctly identified spectra at quality comparable to the unfiltered case. The practical applicability is demonstrated by applying the methods to a whole proteome measurement of *Sorangium cellulosum*. We can show that we can get about 25% more spectrum identifications at the same level of precision.
The next section shows that the new kernel can also be applied to the prediction of proteotypic peptides. These are peptides which can be detected by mass spectrometry-based analysis techniques and which uniquely identify a protein. We furthermore show that the resulting discriminant is very useful for discovering which amino acids influence the likelihood of a peptide to be proteotypic.
The ability of a peptide to induce an immune response depends upon its binding affinity to a specialized receptor, called major histocompatibility complex (MHC) molecule. There are different variants of this receptor that can be classified into two classes. We introduce a kernel-based approach for predicting binding affinity of peptides to MHC class II molecules with high accuracy and show how to build predictors for variants of this receptor, for which no

experimental data exists, based on data for other variants. This enables us to build predictors for about two thirds of all different MHC class II molecules instead of about 6%, for which predictors had already been available.

Acknowledgments

First of all, I would like to thank my supervisor, Professor Oliver Kohlbacher, for giving me the opportunity to pursue this very interesting research, his guidance, epecially at the beginning of my thesis and his sharp and open mind. He always supported me and gave me the opportunity to follow the research that interested me most. I also want to thank Professor Knut Reinert very much for reviewing this thesis. Additionally, I am very thankful to Professor Christian G. Huber and Andreas Leinenbach for great collaborations.
Furthermore, I am very grateful to Peter Meinicke, Professor Burkhard Morgenstern and especially Professor Stephan Waack who introduced me to, and kindled my fascination for, the fields of computational biology and machine learning during my years of study in Göttingen.
Additionally, I want to thank the whole OpenMS team for nice collaboration and retreats, Till-Helge Hellwig and Kay Ohnmeiß for the effort, they put into their bachelor theses, as well as the remaining staff of the Simulation of Biological Systems Department, namely Andreas Bertsch, Sebastian Briesemeister, Magdalena Feldhahn, Nina Fischer, Sandra Gesing, Andreas Kämper, Erhan Kenar, Cengiz Koc, Sven Nahnsen, Lars Nilse, Marc Röttig, Marcel Schumann, Marc Sturm, Philipp Thiel, Nora Toussaint, Jan Schulze, Chun-Wei Tung, and Claudia Walter as well as its former members Torsten Blum, Pierre Dönnes, Annette Höglund, Andreas Kerzmann, and Jana Schmidt for a nice working atmosphere and interesting conversations.

I am deeply grateful to my parents who have always supported and equipped me with all the tools and skills that I have needed.

Last but definitely not least, I am very much obliged to my wife Ina, who fills my life with joy and inspires me to be a better person every day.

Contents

Chapter 1

Introduction

"Wissen und Erkennen sind die Freude und die Berechtigung der Menschheit."
- Alexander von Humboldt, Kosmos, Stuttgart 1845, volume 1, page 36

Translated into English this means, "Knowledge and recognition are the joy and entitlement of mankind". When the famous naturalist and explorer published these words in his five-volume work *Kosmos*, he probably did not think of discovering biological knowledge by machine learning techniques. Nevertheless, he recognized that the wealth of knowledge of a society is highly correlated with its prosperity. Nowadays, there is a large research field that is just concerned with building learning machines. This field is mainly influenced by statistics and optimization techniques.
The term *artificial intelligence* (AI) was first coined by John McCarthy in 1955 in the proposal for the Dartmouth Conference, which took place during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The proposal contained these two sentences: "The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves." (J. McCarthy, M. L. Minsky, N. Rochester, C.E. Shannon, August 31, 1955). A sub-field of AI is the field of machine learning. Machine learning can be described by the last part of the second sentence. The learning algorithm tries to solve a problem. A typical problem is a supervised two-class (binary) prediction problem. In this setting, one has training data for which the classes are known and some extra data, for which the classes are unknown. The problem is to label the extra data with the correct class label. A common application is a spam filter. In this application, the training data consists of mails, for which the label is known (*spam* or *no spam*). The problem is to predict whether an incoming mail is spam or not.
The two most prominent topics in machine learning in the last ten years have been kernel-based learning machines [17] and graphical models [39]. Kernels allow the transformation of data into a (mostly high-dimensional) feature space and solve the learning problem efficiently in this space. The choice of the kernel is in most applications the critical part because it directly relates

to the feature space. If the problem is easily solvable in the feature space, the kernel choice was good, otherwise one has to find a different kernel. If there is no suitable kernel at hand, researchers usually encode their data by certain *features* that they identified to be important for the problem. In the spam filter example, one could think of counts for phrases that occur often in spam mails like *cash bonus*, *free installation*, and *lose weight* as possible features. In this way, the feature spaces are constructed explicitly. One can then use standard methods to solve the problem. In many cases, it is not clear which features are best and, therefore, the standard approach is to define a set of reasonable features and perform a feature selection. In the spam filter example, one could count all English phrases with less than four words and remove all phrases which are not discriminative for one of the two classes. For a given dataset, this method usually suffices to achieve good performance, but the application of the same features to a slightly different dataset might lead to bad results if important features for the new dataset are missing. The kernel approach is usually more general, because it puts some mild assumptions on the data and learns all important features from the given data.

A large application area for machine learning techniques is biology [126]. One of the earliest applications was the prediction of translation inititation sites in *E. coli* by the perceptron algorithm [103]. In biology, it is very often the case that one has a set of sequences that possess a certain property (e.g., the sequence is an RNA sequence that acts as a translation initiation site or not). These sequences are typically measured by time- and money-consuming experiments. Since it is usually not feasible to measure all possible sequences, a common method is to train a machine learning algorithm on the measured data and predict the property for all unseen sequences of interest [43]. There are also settings, whose properties are unknown beforehand and so the machine learning methods are applied to construct clusters, in which the sequences inside the cluster are similar to each other and dissimilar to sequences from other clusters [128]. Furthermore, there are intermediate scenarios where one knows properties for part of the data [47].

In this thesis, we are mainly interested in the first setting. We have a set of training samples with certain properties and want to build a learning machine that is able to predict the property for further sequences very accurately. During the whole thesis, the training samples are parts of proteins, called peptides. The properties that we want to predict for the peptides depend on the application area.

Proteomics deals with the analysis of the proteome, which consists of all proteins. Mostly, the analysis is restricted to a certain cell type of a specific organism at a particular time point. There exist various techniques to measure the proteins under these defined conditions. The usual workflow starts with cutting the proteins into peptides by a digestion enzyme. Then, the peptides are separated by chromatography. The method of choice for large-scale analyses is usually based on tandem mass spectrometry [142, 1]. To be able to measure the peptides by mass spectrometry, they have to be ionized. The peptide ions are then directed into a mass spectrometer. This mass spec-

trometer measures the mass-to-charge ratio of the ions. Typically, the three most abundant peptide ions are chosen for further fragmentation in a collision chamber and directed into a second mass spectrometer. The peptides are then identified by the mass spectrum of the second mass spectrometer, which ideally contains the mass to charge ratios of all possible fragments of the peptide [87]. In database search methods, the measured spectra are compared to theoretical spectra for all peptides contained in the database. The highest scoring candidate is then delivered as identification of the spectrum. Unfortunately, the spectrum quality is not always good enough to identify the peptides correctly. Therefore, the identification routines usually define a certain scoring threshold to decide which of the identifications are certain. In these standard approaches, the chromatographic behavior of the peptide is not used for identification, although it is routinely measured by the instruments.

If high-performance liquid-chromatography is used for chromatographic separation, the peptides elute at a certain point in time, the retention time. There already exist methods for retention time prediction like the approaches by Petritis *et al.* [90, 91]. They trained artificial neural networks with a large number of training samples (several thousands). Since retention behavior of peptides differs for different separation columns, one would have to measure this amount of training peptides before being able to train and use their predictor, whenever the conditions of the column changed. Other approaches, like the linear model by Krokhin [60], are trained for very specific column types. Very recently, Klammer *et al.* [55] introduced a method based on a support vector machine (SVM). They used several features together with the linear as well as the RBF kernel and stated that they needed at least 200 unique spectrum identifications to train their learning machine.

The first goal of this thesis was to develop an efficient learning machine for learning chromatographic behavior of a peptide which does not need that many training samples. Having a good predictor, one can compare the predicted behavior to the measured behavior and filter out identifications for which observed and predicted behavior differ significantly. Therefore, one can lower the threshold of the identification routine to get correct identifications below the original scoring threshold. Since the filter is able to filter out many false identifications, one can achieve the same accuracy as standard identification routines, while identifying more spectra.

Another important property of a peptide with respect to mass spectrometry is its detectability or proteotypicity. It was recently observed that certain peptides are detected more often in mass spectrometric experiments than others [63]. If these peptides can be uniquely assigned to a protein they are called proteotypic. Especially for targeted proteomics (e.g., in multiple reaction monitoring experiments [23]), it is useful to know the proteotypic peptides of a protein. Since a peptide has to be able to pass through all different parts of the experimental setup to finally be detected, there can be very different properties of the peptide that are responsible for not detecting it. For example, there are peptides that do not ionize or fragment as well as others. Tang *et al.* [125] first introduced a method for predicting the detectability of a peptide. Mallick *et al.* [73] and Lu *et al.* [70] also addressed

this issue with slightly different methods. All of these methods were based on several biochemical properties of peptides which were either selected manually or by feature selection algorithms.
An additional important peptide property is its ability to induce an immune response by binding to major histocompatibility complex (MHC) molecules. MHC molecules present peptides at the cell surface. There are two different classes of MHC molecules. MHC class I molecules present peptides that are derived from proteins inside the cell, whereas MHC class II molecules present peptides that originate from outside of the presenting cell. Peptides, derived from proteins of pathogens like bacteria, viruses or fungi, which are bound to MHC class I (MHCI) or MHC class II (MHCII) molecules can be recognized by specialized immune cells, called T cells. These cells can then elicit an immune response. This response may lead to the death of the infected cells and/or clearance of the pathogen from the human body. Since not every peptide can bind to every MHC molecule, it is important to know which peptides bind to which MHC molecule in order to design peptide-based vaccines [114]. These vaccines do not contain all of the proteins of the pathogen. Instead, they contain a set of peptides. To facilitate the selection of peptide candidates for a vaccine, it is important to know which peptides bind to the particular MHC molecules. There have been many studies to address the problem of peptide-MHCII binding affinity prediction. Early approaches were based on positional scoring matrices [9, 80, 96, 99, 116, 124] but approaches with artificial neural networks [8], Gibbs samplers [81] and SVMs [24, 105, 137] with standard kernels were also presented. Especially for MHCII, data from experimental binding studies is very scarce, which complicates the problem of peptide-MHCII binding affinity prediction. Furthermore, the binding core, which is the part of the peptide that mainly affects binding affinity, is unknown for most of the experimental data. This makes the prediction problem quite complicated. Most existing methods are just applicable to a very small subset of known MHCII molecules.

Scientists like von Humboldt discovered biological knowledge by observations. Consequently, the traditional approach to discover which properties a certain peptide possesses would be to measure them by wetlab experiments. Though, in many applications we are just interested in the positive examples, e.g., whether the peptide is proteotypic. We might also be interested in the minimal set of all possible peptides of a bacterial proteome that bind to a predefined number of different MHCII molecules, because these peptides could be the most promising candidates for an epitope-based vaccine [132]. If one wanted to discover these peptides, one would have to measure all possible peptides of the proteome with the traditional approach. Since many experiments are usually needed, a more efficient approach is to build accurate predictors for peptide properties. If experimental confirmation is required, one can at least limit the number of experiments by predicting the most promising peptide candidates for a particular property.
In this work, we introduce two new kernel functions for computational proteomics. They are called the oligo-border kernel (OBK) and the paired oligo-border kernel ($POBK$) and can be used together with an SVM for predicting

chromatographic separation of peptides as well as for predicting proteotypic peptides by mass spectrometry-based experiments. The key idea of these kernels is to modify the oligo kernel, introduced by Meinicke *et al.* [76] for sequences of the same length, to account for sequences of different lengths.
Using the POBK together with an SVM, we show that we can build very accurate predictors for prediction of chromatographic separation in strong anion exchange chromatography that are significantly better than all previous approaches. Furthermore, we show that the POBK together with ν-support vector regression [111] can be used to predict retention times in ion-pair reversed-phase liquid chromatography. These predictors are then used to build a p-value-based filter for identified peptides, measured by this chromatography and tandem mass spectrometry. In this way, we are able to improve the precision of the identifications. Furthermore, the filter allows one to lower mass spectrometric scoring thresholds to identify more spectra with acceptable accuracy. We show the generality of our approach by applying the same methods to a dataset measured by two-dimensional chromatographic separation [20]. Thus, we build accurate predictors for the first separation dimension at pH 10.0 as well as for the second dimension at pH 2.1. The usefulness of this approach is shown on a whole proteome measurement of *Sorangium cellulosum*.
For predicting proteotypic peptides, we combine the POBK with an SVM. This method is compared to other approaches, which were summarized in [40]. In this evaluation the features of the most prominent methods for proteotypic peptide prediction (Mallick *et al.* [73] and Lu *et al.*[70]) were used together with an SVM to compare performances on the data of Mallick *et al.* [73]. We show that for this benchmark our method performs best, although we do not depend on specific features like the other approaches. Therefore, our method should also be applicable to experimental setups other than those presented in [73]. Furthermore, the kernel function allows the visualization of important amino acids for the classifier. These insights might be used for *in silico* design of proteotypic peptides or to discover properties of the involved biochemical processes.
For immunomics, we show how to transform the peptide-MHCII binding affinity prediction problem into a well-known machine learning problem called multiple instance learning. This transformation allows building predictors for MHCII molecules for which there exists training data. A comparison to a large benchmark study by Wang *et al.* [138] shows that the performance of our method is as good as state-of-the-art methods or even better. Furthermore, we introduce a new kernel function for immunomics called the positionally-weighted RBF kernel. This kernel can be used to incorporate knowledge from MHCII molecules into the kernel to build predictors for about two thirds of all known MHCII molecules. Before, predictors were just available for less than 6% of MHCII molecules.

The thesis is structured into five chapters. After this introduction, the second chapter introduces the theoretical as well as the biological background. Our developments for kernel-based machine learning in proteomics are described in the third chapter. The contributions of this work towards solving the

peptide-MHCII binding affinity prediction problem is described in the fourth chapter before the conclusion in the last chapter.

Chapter 2

Background

2.1 Machine Learning

2.1.1 General Idea

In many real-world applications, one is given labeled data and the goal is to come up with predictions for additional unlabeled data, which originates from the same source, based on general properties of the data. This is, for example, the case for stock markets, spam filtering or gene start prediction. More formally, one assumes that the data comes from the same but unknown source. Therefore, the data is *independent and identically distributed* (iid). One situation that is suitable for machine learning is when one has labeled data $\{(x, y)|x \in \mathcal{X} \wedge y \in \{-1, 1\}\}$, which is often referred to as *training data*, and unlabeled data $x \in \mathcal{X}$ with $\mathcal{X}$ being a topological space. The goal is to assign the most probable label y to the unlabeled data based on the knowledge gained from the training data. The optimal Bayes classifier which solves this task can be formulated as $g^*(x) = \text{argmax}_{y \in \mathcal{Y}} P(Y = y|X = x)$. Unfortunately, the optimal Bayes classifier cannot be constructed in general since the underlying distribution P of the data is generally unknown. This is why one has to come up with the best possible approximation of the Bayes classifier to find the best possible solution. To be more precise regarding the best possible approximation, we have to consider some theoretical background in the following sections.
The above task belongs to the *classification* problems and the special case with just two different labels is often referred to as *binary classification*. If there are more than two possible labels, the task is called *multi-class classification*. The task is called *regression* if the domain of the label is continuous (e.g., $y \in \mathbb{R}$).

2.1.2 Finding the best function

We already introduced the optimal Bayes classifier $g^*(x) = \text{argmax}_{y \in \mathcal{Y}} P(Y = y|X = x)$. Since we want to find the best approximation of the optimal predictor (both in classification and in regression tasks), we have to be able to compare the performance of different prediction functions. Therefore, we have to introduce the notion of risks. The risk of a function $f : \mathcal{X} \to \mathcal{Y}$ is the

expected error on all data that comes from the same source as the training data and is therefore iid. This means that

$$R(f) = \int_{\mathcal{X}\times\mathcal{Y}} c(x, y, f(x))\mathrm{d}P(x, y) \tag{2.1}$$

(c.f. [111]). The risk contains the function

$$c : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}. \tag{2.2}$$

This function is called the *loss function*. A common choice in binary classification is the 0-1 loss which is defined as:

$$c(x, y, f(x)) = \begin{cases} 1 & \text{if } f(x) \neq y \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

In general it is not clear what the best loss function for a particular problem is like. Consider, for example, a biomedical multi-class classification problem in which one has three classes. Each label represents a specific type of person. Based on this label the person gets a prescription for a drug. Now consider that we have three drugs d_1, d_2 and d_3. d_1 is very cheap but just helps people from class one. d_2 is more expensive than d_1 and is able to cure people of all classes, but for people from class three it leads to stronger side-effects. d_3 is as expensive as d_2 and is just able to cure people from class two and class three but leads to stronger side-effects in people from class two. If one is mainly interested in curing people, the loss for $c(x, 1, 2)$ should be smaller than the loss $c(x, 1, 3)$ since d_3 would not cure a person from class one. But even in this simple example one could come up with different loss functions if, for example, the price is of greater importance.
It could be important in some prediction tasks to know the amount of certainty of the prediction and not just the predicted label. In binary classification, one could think of the confidence as $y \cdot f(x)$, where $f(x)$ is now a real-valued function (positive values of $f(x)$ correspond to label $+1$ and negative values of $f(x)$ correspond to label -1). Higher values of $y \cdot f(x)$ correspond to higher certainty of the prediction. This leads to the *soft-margin loss function* of Bennett and Mangasarian [111]:

$$c(x, y, f(x)) = \begin{cases} 0 & \text{if } f(x) \cdot y \geq 1 \\ 1 - f(x) \cdot y & \text{otherwise} \end{cases} \tag{2.4}$$

A very similar loss function, called the *quadratic soft margin loss*, is the following:

$$c(x, y, f(x)) = \begin{cases} 0 & \text{if } f(x) \cdot y \geq 1 \\ (1 - f(x) \cdot y)^2 & \text{otherwise} \end{cases} \tag{2.5}$$

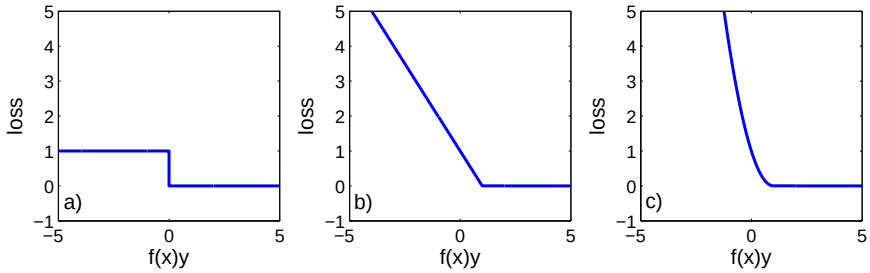

Figure 2.1: ***Different loss functions for binary classification:*** *a) 0-1 loss, b) soft-margin loss, and c) squared soft-margin loss*

Plots of the different loss functions can be seen in Fig. 2.1. For regression problems, the most common loss functions are squared loss

$$c(x, y, f(x)) = (f(x) - y)^2 , \tag{2.6}$$

ε-insensitive loss

$$c(x, y, f(x)) = \max\left(|f(x) - y| - \varepsilon, 0\right), \tag{2.7}$$

in which a deviation between y and $f(x)$ smaller than ε is not penalized, and the l_1-loss

$$c(x, y, f(x)) = |f(x) - y| \tag{2.8}$$

in which every deviation is penalized by its absolute value. Fig. 2.2 shows a plot of these three loss functions. Although we know the most common

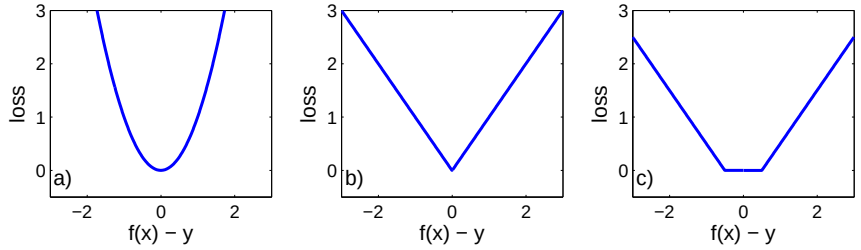

Figure 2.2: ***Different loss functions for regression:*** *a) squared loss, b) ε-insensitive loss, and c) l_1-loss*

loss functions we cannot compute the risk given by formula (2.1) since the distribution $P(x, y)$ is unknown. Nevertheless, we can calculate the risk on the training data, which is assumed to be sampled iid from the distribution

$$R_{emp}(f) = \sum_{i=1}^{n} c(x, y, f(x)) \tag{2.9}$$

(c.f. [111]). This risk is called the empirical risk and it can be used to find a good predictor. The induction principle of *empirical risk minimization* can be described as follows. Given a model class $\mathcal{F}$, which contains several functions, choose the function $f \in \mathcal{F}$, which minimizes the empirical risk (cf. [6]):

$$f_{emp} = \arg\min_{f \in \mathcal{F}} R_{emp}(f). \tag{2.10}$$

It is clear that empirical risk minimization does not guarantee good results. If the class of models contains only very simple models, one cannot expect the risk to be small. For example, given a model class that contains all linear hyperplanes, one could obtain good results if the distribution $P(X, Y)$ is very simple (e.g., as shown in Fig. 2.3), but even for slightly more difficult data (e.g., as shown in Fig. 2.4), the model class would be too simple to find a good classifier. Furthermore, if the function class has very flexible functions,

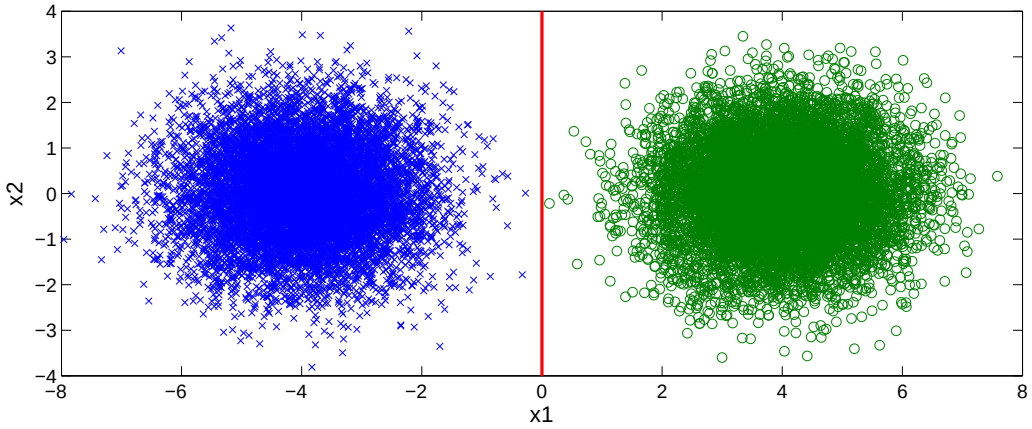

Figure 2.3: ***Example for linearly separable data:*** *The blue points are negative examples and the green points are positive examples. The red line shows one possible separation between these two classes.*

one could expect that a very specialized function could be chosen, i.e. one, which performs very well on training data but does not perform well on unseen data. Therefore, the model class should not be too rich. This becomes clear if one considers the following example. If the class contained all possible functions then one could find a function which has zero empirical risk and classifies every new data point, drawn from the same distribution, wrongly. The classifier would have maximum risk and this is definitely not desirable. The idea of restricting the model class is included in the *structural risk minimization* induction principle (cf.[6]). In this principle one has an infinite sequence of models $\{f_1, f_2, ...\}$ which are sorted by their complexity, starting with the model of lowest complexity. The complexity of the model can be measured in different ways. If our hypothesis space consists, for example, of the union of all axis-parallel rectangles in which one hypothesis is a subset

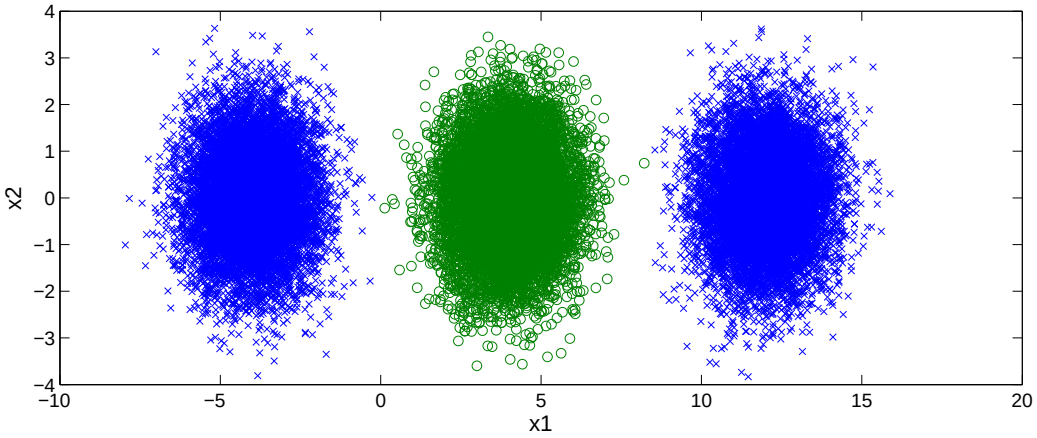

Figure 2.4: ***Example for data that is not linearly separable:*** *The blue points are negative examples and the green points are positive examples.*

of the whole hypothesis space, a straightforward measure of the complexity of the model is the number of rectangles. In *structural risk minimization* one tries to minimize the empirical risk as in empirical risk minimization but additionally, the size of the model is penalized as follows:

$$f_{str} = \arg \min_{f \in \mathcal{F}, d \in I\!N} R_{emp}(f) + pen(d, n), \tag{2.11}$$

in which n is the number of training samples, d is a number reflecting the complexity of the model (e.g., number of rectangles), and $pen(d, n)$ is the penalty function. Since it could be difficult to build an infinite sequence of models there is another slightly different idea, which is called *regularization*. In this induction principle, one chooses a very rich class of models and defines a regularizer on $\mathcal{F}$. In many applications this is simply the norm $\|f\|$ of $f \in \mathcal{F}$. The regularizer penalizes the complexity of the model. Finding the best model reduces to finding the minimium of

$$f_{reg} = \arg \min_{f \in \mathcal{F}} R_{emp}(f) + \lambda \|f\|^2. \tag{2.12}$$

The parameter λ is called the *regularization parameter*. It can be used to find the best trade-off between small model complexity and minimizing the empirical risk. Finding a good value of λ is not trivial. Therefore, one uses validation schemes in which some parts of the training data are left out to get a good estimate of the error on unseen data given a certain value of λ. According to Bousquet *et al.* [6], the most successful methods in machine learning can be thought of as regularization methods.

2.1.3 Error Bounds

In the last section, we showed different principles that can be applied to find a prediction function f. The interesting question is now, how good are these

prediction functions? Therefore, we want to know whether we can bound the error that we make by choosing f. We already introduced the risk of a function. Let

$$R^* = \inf_{g \in \mathcal{G}} R(g), \tag{2.13}$$

in which $\mathcal{G}$ contains all possible measurable functions. The quality of f can be described as:

$$R(f) - R^* = [R(f^*) - R^*] + [R(f) - R(f^*)]\,. \tag{2.14}$$

f^* is the optimal function of the model class $\mathcal{F}$. The right-hand side of (2.14) decomposes into an approximation error (first term) and an estimation error (second term). Since one normally does not know anything about the best target function, one cannot directly bound the approximation error without making assumptions (e.g., about the value of R^*). Therefore, much of the literature deals with bounds on the estimation error, for which one does not need these kinds of assumptions.

2.1.4 Learning Machines

Perceptron Algorithm

One of the oldest and simplest learning machines is the perceptron algorithm introduced by Rosenblatt [103] in 1958. The goal of this algorithm is to find a separating hyperplane between the data points which come from two different classes. Therefore, it tries to adjust the hyperplane according to the misclassified points. Let w be the normal vector and b the offset of the hyperplane. A point x with label $y \in \{-1, 1\}$ is misclassified if $y\,(< w, x > + b)$ is negative. Let w_k and b_k be the parameters of the hyperplane after step k. Let $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ be the training samples. The algorithm proceeds as follows:

```
last_mistake ← 0
k ← 1
i ← 1
initialize w with random values
initialize b with random value
while (k - last_mistake - 1) < n
  IF y_i (⟨w_{k-1}, x_i⟩ + b_{k-1}) < 0
  THEN
    w_k ← w_{k-1} + ρ y_i x_i
    b_k ← b_{k-1} + ρ y_i
    last_mistake ← k
  k ← k + 1
  i ← i + 1
  IF i > n
  THEN
    i ← 1
```

The learning rate of the algorithm, ρ, has to be greater than zero. It was shown that the algorithm converges if the data is linearly separable by a non-zero margin [83]. The motivation behind the update procedure is that one tries to minimize the distance between the misclassified points and the decision boundary. Therefore, the update shifts the hyperplane towards the misclassified data point. If training sample x_i is misclassified by hyperplane $k-1$ ($y_i = 1$ and $(< w_{k-1}, x_i > + b_{k-1}) < 0$ or $y_i = -1$ and $(< w_{k-1}, x_i > + b_{k-1}) > 0$), $y_i\,(< w_{k-1}, x_i > + b_{k-1})$ is smaller than zero. Let

$$L(w_{k-1}, b_{k-1}) = -y_i\,(< w_{k-1}, x_i > + b_{k-1})\,. \tag{2.15}$$

By minimizing L with respect to w_{k-1} and b_{k-1}, the distance between x_i and the actual hyperplane is minimized. This is why the algorithm descends along the gradient of L to find the best solution. The gradients with respect to the parameters of the hyperplane are:

$$\frac{\partial}{\partial w_{k-1}} L(w_{k-1}, b_{k-1}) = -y_i x_i \tag{2.16}$$

and

$$\frac{\partial}{\partial b_{k-1}} L(w_{k-1}, b_{k-1}) = -y_i. \tag{2.17}$$

It is clear that the algorithm will not converge if the data is not linearly separable. Furthermore, the algorithm stops if a separating hyperplane is found. This means that if there are many possible hyperplanes that can separate the data, the values of w and b are influenced by the order of the training samples, because the update takes place after a misclassification. Additionally, the initial values of w and b influence the final hyperplane. Fig. 2.5 a) shows the data that is generated by 400 random draws from the normal distribution leading to 200 two-dimensional samples. The data was split into two classes by adding seven to the second component of half of the data points. Fig. 2.5 b) shows 20 separating hyperplanes, which were found by implementing the above pseudo-code in Matlab and executing the function 20 times. Fig. 2.5 c) shows the whole region which can be covered by separating hyperplanes. Since we know how we generated the data, we also know the best possible function f^* out of the function class $\mathcal{F}$ that contains all lines. In this example, f^* is a line which is parallel to the first axis and has the value 3.5 in the second dimension. To show that not every line which separates the two classes is equally good, we drew 2000 additional samples from the same distributions and plotted them, the 20 discriminants of Fig. 2.5 b), as well as the optimal separating line (thick and red) in Fig. 2.6. It can be seen that the worst separating lines are the ones which are very

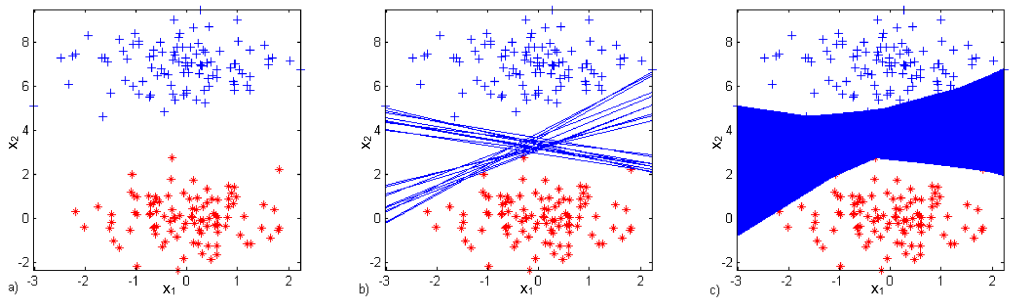

Figure 2.5: ***Visualization of Rosenblatt's perceptron algorithm:*** *This plot shows 200 data points drawn from the normal distribution. One hundred of these points are shifted by seven in the second dimension (crosses): a) shows the data without any separating lines; b) additionally shows 20 separating lines learned on the data using Rosenblatt's Perceptron Algorithm; and c) additionally colors the region in which the lines can be found by Rosenblatt's Perceptron Algorithm.*

close to the training samples. Furthermore, the best separating line (red) has maximal margin with respect to the nearest samples. This motivates why large margin hyperplane classifiers generalize well to unseen data. We will look at these kinds of learning machines in more detail in the next subsection.

Large Margin Classifiers

Let $\mathcal{H}$ be a dot product space. One can define a hyperplane by the normal vector and the offset of the hyperplane. The set of points that lie on the hyperplane can be calculated by projecting the points onto the normal vector w and adding the offset b. If the result is zero, the point lies on the hyperplane. The set of points that lie on the hyperplane is therefore:

$$\{x \in \mathcal{H} | < w, x > +b = 0\}. \tag{2.18}$$

Multiplying the normal vector and the offset by certain factors can yield the same set of points which lie on the corresponding hyperplanes. Therefore, Schölkopf and Smola [111] define the hyperplane with respect to some data points $x_1, x_2, ..., x_n \in \mathcal{H}$. This hyperplane is called the canonical hyperplane:

Definition 2.1 (Canonical Hyperplane)**.** *The parameters $w \in \mathcal{H}$ and $b \in \mathbb{R}$ describe a Canonical Hyperplane with respect to the data $x_1, x_2, ..., x_n \in \mathcal{H}$, if the point closest to the hyperplane has distance $1/\|w\|$, which means that*

$$\min_{i=1,2,...,n} | < w, x_i > +b| = 1. \tag{2.19}$$

We already saw in the last section, that large margin separations seem to be more robust than other separating hyperplanes. To construct a large

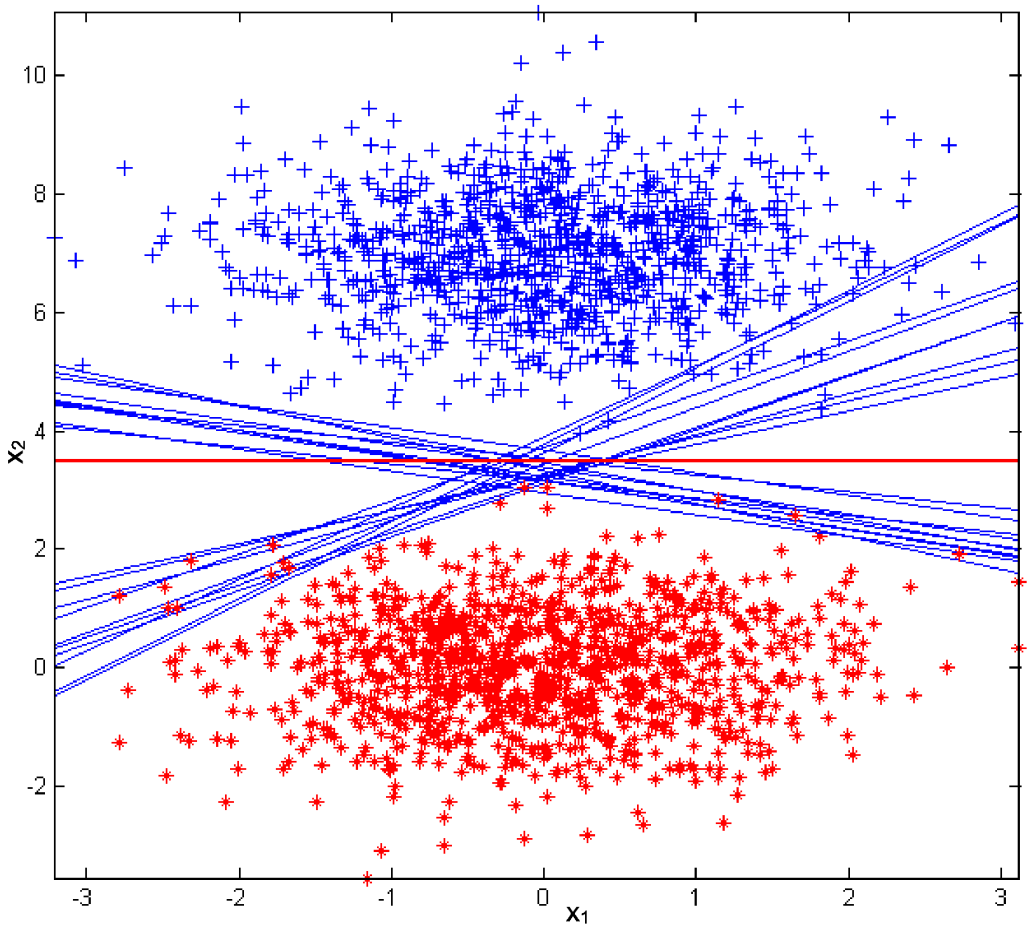

Figure 2.6: ***Visualization of test error of Rosenblatt's perceptron algorithm:*** *This plot shows a binary classification problem. The thin lines are 20 separating lines determined by Rosenblatt's Perceptron Algorithm based on 200 samples. In addition to these 200 samples, the plot contains 2000 extra samples drawn from the same distributions. The thick line corresponds to the optimal separation between the two classes.*

margin classifier one has to find the canonical hyperplane with maximal margin. Since the margin of the canonical hyperplane is $1/\|w\|$, the canonical hyperplane with maximal margin is the one with minimal $\|w\|$. This can be cast into a standard optimization problem:

$$\min_{w \in \mathcal{H}, b \in \mathbb{R}} \|w\|, \tag{2.20}$$
$$\text{subject to } y_i \left(\langle x_i, w \rangle + b \right) \geq 1 \ \forall i = 1, 2, ..., n.$$

The constraints assure that the w with minimal $\|w\|$ is a canonical hyperplane. This optimization problem yields the same solution as

$$\min_{w \in \mathcal{H}, b \in \mathbb{R}} \tfrac{1}{2}\|w\|^2, \tag{2.21}$$
$$\text{subject to } y_i \left(\langle x_i, w \rangle + b \right) \geq 1 \ \forall i = 1, 2, ..., n.$$

Since the optimization problem (2.21) has some nicer properties, it is used in the following. This optimization problem can be solved if the data is separable. To transform the primal problem into a dual problem we can build the Lagrangian:

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i \left(\langle x_i, w\rangle + b\right) - 1\right]. \tag{2.22}$$

To get the solution of the dual problem, the Lagrangian L has to be maximized with respect to α and minimized with respect to w and b (c.f. [62]). This means that we are trying to find a saddle point at which the derivatives of L with respect to the primal variables must vanish:

$$\frac{\partial}{\partial b} L(w, b, \alpha) = 0, \quad \frac{\partial}{\partial w} L(w, b, \alpha) = 0. \tag{2.23}$$

Therefore,

$$\frac{\partial}{\partial b} L(w, b, \alpha) = 0 \Leftrightarrow -\sum_{i=1}^{n} \alpha_i \cdot y_i \cdot 1 = 0 \Leftrightarrow \sum_{i=1}^{n} \alpha_i \cdot y_i \cdot 1 = 0 \tag{2.24}$$

and

$$\frac{\partial}{\partial w} L(w, b, \alpha) = 0 \Leftrightarrow w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0 \Leftrightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i. \tag{2.25}$$

The x_i, for which $\alpha_i > 0$ are called support vectors because they lie directly at the boundary of the margin of the canonical hyperplane. This is shown in Fig. 2.7. The classifier is usually called a support vector machine (SVM). It can be seen that the support vectors determine the hyperplane.

To arrive at the dual problem, one can write equation (2.22) in the following way:

$$\frac{1}{2}\langle w, w\rangle - \sum_{i=1}^{n} \alpha_i y_i \langle x_i, w\rangle - b \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i. \tag{2.26}$$

Substitution of (2.24) and (2.25) into (2.26) yields

$$\frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle - b \cdot 0 + \sum_{i=1}^{n} \alpha_i \tag{2.27}$$

$$= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle \tag{2.28}$$

The dual form of the optimization problem (2.21) is, therefore,

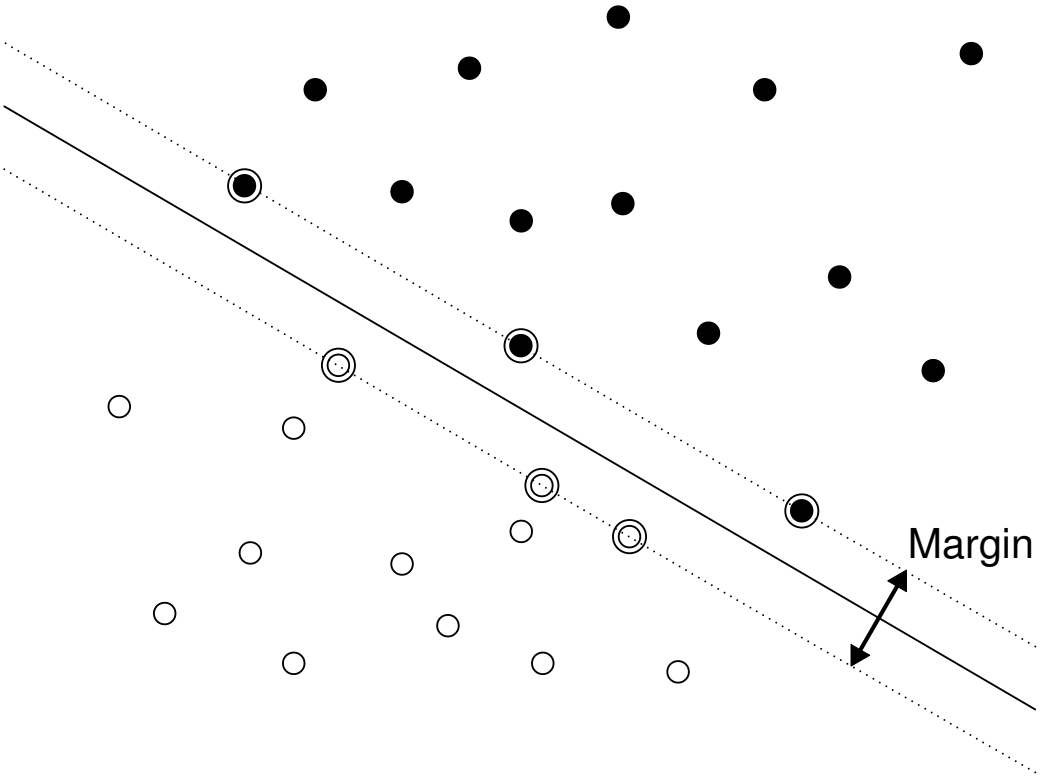

Figure 2.7: ***Separating hyperplane for linearly separable data:*** *This plot shows a two-class problem and a separating hyperplane. The support vectors are marked by additional circles.*

$$\begin{aligned} &\max_{\alpha \in \mathbb{R}} \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle, \\ &\text{subject to } 0 \leq \alpha_i \ \forall i = 1, 2, ..., n \text{ and} \\ &\sum_{i=1}^{n} \alpha_i y_i = 0. \end{aligned} \tag{2.29}$$

It can be shown that the duality gap between the primal and the dual is zero and, therefore, a solution to the dual problem also solves the primal problem. In real-world examples there are often samples, which are not linearly separable. Furthermore, perfect separation is not always the best choice if, for example, one of the points is an extreme outlier. Therefore, Cortes and Vapnik [17] introduced so-called slack-variables $\xi_i \geq 0$. These variables shift the points to the correct side of the canonical hyperplane, which is shown in Fig. 2.8. The classifiers that use slack-variables are called *soft margin classifiers*. Since not every point should be allowed to have a slack variable greater than 0, the value of the slack-variables has to be penalized in the minimization problem. This means that the minimization problem uses the *regularization* induction principle. There exist several different approaches in the literature to weight the slack-variables. The two most prominent ones are the 1-norm soft margin classifier and 2-norm soft margin classifier. We show the primal and dual problem for the 1-norm soft margin classifier. The steps for the 2-norm soft margin classifier are similar:

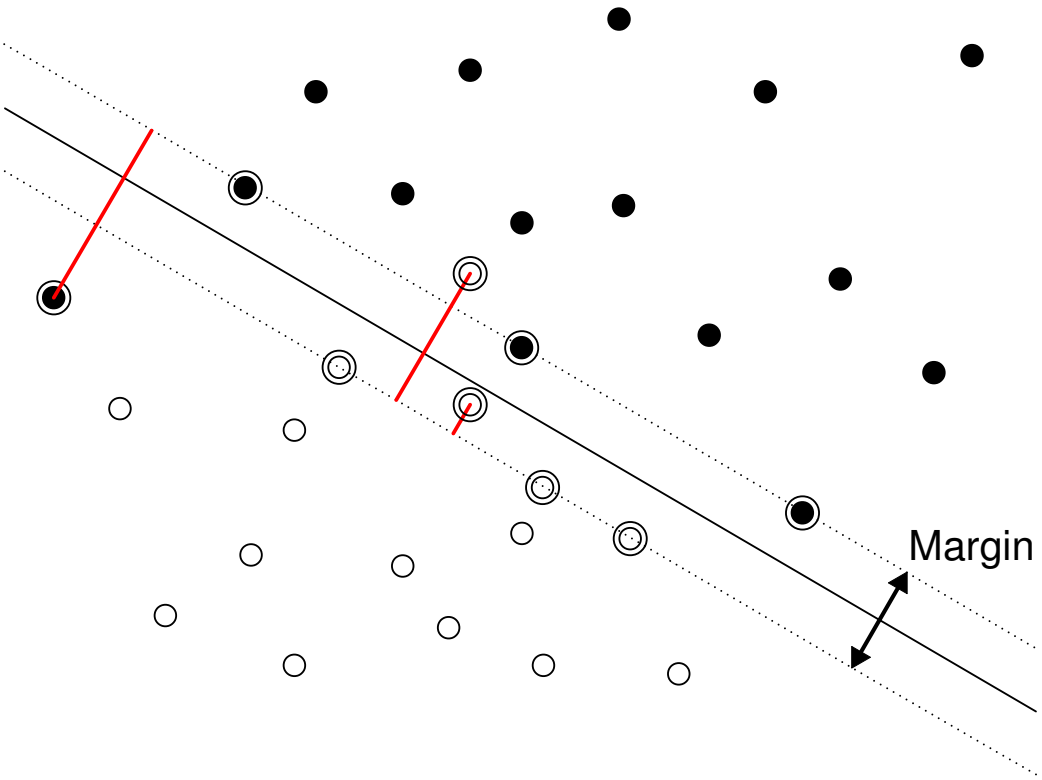

Figure 2.8: ***Separating hyperplane for data that is not linearly separable:*** *This plot shows a two-class problem as well as a separating hyperplane. The support vectors are marked by extra circles and the penalty of the ξ_i is indicated by the red lines.*

$$\min_{w\in\mathcal{H},b\in\mathbb{R},\xi\in\mathbb{R}^n} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i, \qquad (2.30)$$
$$\text{subject to } y_i\left(\langle x_i, w\rangle + b\right) \geq 1-\xi_i \;\forall i = 1,2,...,n.$$

The Lagrangian of (2.30) is

$$L(w,b,\xi,\alpha,\beta) = \\ \frac{1}{2}\langle w,w\rangle + C\sum_{i=1}^{n}\xi_i \quad -\sum_{i=1}^{n}\alpha_i\left[y_i\left(\langle x_i,w\rangle + b\right) + \xi_i - 1\right] - \sum_{i=1}^{n}\beta_i\xi_i. \qquad (2.31)$$

As in the separable case, we try to find a saddle point at which the derivatives of L with respect to the primal variables must vanish:

$$\frac{\partial}{\partial b}L(w,b,\xi,\alpha,\beta) = 0 \Leftrightarrow -\sum_{i=1}^{n}\alpha_i\cdot y_i\cdot 1 = 0 \Leftrightarrow \sum_{i=1}^{n}\alpha_i\cdot y_i\cdot 1 = 0, \qquad (2.32)$$

$$\frac{\partial}{\partial w}L(w,b,,\xi,\alpha,\beta) = 0 \Leftrightarrow w - \sum_{i=1}^{n}\alpha_i y_i x_i = 0 \Leftrightarrow w = \sum_{i=1}^{n}\alpha_i y_i x_i \qquad (2.33)$$

and

$$\frac{\partial}{\partial \xi_i} L(w, b, \xi, \alpha, \beta) = 0 \Leftrightarrow C - \alpha_i - \beta_i = 0. \tag{2.34}$$

Substitution of (2.32) and (2.33) into (2.31) yields

$$\begin{aligned} & \tfrac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle && - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_{i=1}^{n} \alpha_i + C \sum_{i=1}^{n} \xi_i \\ & && - \sum_{i=1}^{n} \alpha_i \xi_i - \sum_{i=1}^{n} \beta_i \xi_i \\ = & \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle && + \sum_{i=1}^{n} (C - \alpha_i - \beta_i) \xi_i. \end{aligned} \tag{2.35}$$

Using equation (2.34) we obtain

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle. \tag{2.36}$$

Since $\beta_i \geq 0 \ \forall i = 1, 2, ..., n$ the dual form of the optimization problem is

$$\begin{gathered} \max_{\alpha \in \mathbb{R}} \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle, \\ \text{subject to } 0 \leq \alpha_i \leq C \ \forall i = 1, 2, ..., n \text{ and} \\ \sum_{i=1}^{n} \alpha_i y_i = 0. \end{gathered} \tag{2.37}$$

From (2.25) and (2.33) the final prediction function for the separable as well as the non-separable case follows:

$$f(x) = \operatorname{sign} \left(\sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle + b \right). \tag{2.38}$$

Up to now, we only considered binary classification. If there are more than two different possible labels, one has to extend the introduced approaches. Basically, there are three different ways of dealing with *multi-class* prediction problems. The first possibility is to train a classifier for every class, which discriminates the class from all other classes (one versus the rest). The class to which the classifier with maximal prediction value belongs determines the predicted class.

The second possibility is to train single classifiers for every pair of classes (pairwise classification). The final prediction is the class that is predicted the largest number of times.
The third possibility is to formulate the problem as a single optimization problem. This was shown in [111] but there are also very recent approaches in which the classifier tries to learn a large margin between the correct class and the other classes [145].

Support Vector Regression

To generalize support vector classification of the large margin classifiers to a regression problem, we have to restate one of the key observations from the last subsection. The weight vector $\|w\|$ can be described by a linear combination of a subset of the training points (support vectors with $\alpha_i > 0$). To get a similarly sparse solution for regression, Cortes and Vapnik [17] introduced an ε-insensitive band around the regression function where a deviation is not penalized. To allow for bigger deviations, the authors introduced two kinds of slack-variables $\xi_i \in \mathbb{R}$ and $\xi_i^* \in \mathbb{R}$. The ξ_i allow predictions which are larger than $y_i + \varepsilon$ and the ξ_i^* allow predictions smaller than $y_i - \varepsilon$. This can be seen in Fig 2.9. The ε-support vector regression optimization problem is

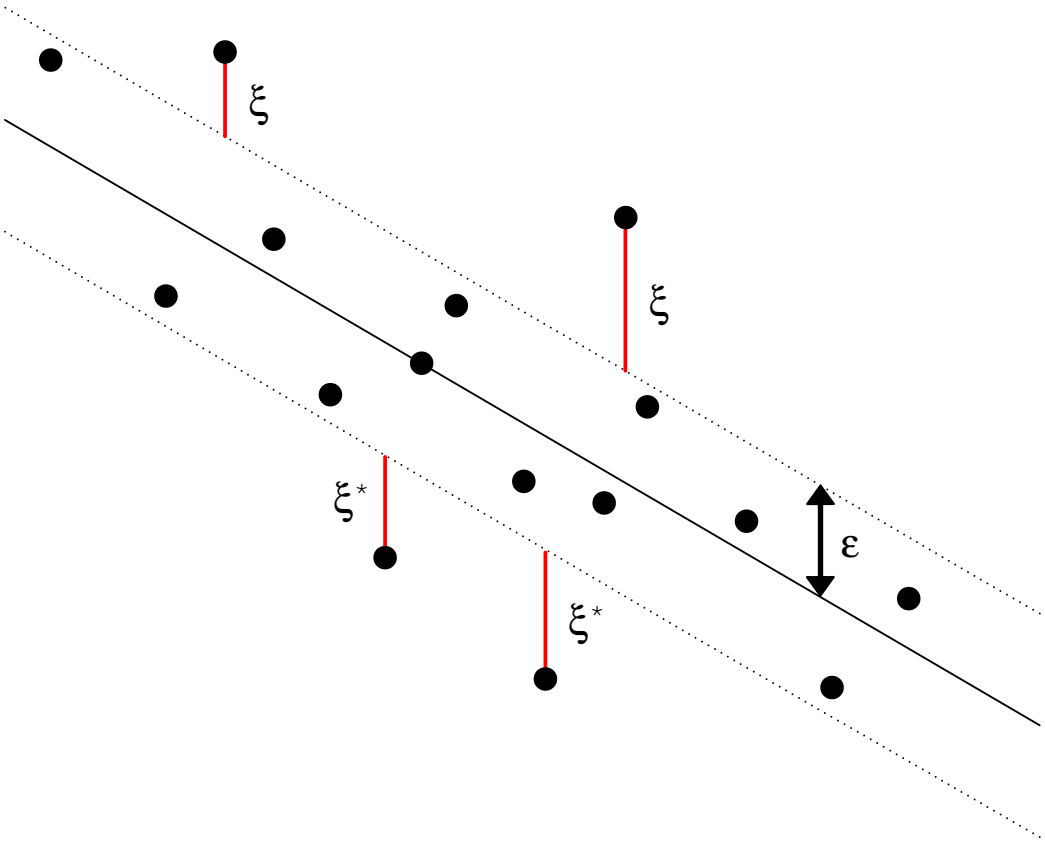

Figure 2.9: ***ε-SVR:*** *This picture shows the ε-insensitive tube around the regression line. Mistakes inside the tube are not penalized. All points on and outside the tube are called support vectors in this case.*

defined as:

$$\begin{aligned} & \min_{w\in\mathcal{H},b\in\mathbb{R},\xi_i\in\mathbb{R}^n,\xi_i^*\in\mathbb{R}^n} \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i+\xi_i^*), \\ \text{subject to} \quad & (\langle x_i, w\rangle + b) - y_i \le \varepsilon + \xi_i \ \forall i = 1,2,...,n \\ & y_i - (\langle x_i, w\rangle + b) \le \varepsilon + \xi_i^* \ \forall i = 1,2,...,n \\ & \xi_i \ge 0 \\ & \xi_i^* \ge 0. \end{aligned} \tag{2.39}$$

There also exists a dual formulation of the problem. Since in many applications one does not know the value of ε beforehand, there exists a slightly different formulation in which the optimal ε is identified during the optimization. It is called ν-support vector regression (ν-SVR).

$$\begin{aligned} & \min_{w\in\mathcal{H},\varepsilon,b\in\mathbb{R},\xi_i\in\mathbb{R}^n,\xi_i^*\in\mathbb{R}^n} \tfrac{1}{2}\|w\|^2 + C\left(\nu\varepsilon + \sum_{i=1}^{n}(\xi_i+\xi_i^*)\right), \\ \text{subject to} \quad & (\langle x_i, w\rangle + b) - y_i \le \varepsilon + \xi_i \ \forall i = 1,2,...,n \\ & y_i - (\langle x_i, w\rangle + b) \le \varepsilon + \xi_i^* \ \forall i = 1,2,...,n \\ & \xi_i \ge 0 \\ & \xi_i^* \ge 0. \end{aligned} \tag{2.40}$$

The Lagrangian of this problem is

$$\begin{aligned} & \tfrac{1}{2}\|w\|^2 + C\nu\varepsilon + C\sum_{i=1}^{n}(\xi_i+\xi_i^*) - \beta\varepsilon - \sum_{i=1}^{n}(\eta_i\xi_i + \eta_i^*\xi_i^*) \\ & - \sum_{i=1}^{n}\alpha_i(\xi_i + y_i - \langle w, x_i\rangle - b + \varepsilon) - \sum_{i=1}^{n}\alpha_i(\xi_i^* - \langle w, x_i\rangle + b - y_i + \varepsilon). \end{aligned} \tag{2.41}$$

Setting the derivatives equal to zero with respect to the primal variables and substituting into the Lagrangian leads to the dual optimization problem:

$$\begin{aligned} & \max_{\alpha\in\mathbb{R},\alpha^*\in\mathbb{R}} \sum_{i=1}^{n}(\alpha_i-\alpha_i^*)y_i - \tfrac{1}{2}\sum_{i,j=1}^{n}(\alpha_i-\alpha_i^*)\left(\alpha_j-\alpha_j^*\right)\langle x_i, x_j\rangle, \\ \text{s.t.} \quad & \sum_{i=1}^{n}(\alpha_i-\alpha_i^*) = 0, \\ & \alpha_i \in [0, C] \\ & \sum_{i=1}^{n}(\alpha_i-\alpha_i^*) \le C\cdot n\cdot\nu. \end{aligned}$$

The prediction function is thus:

$$f(x) = \sum_{i=1}^{n}(\alpha_i-\alpha_i^*)\langle x_i, x\rangle + b. \tag{2.42}$$

2.1.5 Kernels

So far we have only considered linear relationships in the data. The Perceptron Algorithm and the large margin classifiers were introduced to find a linear separation between classes and the SVR methods could also learn linear functions only. In many real-world datasets there are nonlinear relationships between the different entries of the input vector. A method that is not able to detect these similarities is expected to perform badly on this kind of data. Therefore, there exist nonlinear extensions for many linear learning approaches like SVMs, multiple linear regression, PCA, and Gaussian processes, to name just a few. Usually, this is done by mapping the data into a (mostly higher dimensional) feature space. The linear relationships in these feature spaces then correspond to more complex relationships in input space. A simple example is shown in Fig. 2.10. The circle in input space corresponds to a line in the feature space which, in this example, is able to separate the two classes visualized by blue crosses and red stars.

The computation of the mapping into the potentially infinite-dimensional

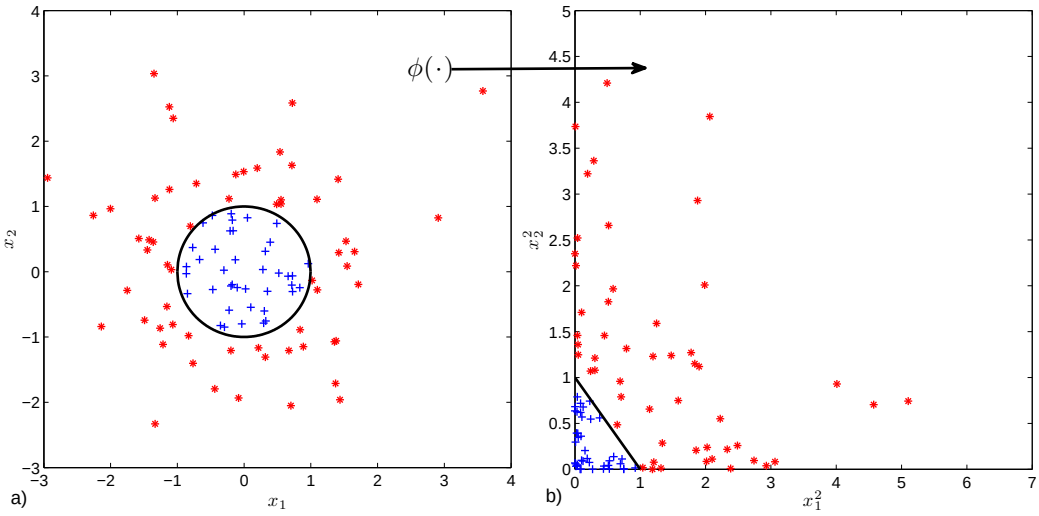

Figure 2.10: ***Mapping into feature space:*** *a) This plot shows 100 data points drawn from the normal distribution. All points inside the unit circle are positive (crosses) and the points outside are negative (stars); b) shows the same data mapped by the function* $\phi : \phi\binom{x_1}{x_2} = \binom{x_1^2}{x_2^2}$.

feature spaces can be quite time-consuming. Therefore, it is desirable to circumvent the explicit computation of the mapping. This is possible for all algorithms for which the data is repsesented by inner products (e.g., $\langle x_i, x_j \rangle$). The inner product of the mapped data is replaced by the so-called kernel function $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. Usually the kernel computation needs time proportional to the size of the data in input space. This is why one can tackle even infinite-dimensional feature spaces by using this so-called *kernel trick*. We will look at certain properties of kernels in more detail and then introduce specific kernels that are applicable for a huge variety of learning problems.

Properties of Kernels

The kernel function of two input sequences x_i, x_j has to be equal to the inner product of the mapped vectors, which means that $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. Let G be the Gram matrix with $G_{ij} = k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. Shawe-Taylor and Christianini [115] showed that every kernel function which fulfills this property is positive semidefinite, since for any vector v,

$$v^T G v = \sum_{i,j=1}^{n} v_i v_j G_{ij} = \sum_{i,j=1}^{n} v_i v_j \langle \phi(x_i), \phi(x_j) \rangle$$
$$= \langle \sum_{i=1}^{n} v_i \phi(x_i), \sum_{j=1}^{n} v_j \phi(x_j) \rangle = \| \sum_{i=1}^{n} v_i \phi(x_i) \|^2 \geq 0.$$

This directly implies that a function f with matrix $M_{ij} = f(x_i, x_j)$ which is not positive semidefinite cannot correspond to an inner product of feature vectors. This can be proven by contradiction using the above result. Positive semidefiniteness of the Gram matrix is one of the main properties a kernel has to have. For many kernels (e.g., the polynomial kernel) a map can be directly given such that $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. Nevertheless, there are numerous kernels for which no suitable feature map is known. In these cases, it is crucial to show that the corresponding Gram matrix is positive semidefinite, because otherwise the kernel could not correspond to any inner product in a feature space and, therefore, all learning algorithms would not be applicable. This is why we show positive semidefiniteness of our new kernel functions in the later chapters. It can be shown (c.f., [111, 115]) that for every positive semidefinite kernel k there exists a feature mapping ϕ into a feature space.

Reproducing Kernel Hilbert Spaces

In the last section we stated that there exists a map into a feature space for every positive semidefinite kernel. Furthermore, one can define a Hilbert space for these kernels which is called the *reproducing kernel Hilbert space* (RKHS) [111]:

Definition 2.2 (Reproducing Kernel Hilbert Space)**.** *Let $\mathcal{X}$ be a nonempty set and $\mathcal{H}$ a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$. $\mathcal{H}$ is called a reproducing kernel Hilbert space endowed with the dot product $\langle \cdot, \cdot \rangle$ and the norm $\|f\| := \sqrt{\langle f, f \rangle}$ if there exists a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following properties:*

1. *k has the reproducing property*

$$\langle f, k(x, \cdot) \rangle = f(x) \; \forall f \in \mathcal{H}, \tag{2.43}$$

   which means in particular that

$$\langle k(x, \cdot), k(x', \cdot) \rangle = k(x, x'). \tag{2.44}$$

2. *k spans $\mathcal{H}$, i.e. $\mathcal{H} = \overline{\text{span}\{k(x, \cdot) | x \in \mathcal{X}\}}$. $\overline{X}$ denotes the completion of the set X.*

One might argue that SVMs together with a kernel function which maps into a possibly high-dimensional feature space might not allow representation of the optimal hyperplane by a linear combination of the support vectors. Fortunately, Schölkopf *et al.* [109] showed that this is possible for all positive semidefinite real-valued kernels. More generally they showed [109]:

Theorem 2.1 (Nonparametric Representer Theorem)**.** *Given a nonempty set $\mathcal{X}$, a positive semidefinite real-valued kernel k on $\mathcal{X}\times\mathcal{X}$, a training sample $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n) \in \mathcal{X}\times\mathbb{R}$, a strictly monotonically increasing real-valued funtion g on $[0, \infty[$, an arbitrary cost function $c : (\mathcal{X} \times \mathbb{R}^2)^n \to \mathbb{R} \cup \{\infty\}$, and a class of functions*

$$\mathcal{F} = \left\{ f \in \mathbb{R}^{\mathcal{X}} | f(\cdot) = \sum_{i=1}^{\infty} \beta_i k(\cdot, z_i), \beta_i \in \mathbb{R}, z_i \in \mathcal{X}, \|f\| < \infty \right\}. \quad (2.45)$$

Here, $\|\cdot\|$ is the norm in the RKHS H_k associated with k, i.e., for any $z_i \in \mathcal{X}$, $\beta_i \in \mathbb{R}$ $(i \in \mathbb{N})$,

$$\left\| \sum_{i=1}^{\infty} \beta_i k(\cdot, z_i) \right\|^2 = \sum_{i,j=1}^{\infty} \beta_i \beta_j k(z_i, z_j). \quad (2.46)$$

Then any $f \in \mathcal{F}$ minimizing the regularized risk functional

$$c\left((x_1, y_1, f(x_1)), ..., (x_n, y_n, f(x_n))\right) + g\left(\|f\|\right) \quad (2.47)$$

admits a representation of the form

$$f(\cdot) = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i). \quad (2.48)$$

This theorem directly shows that large margin classifiers and SVR can be extended by using a kernel function. The prediction function of large margin classifiers was given in (2.38). Replacing the inner product in input space with the inner product in feature space and substituting the kernel function into it, we arrive at

$$\begin{aligned} f(x) = &\quad \text{sign}\left(\sum_{i=1}^{n} \alpha_i \langle \phi(x), \phi(x_i) \rangle + b \right) \\ = &\quad \text{sign}\left(\sum_{i=1}^{n} \alpha_i k(x, x_i) + b \right). \end{aligned} \quad (2.49)$$

Since the representer theorem tells us that the solution to the regularized risk functional admits a representation of the so-called *support vector expansion*, it is guaranteed that the optimal solution of large margin classifiers in feature space and, therefore, the prediction function exists, given the kernel is positive semidefinite. For SVR the argument is analogous.

Kernels for Real-Valued Data and Strings

In the literature there exist kernels for various different input sequences like graphs, strings, real-valued data, sets, and trees. Throughout this thesis, we consider a string $s = \{(s_1, s_2, ..., s_n)|s_i \in \mathcal{A}\}$ as a sequence of letters from a given alphabet $\mathcal{A}$. Since this work focuses on the sequence-based prediction problems, we will just go into detail for kernels on strings. A more comprehensive overview of all different kinds of kernels can be found in [115]. In many learning problems one has real-valued data. For example, this could be a number of different finance values if one wants to predict financial liability. There are also applications where strings are encoded by real values. One example is the encoding of a protein sequence by physicochemical properties [51]. In other applications where the sequences have the same length, a very common approach is to represent sequences by *sparse binary encoding*. In this encoding, each letter is represented by a vector containing as many entries as the number of letters in the alphabet. The vectors contain a one at the position which corresponds to the letter, and all other entries are zero. Standard kernels for real-valued data are:

- Polynomial kernels: $k(x, x') = \langle x, x' \rangle^d,\ d \in \mathbb{N}$
- Gaussian or Radial Basis Function (RBF) kernels: $k(x, x') = \exp -\frac{\|x-x'\|^2}{2\sigma^2},\ \sigma > 0.$

Usually a kernel contains a parameter, which is also called a hyperparameter. These parameters allow adapting the kernel to the different problems. For example, the parameter d of the polynomial kernel controls the degree of the polynomials that are considered, whereas the parameter σ of the RBF kernel controls the width of the Gaussians.
One of the first kernels introduced for strings was the spectrum kernel [66]. It uses histograms of (contiguous) substrings of a certain length p. The feature space consists of vectors with as many entries as there are different strings of length p possible, given the alphabet $\mathcal{A}$. The more substrings sequence s_i and sequence s_j have in common, the higher will be the dot product in feature space between them. Leslie *et al.* [66] showed how to efficiently calculate a kernel function, which is equal to the inner product in feature space, and applied their kernel function to the problem of remote homology detection.
There are various extensions of the spectrum kernel. One can consider, for example, all contiguous or non-contiguous subsequences of a string. This kernel is called the *all-subsequences kernel* [115]. If one fixes the length of the allowed subsequences, one gets the so-called *fixed length subsequences kernel* [115].
The string kernels we have introduced up to now did not consider the position of the signal (contiguous or non-contiguous substrings). They are not position-aware. The *locality-improved kernel* introduced by Zien *et al.* [146] does not just look at matching characters or substrings of strings, but it also takes the positions of the substrings into account. Therefore, a certain window around a position in the string is defined. Inside this window, the measure looks for matching characters, weighting matches with increasing

weights from the border to the middle. The window is shifted over the whole sequence and an even higher-dimensional feature space is constructed by taking the measure to the power of a certain value.
Another position-aware string kernel is the *weighted degree* (WD) *kernel* [112]. This kernel can be considered as a position-aware variant of the all-subsequences kernel in which the matches of different length are weighted by a certain factor corresponding to the lenght.
A further extension to the position-aware string kernels was the incorporation of positional uncertainty. This can be motivated by considering an example in which a random sequence s_1 and the same sequence shifted by one letter s_2 are compared. The locality-improved kernel as well as the WD kernel would just detect random similarities, meaning that s_1 should have a higher kernel value with a sequence s_3 containing parts of the sequence s_1 and some random characters. This is certainly not desirable. A position-aware string kernel with positional uncertainty, the so-called *oligo kernel*, was introduced in 2004 [76]. The kernel considers similarities of substrings of a certain length while the positional uncertainty is modelled by a Gaussian function around the positions where the substring occurs. The incorporation of positional uncertainty into the WD kernel was proposed in 2005 [97] by allowing patterns to be shifted by a certain amount of letters.

2.1.6 Consistency of Support Vector Machines

We already explained why large margin classifiers should generalize well to unseen data. Nevertheless, we did not show that, given enough data, the algorithm will converge to the best possible predictor. In this sense one is usually interested in consistency of the learning algorithm. Loosely speaking, this means that, given an infinite amount of data from the source, the probability that the prediction function will deviate by $\varepsilon > 0$ tends to zero. More formally, consistency is defined as:

Definition 2.3 (Consistency)**.** *Let f_t be the target function of the learning algorithm. A classifier is said to be weakly/strongly universally consistent if*

$$\lim_{n\to\infty} R(f_t) = R^* \tag{2.50}$$

holds in probability/almost surely for all distributions P on $\mathcal{X} \times \mathcal{Y}$.

Convergence in probability means that the probability that the deviation is greater than $\varepsilon > 0$ converges to zero as n goes to infinity. Almost sure convergence means that

$$P(\lim_{n\to\infty} |R(f_t) - R^*| = 0) = 1. \tag{2.51}$$

It was shown that the 1-norm soft margin classifier and the 2-norm soft margin classifier are strongly universally consistent if a universal kernel is used and the regularization parameter is chosen properly [119].

2.2 Proteomics

2.2.1 General Overview

The proteome is the set of all proteins that can be made out of the genome of an organism. Given a biological sample, one interesting question to ask is which proteins are contained therein. The first approaches to answer this question were developed by Edman and one of these methods is nowadays called Edman degradation [27]. In this technique, the protein is degraded from the N-terminus, one amino acid at a time. The identity of the removed amino acid is then determined by analytical methods like HPLC, which we will consider in more detail in Section 2.2.2. The removal and analysis of the amino acid is called a cycle. To identify the protein at least six or seven cycles are usually required to get a unique protein hit. Although the method has been improved over the years, there are some shortcomings. First of all, each cycle takes about 45 minutes [52]. This limits the number of analyzed samples per day to two or three. The second shortcoming is that there are many proteins with blocked N-termini. Consequently, these proteins cannot be identified by Edman degradation. Additionally, the sensitivity of the method is not high enough.
Protein identification based on mass spectrometry (MS) analysis has been around for more than 40 years [5]. Nevertheless, the wide application of MS-based methods for protein analysis did not start until the commercialization of electrospray ionization (ESI) and matrix-assisted laser desorption/ionization (MALDI) [53]. The importance of these methods to science was underlined by the Nobel committee, which awarded half of the 2002 Nobel prize in chemistry to the scientists who introduced these two methods. There are mainly two different approaches. One of them is called the "top-down approach". In this approach intact proteins are measured by the mass spectrometer [38]. Two-dimensional (2D) gel electrophoresis is a common method to separate the proteins before directing them to the mass spectrometer. The proteins are first separated with respect to their isoelectric point using isoelectric focusing. Afterwards, the proteins are separated according to their molecular weight along the second, orthogonal dimension. One disadvantage of gel electrophoresis is that it cannot be directly coupled to an ESI source. Instead, the proteins of interest have to be cut out of the gel manually. This intervention is not needed in a "bottom-up approach" using chromatography for separating the analytes. Typical steps in this approach are:

1. digestion of proteins into smaller parts (peptides)

2. separation of peptides according to certain properties

3. ionization of peptides

4. analysis of peptides by mass spectrometry

5. identification of peptides/proteins from mass spectrometry data.

The first step can be performed in a solution using proteolytic enzymes like trypsin or chymotrypsin. These enzymes usually cut at very distinct positions. Trypsin, e.g., cleaves after the amino acids arginine and lysine but not before a proline residue.
We will look at steps two, three, four, and five in more detail for strong anion exchange and high-performance liquid chromatography coupled to ESI MS/MS. For an introduction to MALDI, the interested reader is referred, e.g., to [52].
Although spectra can be interpreted manually, current high throughput experiments require computational methods for fast and accurate analysis of mass spectrometry data to identify and quantitate the measured proteins. These methods are introduced in section 2.2.5.

2.2.2 Chromatographic Separation

Due to the complexity of the sample, it is often beneficial to separate peptides before analyzing them by mass spectrometry. The most widely used technique for this purpose is liquid chromatography (LC), which separates the peptides according to certain properties of the peptide. With this technique, the peptides are directed through a column and, depending on properties like hydrophobicity, length, molecular mass, and amino acid composition, each peptide will elute from the column at a certain timepoint. This means that peptides with similar properties should elute at similar timepoints. We will now review the most common chromatography techniques.
In High-Performance Liquid Chromatography (HPLC), a sample is directed through a column to separate the peptides depending on specific properties. The liquid that is pumped through the column is called the *mobile phase*. The substances that are fixed to the column are part of the *stationary phase*. Usually, the stationary and the mobile phases have different chemical properties. According to the properties of the peptides, each peptide will have a stronger interaction with either the stationary or the mobile phase. If a peptide interacts stronger with the mobile phase than with the stationary phase, it will flow faster through the column than peptides that interact stronger with the stationary phase. Different combinations of stationary and mobile phases are known, but the most widely used is called reversed-phase. Therefore, reversed-phase HPLC will be explained in more detail. Strong anion exchange chromatography is also introduced, since we also analyze data obtained by this technique in this thesis.

Reversed-Phase HPLC

In reversed-phase HPLC, the stationary phase is non-polar and the mobile phase consists of an aqueous, moderately polar solution. The more hydrophobic the peptides are, the greater is the tendency of the column to retain them. Consequently, the more hydrophilic the peptides are, the faster they flow through the column.

Strong Ion Exchange Chromatography

In ion exchange chromatography, the stationary phase either contains cations or anions. In strong anion exchange (SAX) chromatography the peptides that have many positively charged side-chains interact stronger with the column. The main practical difference between strong ion exchange and reversed-phase HPLC is that strong ion exchange can just separate the peptides into different fractions (e.g., 15 fractions if coupled to a mass spectrometer on-line or 96 fractions via an off-line combination [86]), whereas peptides in reversed-phase HPLC elute at a distinct point in time.

Two-Dimensional Chromatographic Separation

To get even better separation, it is common to combine two chromatographic separations that separate the peptides according to different criteria. One possibility for a two-dimensional separation is to use strong ion exchange chromatography prior to a reversed-phase HPLC. Washburn *et al.* [139] applied this two-dimensional separation with a strong cation exchange chromatography followed by a reversed-phase chromatography to perform a large-scale proteome analysis. This technique is called MudPIT and is based on work by Link *et al.* [69]. Very recently, Delmotte *et al.* [20] introduced a combination of two reversed-phase HPLC separations at different pH values.

2.2.3 Ionization

Electrospray ionization (ESI) was introduced in 1985 by Fenn and co-workers [141]. This technique can be used to ionize peptides in the solvent phase and bring them into the gas-phase. A schematic illustration of ESI is shown in Fig. 2.11. A watery, acidic solution, which contains peptides, is sprayed through a very thin needle. The high positive voltage at the tip of the needle leads to sputtering of droplets. A negative voltage is applied to the mass spectrometer. Therefore, the positively charged ions travel towards the mass spectrometer. Since the ions travel through a heated near-vacuum region, the ions get desolvated, which finally leads to protonated peptides in gas-phase.

2.2.4 Tandem Mass Spectrometry

Tandem mass spectrometry or MS/MS usually refers to the analysis of a sample using two mass spectrometers consecutively. With just one mass spectrometer only the mass-to-charge ratio (m/z) of a peptide can be measured. Since one cannot distinguish sequences with the same amino acid composition from each other by this information alone, a single mass spectrometer does not suffice to identify peptide samples with high accuracy. In MS/MS, there is a collision chamber between the two mass spectrometers. In this collision chamber there is an inert gas like argon or helium. When the peptide flies through the chamber, it collides with these inert gas atoms/molecules and breaks apart (fragmentation). For collision-induced dissociation (CID) the peptide usually breaks at an amide bond. If the charge is retained at the N-terminus, the corresponding ion is called a *b-ion* and if the charge is

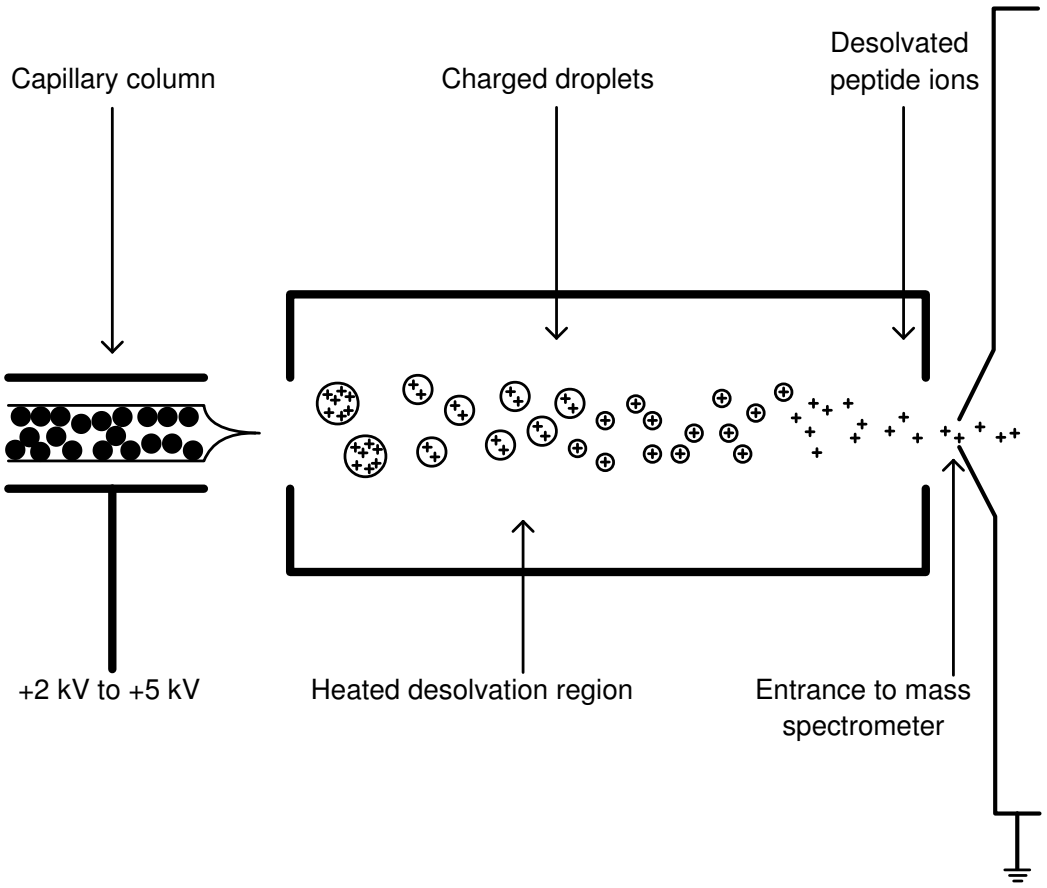

Figure 2.11: ***Electrospray ionization:*** *The sample is directed through a capillary column in an acidic solution. At the tip of the needle, a high voltage is applied. Positively charged droplets form, which are directed towards the entrance of the mass spectrometer. During the flight through the heated, near-vacuum region, the droplets are desolvated (adapted from [54]).*

retained at the C-terminal part, the ion is called a *y-ion*. A peptide together with the b- and y-ions can be seen in Fig. 2.12. The whole measurement process can be seen in Fig. 2.13. The ion which flies through the first mass spectrometer is usually called a *precursor ion* and the b- and y-ions of the precursor ion are called product ions. Usually the three highest peaks in an MS1 spectrum are selected for further fragmentation. These peaks are found by so-called survey scans, which scan a certain mass-to-charge range. A survey scan together with the product ion spectrum of the most intense precursor peak are shown in Fig. 2.14.

Two of the most prominent mass spectrometers are the quadrupole and time-of-flight (TOF) types. In quadrupole mass spectrometers, like in Fig. 2.13, only ions with a certain m/z value (± a certain tolerance) can travel through the electrostatic field on a stable path. All other ions collide with the rods and do not reach the detector. To measure the whole sample, the whole range of possible m/z values is probed from lowest to highest.
In TOF mass spectrometers, the principle is simpler. The ions are accelerated towards the detector via an electric field. Then, they travel through a field-free region. The higher the m/z value of the ion, the slower it will travel

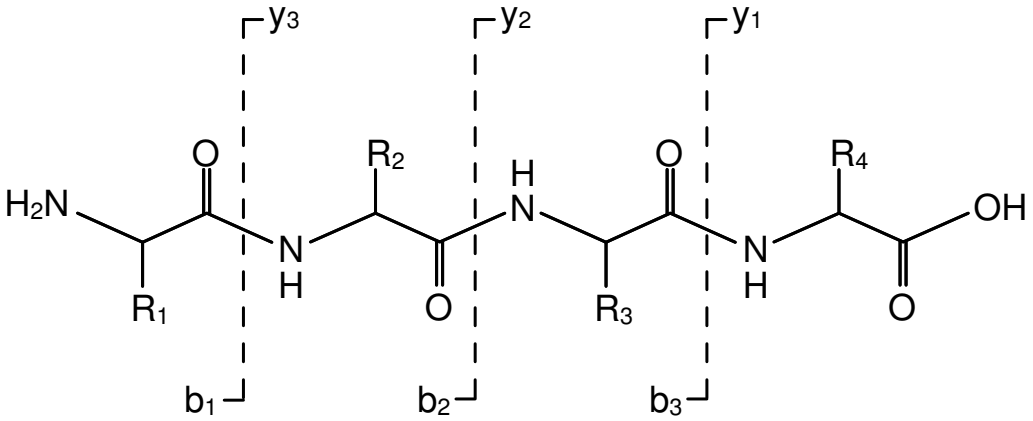

Figure 2.12: ***Peptide with b- and y-Ions:*** *If the peptide ion breaks at an amide bond, the resulting ions are called b- and y-ions.*

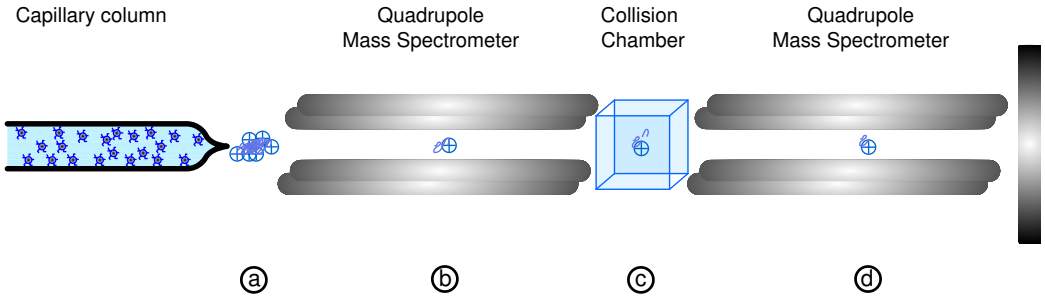

Figure 2.13: ***Simplified overview of an MS/MS experiment:*** *a) The peptides leave the capillary column and are protonated by ESI; b) Only the peptides which match the selected m/z value can travel on a stable path through the quadrupole mass spectrometer; c) In the collision chamber the peptide fragments; d) The peptide fragments travel through the second quadrupole mass spectrometer, but only the ions which match the selected m/z value can travel on a stable path and finally reach the detector.*

through the analyzer. Therefore, the m/z can be calculated given the length of the travel and the time the ion needed between entering the field-free region and hitting the detector.

2.2.5 Computational Annotation of Tandem Mass Spectra

De Novo Identification and Database Search Methods

A tandem mass spectrum contains the product ions of a particular precursor ion. If the spectra contained all possible y-ions and all ions had the same charge, the peptide sequence could be constructed easily by transforming the identification problem into a graph problem. Every m/z value in this graph corresponds to an m/z value of one of the ions. A node at value zero (source) and a node at the m/z value of the precursor ion (sink) are added. Nodes are connected by an edge if the m/z difference corresponds to the m/z value of an amino acid with the same charge. The sequence can then be constructed

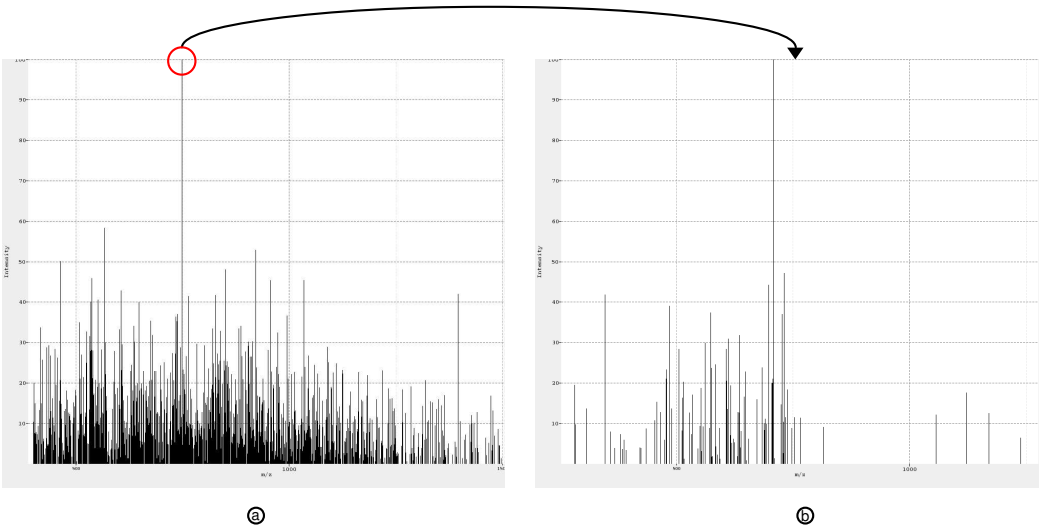

Figure 2.14: ***MS spectrum of potential precursor ions and product ion spectrum of selected precursor ion:*** *The graphs plot m/z value against intensity, given in percentage, according to the peak intensity of the most intense peak. a) An MS scan of all ions at a certain retention time. The ion with the highest intensity is chosen for fragmentation. b) The product ion spectrum of the selected precursor ion. The graphs are made with TOPPView [123].*

by finding the longest path. This kind of identification is usually called de novo identification or de novo sequencing [127, 31].
Unfortunately, many b- and y-ions in a spectrum are missing and with instruments which do not have good precision the charge state cannot be determined accurately. Consequently, there are other approaches for identification which are not that sensitive to spectra quality. One big class of methods can be called *database search methods*. In these approaches, the experimental spectra are compared to theoretical spectra which are constructed from all possible peptides of the protein database. One example of an experimental spectrum together with its theoretical spectrum (only y-ions) is shown in Fig. 2.15. Various methods for scoring and assessing the significance of these matches have been introduced. Among the first introduced methods were SEQUEST [29] and Mascot [87], but still nowadays there is room for improvement of database search methods since a significant number of spectra remain unidentified [12].

Perkins *et al.* [87] did not reveal the details of the Mascot search engine because it is proprietary software. Nevertheless, they stated that Mascot uses probability-based scoring. This means that for each match between an experimental and a theoretical spectrum the probability that a match with the same or better score occurs by chance is estimated. For each experimental spectrum, the match with the lowest probability is considered the best possible identification. The peptide sequence of the corresponding theoretical spectrum is returned as annotation of the spectrum. Furthermore, the significance of the best identification can be assessed via a p-value. To calculate probabilities one has to assume a certain distribution underlying random matches. Since Perkins *et al.* did not state which distribution un-

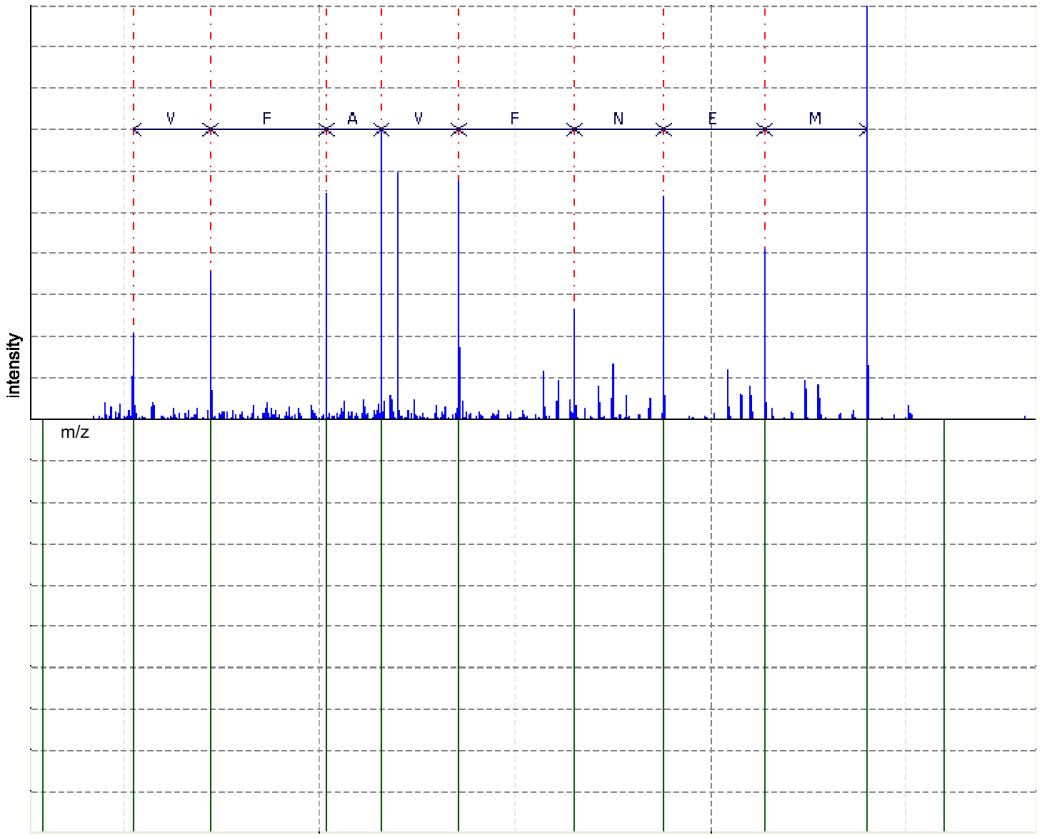

Figure 2.15: ***Experimental spectrum together with theoretical spectrum:*** *An experimental spectrum (top) is shown together with the best matching theoretical spectrum (bottom) visualized with TOPPView [123]. The peptide sequence is TVMENFVAFVDK.*

derlies the Mascot model, we look at a database search method by Sadygov and Yates [104] called PEP_PROBE. This method uses the hypergeometric distribution to model the frequencies of matches between experimental and theoretical spectra. It is a modified version of SEQUEST [29].
Let m be the number of red balls and N the total number of balls. The hypergeometric distribution can be used to estimate the probability that after n draws without replacement we end up with exactly k red balls:

$$P(X = k) = \frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}} \tag{2.52}$$

In PEP_PROBE, N is the number of all predicted fragment ions in a sequence database that consists of all peptide sequences matching the precursor mass $(M+H)^+$ of the tandem mass spectrum. The variable m represents the number of all of these fragments that match a peak in an experimental tandem mass spectrum. If one considers just b- and y-ions, the number of draws for a peptide sequence of length L is $n = 2\,(L-1)$. The hypergeometric distribution can then be used to estimate the probability that k of the fragments of the peptide sequence match the experimental spectrum just by chance.

The peptide sequence with the lowest probability is the resulting annotation of the experimental spectrum. Furthermore, PEP_PROBE delivers a p-value for the null hypothesis that the hit is random.

False Discovery Rates and q-Values

Since not every database search method uses the same underlying distribution and some models do not even provide p-values, other measures for assessing the significance of identifications have been introduced [28]. One very common measure is estimating the *false discovery rate* (FDR) of an identification. Given hypotheses i with associated score s_i, the *false discovery rate* [120] is defined as the expected ratio of the number of false hypotheses to the number of all hypotheses that are considered significant at threshold t:

$$\mathrm{FDR}(t) = E\left[\frac{F(t)}{S(t)}\right], \tag{2.53}$$

with $F(t)$ being the number of all significant false hypotheses and $S(t)$ the number of all significant hypotheses at threshold t. Since it is usually impossible to estimate this expectation value, the following approximation is widely used:

$$\mathrm{FDR}(t) = E\left[\frac{F(t)}{S(t)}\right] \approx \frac{E\left(F(t)\right)}{E\left(S(t)\right)}. \tag{2.54}$$

For identifications from tandem mass spectrometry experiments, the numerator is estimated by using the database search method on a decoy database. This decoy database contains bogus protein sequences. These can be constructed by reversing or shuffling the sequences of a normal database. The denominator can be estimated by using the database search method on a normal database [48]. Let $d_1, d_2, ..., d_n$ be the scores of spectrum identifications to the decoy database and let $s_1, s_2, ..., s_n$ be the scores of the spectrum identifications to the normal database. Without loss of generality, we assume that a hypothesis is the more significant the larger its corresponding score is. The FDR of a certain score threshold t can then be approximated by

$$\widehat{\mathrm{FDR}}(t) = \frac{\#\{d_i | d_i \geq t \wedge i \geq 1 \wedge i \leq n\}}{\#\{s_i | s_i \geq t \wedge i \geq 1 \wedge i \leq n\}}. \tag{2.55}$$

Since the FDR can be smaller for hypothesis i with score s_i than for hypothesis j with score s_j even though $s_i < s_j$ (e.g., if all hypotheses with score larger than s_i and smaller than s_j are true), the FDRs cannot directly be used as filter thresholds. Storey and Tibshirani [120] introduced q-values.

A q-value for hypothesis i is the minimum FDR that can be attained when calling hypothesis i significant ($s_i \geq t$). Käll *et al.* [48] used q-values for assigning significance to identifications from tandem mass sectrometry data. In this setting, the q-value of a certain spectrum identification is the smallest FDR at which the spectrum identification is accepted. Therefore,

$$q(i) = \min_{t \leq s_i} \widehat{\mathrm{FDR}}(t). \tag{2.56}$$

2.3 Immunomics

2.3.1 General Overview

The latin word *immunis* can be translated as *exempt*. This word was chosen as a basis for immunity and immunology because the whole field emerged from the observation that people who had recovered from particular infectious diseases were afterwards exempt from falling ill to the disease. These people were *immune* to the corresponding disease. From then on, the goal of doctors and scientists in this field has been to make people immune to a certain disease even before a first infection. The first disease for which reports exist was the smallpox disease. In the fifteenth century, the Chinese and Turks tried to immunize people with the dried crusts of smallpox pustules. In 1798, Edward Jenner made the observation that milkmaids who fell ill with cowpox and subsequently recovered were later immune to smallpox. Therefore, he used the fluid of a cowpox pustule to inoculate a person and he showed that the person was then immune to smallpox. Between this discovery and the current state of immunological knowledge there were many ground-breaking studies, which can also be seen by the fact that 16 Nobel prizes had been awarded for immunologic research by 1996 [100].
There are different mechanisms of the body which lead to immunity to a certain infectious agent and they can be divided into two categories called *Innate Immunity* and *Adaptive Immunity*. Innate immunity contains non-adaptive barriers to infectious agents. There are four different types of barriers: anatomic barriers, physiological barriers, phagocytic/endocytic barriers, and inflammatory barriers. In contrast, adaptive immunity comprises all defense mechamisms which are able to adaptively recognize and destroy specific agents.

2.3.2 Innate Immune System

The innate immune system can be seen as the first line of defense against invading microorganisms. The underlying mechanisms prevent a large class of microorganisms from entering or staying in the human body. The first part of this system comprises the anatomic barriers. The skin, which is made up of the epidermis and the dermis, is the outer barrier. The epidermis contains dead cells as well as the protein keratin, which makes this layer waterproof. The dermis is composed of connective tissue. Because of the low pH, between 3 and 5 in this tissue, it inhibits the growth of most infectious agents.
Another important anatomic barrier is the mucosal surface. It can be found on the mucosal membranes of the alimentary, urogenital, and respiratory tracts as well as on the mucosal membrane of the conjunctivae. Since these membranes are easier for the microorganisms to penetrate, there exist a number of non-specific defense mechanisms that serve to remove the invading microorganism from the body. These defense mechanisms are, e.g., tears, saliva and mucous secretion, and hairlike protrusions called cilia.
The second part of the innate immune system comprises the physiological barriers. All physiological conditions in the body which inhibit the growth

of, or destroy microorganisms, can be seen as manifestations of these barriers. The temperature of the human body, for example, inhibits the growth of certain organisms and the low pH in the stomach leads to the destruction of many microorganisms. Furthermore, there are chemical mediators like lysozyme, complement, and interferon, which favor the lysis of certain pathogens or facilitate phagocytosis.
The third part of the innate immune system comprises the phagocytic and endocytic barriers. Some specialized cells in the body (e.g., monocytes, neutrophils, and macrophages) are able to phagocytose foreign organisms. This means that the cells surround the pathogens to internalize them in so-called phagosomes. The content of these phagosomes is then digested by lysosomal enzymes and the products of this reaction are released from the cell. Phagocytosis is a special form of endocytosis which describes the uptake of material from its surrounding. A certain type of endocytosis, which is called receptor-mediated endocytosis, allows a cell to specifically uptake certain extracellular molecules after they have been bound by the corresponding receptors.
The fourth part of the innate immune system comprises the inflammatory barrier. If a tissue is ruptured, usually an inflammatory response is triggered, in which vascular fluid, containing serum proteins with antibacterial activity as well as phagocytic cells, are released into the affected region. After the activation of an enzyme cascade, insoluble fibrin strands separate the ruptured area from the rest of the body to prevent further microorganisms from entering.

2.3.3 Adaptive Immune System

The adaptive immune system can very selectively recognize and eliminate invading microorganisms. There are many cells and receptors involved in this adaptive response. For such a system to work there are a few requirements. First of all, there has to be a mechanism to discriminate self from non-self cells. Since invading microorganisms can be quite diverse, there has to be a mechanism by which the immune system can construct very diverse but also specific cells which recognize the pathogens. To be able to react to an infectious agent as fast as possible, it is also desirable to have a sort of memory which enables a faster reaction if the person is infected by an infectious agent against which it has already reacted. The adaptive immune system fulfills all these requirements. We will now look at parts of the system in more detail.
The term *antigen* will be used quite frequently in this thesis. When it was first introduced in the literature, it stood for any substance which could stimulate *anti*body *gen*eration. Nowadays, the term is used for any substance which can be recognized by the adaptive immune system.

Major Histocompatibility Complex

The major histocompatibility complex (MHC) is a cluster of genes whose encoded proteins are responsible for many important parts of the adaptive immune system of mammals. In humans, the MHC is referred to as the hu-

man leukocyte antigen (HLA) complex. T cells, which are introduced in the next section, can only recognize antigens that are presented by MHC class I (MHCI) or MHC class II (MHCII) molecules. Throughout this thesis we use *MHCI molecule* (sometimes also *MHCI allele*) as a synonym for the gene products HLA-A, HLA-B, and HLA-C encoded by the HLA complex regions A, B, and C. We use the term *MHCII molecule* (sometimes also *MHCII allele*) as a synonym for the gene products HLA-DP, HLA-DQ, and HLA-DR encoded by the HLA complex regions DP, DQ, and DR. MHCI molecules can be found on nearly all nucleated cells. In contrast, MHCII molecules can just be found on antigen-presenting cells such as B cells, dendritic cells, and macrophages. The structures of the MHCI and MHCII molecules are very similar. Both have a binding cleft in which about nine amino acids can fit. This enables peptides to interact with and therefore bind to the molecule. Examples of MHCI and MHCII molecules can be seen in Fig. 2.16. The main difference between these two classes of molecules is that the binding cleft is closed at the ends for MHCI and open for MHCII. Therefore, MHCI

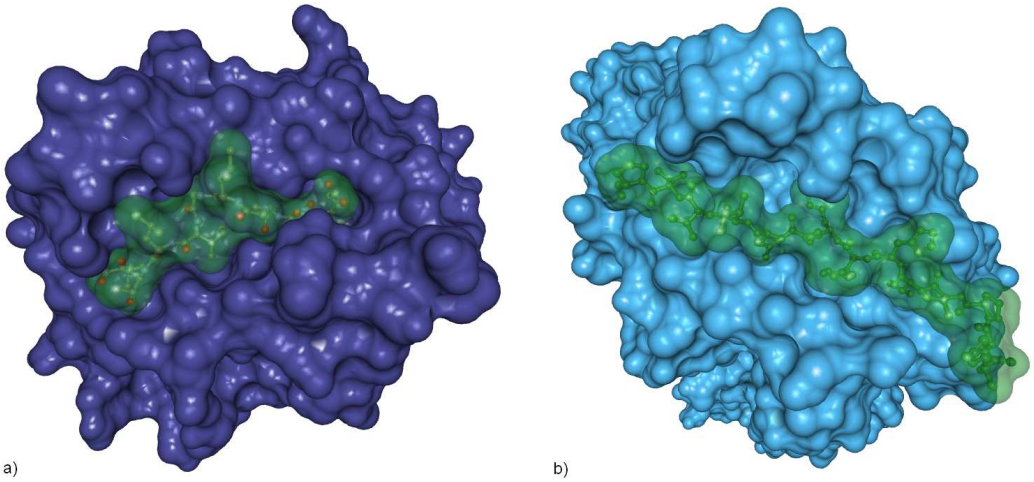

Figure 2.16: ***Structure of MHC molecules with binding peptide:*** *a) This picture shows an MHCI molecule (purple) together with a binding peptide (green). The PDB ID of the structure is 1JF1. b) This picture shows an MHCII molecule (blue) together with a binding peptide (green). The PDB ID of the structure is 1BX2. Both MHC molecules were visualized with BALLView [77].*

molecules can just bind peptides of a narrowly defined length. The binding peptides are usually between eight and twelve amino acids long. In contrast, the peptides that can bind to MHCII molecules can have even more than twenty amino acids [16].
MHCI molecules present peptides which are derived from proteins inside the cell. In contrast, MHCII molecules present peptides derived from proteins outside of the cell. The proteins enter the cell via the endosomal pathway. Inside the cell, they are exposed to several proteases, which cut them into smaller parts (peptides). The peptides are then transported to a compartment known as the MIIC (MHCII-rich endosomal compartment). In this compartment, the peptides are loaded into MHCII molecules. Afterwards, the peptide-MHCII complex is transported to the cell surface.

Every human expresses at most six different types of MHCI molecules and twelve different types of MHCII molecules [101].

T Lymphocytes

T lymphocytes or T cells are cells which arise in the bone marrow and mature in the thymus gland (hence the name). T cells express a unique antigen-binding receptor, called the T cell receptor, which is not able to recognize an unbound antigen. Instead, it recognizes antigens bound to MHC molecules. T cells can be divided into two groups, namely T helper (T_H) and T cytotoxic (T_C) cells. They can be distinguished by the existence of glycoproteins on the cell surface. T_H cells express CD4 and T_C cells express the glycoprotein CD8. Examples of both kinds of T cells are depicted in Fig. 2.17. If T cells recognize an antigen, they get activated. An activated T_H cell releases cytokines, which are important for the activation of T_C cells, B cells, macrophages, and a number of other cells that belong to the immune system. An activated T_C cell differentiates into a cytotoxic T lymphocyte (CTL), which exhibits cytotoxic activity. The whole process is described in more detail in Section 2.3.3.

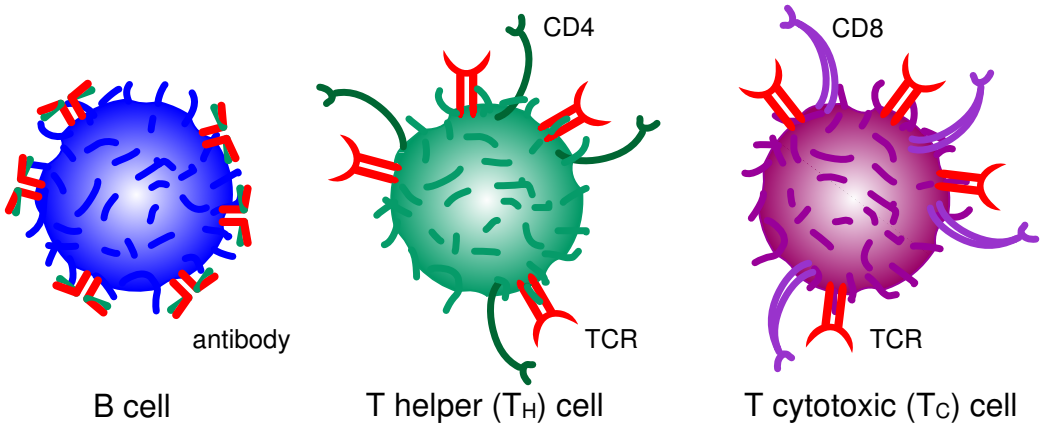

Figure 2.17: ***Lymphocytes:*** *This figure shows different types of lymphocytes. From left to right, the figure shows a B cell, a T_H cell, and a T_C cell. The B cell expresses membrane-bound antibodies. The T cells are depicted with a T cell receptor (TCR), the T_H cell additionally contains CD4, and the T_C cell contains CD8.*

B Lymphocytes

B Lymphocytes or B cells arise in the bone marrow like T cells but unlike T cells they stay in the bone marrow for maturation (hence the name). As shown in Fig. 2.17, B cells express an antigen-binding receptor, which is very specific. Each B cell expresses several copies of the same receptor, which is a membrane-bound antibody, when it leaves the bone marrow. If a B cell encounters an antigen to which the membrane-bound antibodies can bind, it starts dividing rapidly. The resulting cells differentiate into memory B cells and plasma cells. Memory B cells live longer than standard B cells still having the same membrane-bound antibody. Plasma cells usually express

little or no membrane-bound antibodies. They are able to produce copious numbers of the antibody and secrete them. The whole process is described in more detail in Section 2.3.3.

Antibody Generation

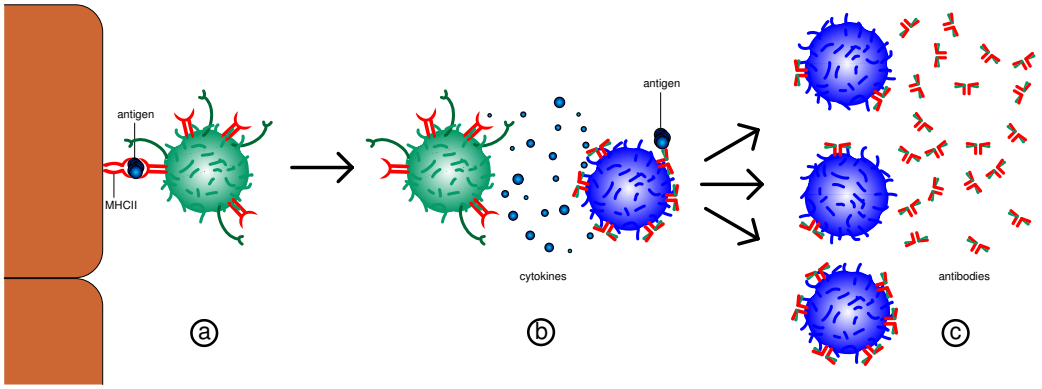

Figure 2.18: ***Antibody production after recognition of antigen:*** *a) A naive T_H cell gets activated after recognizing an MHCII-bound antigen. b) After activation it releases cytokines. The cytokines are absorbed by nearby B cells. c) If one of these B cells recognizes an antigen, it divides into antibody-producing plasma cells and memory B cells.*

The humoral response to antigens is shown in Fig. 2.18. If a naive T_H cell encounters an antigen which is presented by an MHCII molecule, the T_H cell gets activated. The activation leads to the release of cytokines. A B cell, which absorbs these cytokines and recognizes an antigen via one of its membrane-bound antibodies, divides into plasma cells and memory B cells. The plasma cells produce a huge amount of the antibody and secrete it. Usually these plasma cells have little or no membrane-bound antibodies. Since the antibodies can bind to the antigen and these antigens are likely to be proteins that are expressed on the exterior of bacteria or viruses, the existence of a large number of these antibodies promotes the clearance of the infectious agents.

CTL Response to Antigens

The cell-mediated response to an antigen is shown in Fig. 2.19. One requirement for a T_C cell to differentiate into a CTL is that it encounter cytokines. These are produced with the same mechanisms as described in Section 2.3.3. If a CTL encounters an antigen presented at the cell wall of an infected or altered cell via an MHCI molecule, the CTL induces apoptosis in the corresponding cell. Since MHCI molecules present peptides, which are derived from proteins inside the cell, the recognition of a foreign peptide signals that either the cell is infected by an infectious agent or the cell has altered protein sequences, which happens, e.g., in cancer. Therefore, it can be assumed that mechanisms which promote apoptosis of these kinds of cells proved to be advantageous during evolution.

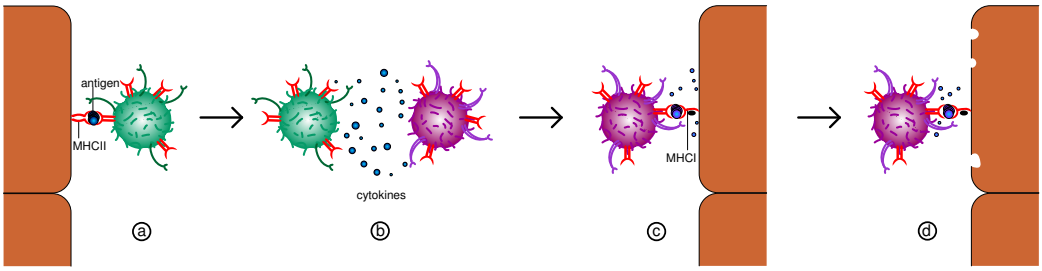

Figure 2.19: ***Cell-mediated response to antigens:*** *a) A naive T_H cell gets activated after recognizing an MHCII-bound antigen. b) After activation it releases cytokines, which are absorbed by nearby T_C cells. This contributes to the differentiation of T_C cells into CTLs. c) A CTL, which recognizes an antigen presented by an MHCI molecule, releases signalling molecules which induce apoptosis. d) The cell wall of the infected or altered cell evaporates.*

2.3.4 Epitope-Based Vaccine Design

We have already defined an antigen as a substance, which can be recognized by the immune system. In many cases, not the whole substance is recognized. Instead, just a smaller part interacts with B or T cells. For example, if a protein of the cell wall of a bacterium can be recognized by the immune system, this protein is termed an antigen. The peptides from this protein that can bind to the B cell or T cell receptors are called *antigenic determinants* or *epitopes*. As mentioned in Section 2.3.1, the traditional approach of vaccination is to expose an individual to a non-pathogenic form of the pathogen. The vaccination strategy in *epitope-based vaccine design* [114] is a little bit different. In this approach, the vaccine just contains a set of epitopes.
T cells can only recognize an epitope if it is bound to an MHC molecule. One prerequisite for the rational design of an epitope is, thus, that these epitopes are able to bind to MHC molecules. Since every human has at most six different MHCI molecules and twelve different MHCII molecules [101], it is important to know which peptides can bind to the MHC molecules of the patient. Several databases exist that contain data from binding studies, measuring whether a peptide can bind to a certain MHC molecule [89], but these databases just contain data for a small fraction of all known MHC molecules [79]. Therefore, many approaches for peptide-MHC binding prediction have been introduced (class I and class II). Two recent benchmarks of available methods can be found in [88, 138].
Many epitope identification studies have been performed [114]. Providing researchers with good predictors for peptide-MHC binding can significantly reduce the number of necessary experiments. If epitopes are known for a certain pathogen, there are different approaches for how to combine them into a vaccine [131], but there are still many open questions in this field (e.g., how to best deliver the epitopes, how many epitopes should be used).

Chapter 3

Applications in Proteomics

3.1 A New Kernel for Chromatographic Separation Prediction

3.1.1 Introduction

Experimental techniques for determining the composition of complex proteomes have been improving rapidly over the past decade. The application of tandem mass spectrometry-based identification has resulted in the generation of enormous amounts of data, requiring efficient computational methods for their evaluation. There are numerous database search algorithms for protein identification such as Mascot [87], Sequest [29], OMSSA [34] and X!Tandem [18], as well as *de novo* methods like Lutefisk [127] and PepNovo [31]. Furthermore, there are a few methods like InsPecT [32] which use sequence tags for pruning the possible search space using more computationally expensive and more accurate scoring functions afterwards. Database search algorithms generally construct theoretical spectra for a set of possible peptides and try to match these theoretical spectra to the measured ones to find the candidate(s) which match(es) best. In order to distinguish between true and random hits, it is necessary to define a scoring threshold, which eliminates all peptide identifications with scores below the scoring threshold. This threshold value is chosen quite conservatively to get very few false positives. Consequently, there is a significant number of correct identifications below the threshold that are not taken into account, although these spectra often correspond to interesting (e.g., low abundance) proteins. One of the goals of this work was to increase the number of reliable identifications by filtering out false positives in this 'twilight zone' below the typical threshold. There are various studies addressing this issue [26, 72, 78] by calculating the probability that an identification is a false positive.
Standard identification algorithms are based on MS/MS data and do not use the information inherent to the separation processes typically used prior to mass spectrometric investigation. Since this additional experimental information can be compared to predicted properties of the peptide hits suggested by MS/MS identification, false positive identifications can be detected. In SAX-SPE, it is important to know whether a peptide binds to the column or

flows through. This information can also be incorporated into the identification process to filter out false positive identifications. Oh *et al.* [84] elaborated several chemical features such as molecular mass, charge, length and a so-called sequence index of the peptides. These features were subsequently used in an artificial neural network approach to predict whether a peptide binds to the SAX column or not. The sequence index is a feature reflecting the correlation of pI values of consecutive residues. Strittmater *et al.* [121] included the experimental retention time from an ion-pair reversed-phase liquid chromatographic separation process into a peptide scoring function. They used a retention time predictor based on an artificial neural network [90] but a number of other retention time predictors exist [91, 35]. If the deviation between observed and predicted retention time is large, then the score of the scoring function becomes small. Since they only consider the top scoring identifications (rank = 1), they miss correct identifications of spectra where a false positive identification has a larger score than the correct one. We also address these cases in our work, demonstrating that filtering out identifications with a large deviation between observed and predicted retention time significantly improves the classification rate of identifications with small maximal scores. Recently, Klammer *et al.* [55] used support vector machines (SVMs) [11] to predict peptide retention times. Nevertheless, they used standard kernel functions and stated that they needed at least 200 identified spectra with high scores to train the learning machine.

When applying machine learning techniques to the prediction of chromatographic retention, a concise and meaningful incorporation of the peptide properties is crucial. The features used for this incorporation must capture the essential properties of the interaction of the peptide with the stationary and the mobile phases. These properties are mostly determined by the overall amino acid composition, by the sequence of the N- and C-terminal ends, and by the sequence in general. One of the most widely applied machine learning techniques are SVMs, introduced in Section 2.1.4. SVMs use a *kernel function* which is used to encode distances between individual data points (in our case, the peptides). There are numerous kernel functions described in the literature which can be applied to sequence data. An overview is presented in Section 2.1.5. All of these kernels were either introduced for sequences of the same length or not position-aware. However, the length of peptides typically encountered in computational proteomics experiments varies significantly, ranging roughly from 4 to 40 amino acids. Because it can be assumed that the local alignment kernel [136], which can also handle sequences of different lengths, does not suit this kind of problem perfectly, we propose a new kernel function, which can be applied to sequences of different lengths. Consequently, this new kernel function is applicable to a wide range of computational proteomics applications.

In 2006 Petritis *et al.* [91] evaluated different features like peptide length, sequence, hydrophobicity, hydrophobic moment and predicted structural arrangements like helix, sheet or coil for the prediction of peptide retention times in reversed-phase liquid chromatography-MS. They used an artificial neural network and showed that the sequence information, together with sequence length and hydrophobic moment yield the best prediction results.

In their study, they used only the border residues of the peptide sequences; their evaluation showed that a border length of 25 worked best for their dataset. Since they used one input node for every position of the borders of the peptide, they needed a very large training set. They trained their learning machine on 344,611 peptide sequences.
Since one cannot routinely measure such an amount of training sequences before starting the actual measurements, it is reasonable to apply a sort of Gaussian smoothing effect to the sequence positions. This means that in our representation, not every amino acid at every position is considered but rather regions (consecutive sequence positions) where the amino acid occurs. The distance of the amino acids of two sequences is scored with a Gaussian function. The size of this region modeled by our kernel function can be controlled by the kernel parameter σ of the kernel function and can be found by cross validation. By this and because we use support vector machines in combination with our kernel function, the number of necessary training sequences can be decreased dramatically. By just using the amino acid sequence, we do not rely on features which are important for certain separation processes. This means that we learn the features (e.g., composition (using a large σ in the kernel function), sequence length, hydrophobic regions) which are important for the prediction process within the data because they are reflected in the amino acid sequence. This is why our kernel function can be used for retention time prediction in IP-RP-HPLC as well as for fractionation prediction in SAX-SPE.
When applied to the same dataset as Oh *et al.* [84] used, our kernel function in conjunction with support vector classification predicts 87% of the peptides correctly. This is better than for all reported methods. Furthermore, our retention time prediction model is based on a new kernel function in conjunction with support vector regression [110], which allows to predict peptide retention times very accurately, requiring only a very small amount of training data. This method has a better performance on a test set than the artificial neural network method used by Strittmater *et al.* [121], even with a much smaller training set. Additionally, our method outperforms the methods introduced by Klammer *et al.* [55]. Section 3.1.2 describes our new kernel function and we explain our p-value-based filtering approach. Section 3.1.3 introduces the datasets used in this study. In Section 3.1.4, we demonstrate that our new kernel function, in combination with support vector classification, achieves better results in SAX-SPE fractionation prediction than any other published method. Next, we show that our kernel function also performs very well for peptide retention time prediction in IP-RP-HPLC with very little training data required. This allows us to train our predictor on a small dataset to predict retention times for further datasets, and to filter the data by deviation in observed and predicted retention time. This leads to a huge improvement in the precision of the identifications of spectra for which only identifications with small scores can be found, and also improves the precision of high-scoring identifications.

3.1.2 Machine Learning Methods

In this thesis, we introduce a new kernel function, which can be used to predict peptide properties using support vector classification and ν-support vector regression [110], introduced in Section 2.1.4. We apply this kernel function to predict fractionation of peptides in SAX-SPE as well as peptide retention times in IP-RP-HPLC. To show the superior performance of the new kernel function, we provide comparisons to established kernel functions and the latest approaches of other working groups [84, 91, 55].

Kernel Function

The oligo kernel introduced by Meinicke *et al.* [76] is a kernel function that can be used to find signals in sequences for which the degree of positional uncertainty can be controlled by the factor σ of the kernel function. The standard oligo kernel was introduced for sequences of fixed length. Since there are many problems in which the length of the sequences varies significantly (e.g., peptide retention time prediction), this kernel function cannot be applied to them directly.
Petritis *et al.* [91] predicted peptide retention times very accurately by encoding the border residues directly, meaning that they accounted for 25 amino acids from each border (starting at the termini). This led to a very large neural network, which was therefore trained with about 345,000 peptides. As stated in [43], the oligo kernel can be used as a motif kernel. Therefore, one can focus on important signals instead of using all k-mers of a sequence. This motivated us to construct a kernel which only considers the border residues of a peptide for a fixed border length b. Consequently, the kernel function is called *oligo-border kernel* (*OBK*). Here, a motif is a certain k-mer at a position inside the b residue border at each side where $b \in \{1, \ldots, 30\}$. This means that every k-mer at the leftmost b residues contributes to its oligo function as well as every k-mer at the rightmost b ones. For the peptide sequence $s \in \mathcal{A}^n$, the left border L is defined as $L = \{1, 2, \ldots, \min(n, b)\}$ and $R = \{\max(0, n - b + 1), \ldots, n\}$. The set $S_\omega^L = \{p_1, p_2, \ldots\}$ contains the positions where the k-mer $\omega \in \mathcal{A}^k$ occurs inside the left border and $S_\omega^R = \{p_1, p_2, \ldots\}$ the k-mer positions for the right border. This means that $S_\omega^L \cap L = S_\omega^L$ and $S_\omega^R \cap R = S_\omega^R$. In [76] the feature space representation of a sequence is a vector containing all of its oligo functions. These oligo functions are the sums of gaussians for every particular k-mer:

$$\mu_\omega(t) = \sum_{p \in S_\omega} e^{-\frac{(t-p)^2}{2\sigma^2}}. \tag{3.1}$$

Consequently, the *oligo-border* function is:

$$\mu_\omega^M(t) = \sum_{p \in S_\omega^M} e^{-\frac{(t-p)^2}{2\sigma^2}}, \tag{3.2}$$

where $M \in \{L, R\}$. This leads directly to the feature map:

$$\Phi(s) = [\mu_{\omega_1}^L(t), \ldots, \mu_{\omega_{|\mathcal{A}^k|}}^L(t), \mu_{\omega_1}^R(t), \ldots, \mu_{\omega_{|\mathcal{A}^k|}}^R(t)]^T \tag{3.3}$$

Let $U = L \cup R$ and let $S_\omega^{U_i}$ be the set S_ω^U of sequence s_i. Let

$$\text{ind}(p,q) = [[(p \in L_i \wedge q \in L_j) | (p \in R_i \wedge q \in R_j)]] \tag{3.4}$$

for $p \in U_i$ and $q \in U_j$ in which $[[condition]]$ is the indicator function. This function equals one if *condition* is true and zero otherwise. Similar to [76] one can derive the kernel function:

$$\begin{aligned}
\langle \Phi(s_i), \Phi(s_j) \rangle = & \begin{bmatrix} \mu_{\omega_1}^{L_i}(t) \\ \cdot \\ \cdot \\ \cdot \\ \mu_{\omega_{|\mathcal{A}^k|}}^{L_i}(t) \\ \mu_{\omega_1}^{R_i}(t) \\ \cdot \\ \cdot \\ \cdot \\ \mu_{\omega_{|\mathcal{A}^k|}}^{R_i}(t)]^T \end{bmatrix} \cdot \begin{bmatrix} \mu_{\omega_1}^{L_j}(t) \\ \cdot \\ \cdot \\ \cdot \\ \mu_{\omega_{|\mathcal{A}^k|}}^{L_j}(t) \\ \mu_{\omega_1}^{R_j}(t) \\ \cdot \\ \cdot \\ \cdot \\ \mu_{\omega_{|\mathcal{A}^k|}}^{R_j}(t)]^T \end{bmatrix} && (3.5) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{(t-p)^2}{2\sigma^2}} \cdot e^{-\frac{(t-q)^2}{2\sigma^2}} dt && (3.6) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{(t-0)^2}{2\sigma^2}} \cdot e^{-\frac{(t-u)^2}{2\sigma^2}} dt && (3.7) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{t^2}{2\sigma^2} - \frac{(t-u)^2}{2\sigma^2}} dt && (3.8) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{t^2 + (t-u)^2}{2\sigma^2}} dt && (3.9) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{2t^2 - 2tu + u^2}{2\sigma^2}} dt && (3.10) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{t^2 - tu + \frac{u^2}{4} + \frac{u^2}{4}}{\sigma^2}} dt && (3.11) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot \int e^{-\frac{(t-\frac{u}{2})^2}{\sigma^2}} \cdot e^{-\frac{u^2}{4\sigma^2}} dt && (3.12) \\
= & \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot e^{-\frac{u^2}{4\sigma^2}} \cdot \int e^{-\frac{(t-\frac{u}{2})^2}{\sigma^2}} dt && (3.13) \\
= & \sqrt{\pi}\sigma \sum_{\omega \in \mathcal{A}^k} \sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \text{ind}(p,q) \cdot e^{-\frac{(p-q)^2}{4\sigma^2}} && (3.14) \\
=: & k_{OBK}(s_i, s_j) && (3.15)
\end{aligned}$$

From (3.6) to (3.7), we shifted both gaussians by $\min(p,q)$ to the left and defined $u = |p-q|$ and in (3.13) we used the fact that $\int e^{-\frac{(t-\frac{u}{2})^2}{\sigma^2}} dt = \sqrt{\frac{2\pi\sigma^2}{2}} = \sqrt{\pi}\sigma$. Another variant of the *OBK* is to also consider similarities between

opposite borders. This means that there is only one oligo function for a certain oligo and the occurrence positions of signals in the right border are numbered from one to $\min(n, b)$ from right to left. In this way, a high similarity between the right border of a peptide and the left border of another peptide can also be detected. Throughout the thesis, this kernel is called the *paired oligo-border kernel* (*POBK*) and the kernel function is:

$$
\begin{aligned}
k_{POBK}(s_i, s_j) = & \quad \sqrt{\pi}\sigma \sum_{\omega \in \mathcal{A}^k} \\
& \times \left[\sum_{p \in S_\omega^{U_i}} \sum_{q \in S_\omega^{U_j}} \mathrm{ind}(p,q) \cdot e^{-\frac{(p-q)^2}{4\sigma^2}} \right. \\
& + \quad \sum_{p \in S_\omega^{R_i}} \sum_{q \in S_\omega^{L_j}} e^{-\frac{((n-p+1)-q)^2}{4\sigma^2}} \\
& + \quad \left. \sum_{p \in S_\omega^{L_i}} \sum_{q \in S_\omega^{R_j}} e^{-\frac{(p-(n-q+1))^2}{4\sigma^2}} \right]
\end{aligned}
$$

This kernel function can be computed as efficiently as the oligo kernel by appropriate position encoding. The kernel matrix is positive definite which follows directly from [43], because the oligo border functions are also finite sums of Gaussians. Since preliminary experiments showed that the *POBK* performs better for chromatographic separation prediction than the *OBK*, we used only the *POBK* for prediction of chromatographic separation in this thesis. A comparison of the *OBK* and the *POBK* can be found in Section 3.3.3 for proteotypic peptide prediction. Furthermore, the preliminary experiments showed that the best performance of the k-mer length is one which is quite reasonable, since the peptides are very short compared to the number of different amino acids. This is also supported by a study on protein sequences [68], in which histograms of monomer distances performed better than distance histograms of longer k-mers. A combination of different lengths as in [43] also led to inferior results, which could be due to the normalization of the single kernel functions. Consequently, in the whole thesis, we only used k-mer length one.

P-Value Calculation and Filtering

As stated earlier, the retention time prediction is used in this work to improve the certainty of peptide identifications found by search engines like Mascot and to filter out false identifications. This is done by fitting a linear model to the prediction data in the training set. The model reflects the fact that retention times of late eluting peptides show a higher deviation than early ones. This can be explained by the constant relative error for retention times, which sums up the larger the RT becomes. Poorer performance in retention time prediction for longer peptides was also observed in [91]. For our predictions, we therefore match an area to the prediction data of the

training set which contains $\geq 95\%$ of the points and allows a larger deviation between observed and predicted normalized retention time (NRT) for larger retention times. An application of the model can be found in Fig. 3.8 b) and Fig. 3.8 c). We call the smallest distance in the model γ_0 at NRT equal to zero, and γ_{max} is the biggest gamma at NRT = 1. We can consequently calculate a corresponding gamma for every normalized retention time t_{nor} by $\gamma = \gamma_0 + t_{nor} \cdot (\gamma_{max} - \gamma_0)$. Since we assume Gaussian error distribution, gamma corresponds to double the *standard deviation* of the normal distribution such that a p-value can be calculated for every retention time prediction by calculating the probability that a correct identification has a larger deviation between observed and predicted normalized retention time. The null hypothesis is that the identification is correct. For filtering identifications, we use these p values in the following way.
Since we do not want to filter out correct identifications, the probability of filtering out a correct identification can be controlled by a significance level. In the experiments, we set the significance level to 0.05. This means that the probability that a correct identification has a deviation between observed and predicted retention time equal or greater than the allowed deviation is 0.05. The probability of filtering out correct identifications is thus 5%. Concerning the p-values mentioned above, this means that p has to be greater than 0.05.

Computational Resources

All methods introduced in this section were integrated into OpenMS, a software platform for computational mass spectrometry [122], which has a wrapper for the LIBSVM [14]. This library was used for the support vector learning. Furthermore, we integrated the prediction models into TOPP [58]. Some additional evaluations for peptide sample fractionation prediction were performed using shogun [117].

3.1.3 Experimental Methods and Additional Data

For peptide sample fractionation prediction, we used the data from Oh *et al.* [84] to assess the performance of our method. For peptide retention time prediction, we used different datasets. The first one is a validation dataset which was used by Petritis *et al.* in 2006 [91] to predict peptide retention times using artificial neural networks. In their experiment, they measured more than 345,000 peptides, and chose 1303 high-confidence identifications for testing and the remaining peptides for training. Since they only published the 1303 test peptides, we could only use this small number of peptides. The dataset was used in our study in order to to show the performance of our methods compared to other well established methods for peptide retention time prediction. Further datasets for retention time prediction were measured by Andreas Leinenbach, who was then in the laboratory of Prof. Dr. Christian Huber at Saarland University, to show that training on the data of one run suffices to predict retention times on the next runs very accurately and to improve spectrum identifications significantly.

Experimental Setup

The datasets for training and evaluation of the retention time predictor had to fulfill two basic requirements. First, the identity of the studied peptides had to be known with high certainty in order to avoid incorrect sequence annotations for the training dataset. Second, retention times had to be measured with high reproducibility. Altogether, Andreas Leinenbach measured 19 different proteins, which were purchased from Sigma (St. Louis, MO) or Fluka (Buchs, Switzerland). To avoid excessive overlapping of peptides in the chromatographic separations, the proteins were divided into three artificial protein mixtures and subsequently digested using trypsin (Promega, Madison, WI) using published protocols [106]. The protein mixtures contained the following proteins in concentrations between 0.4 - 3.2 pmol/μl:

- Mixture 1: β-casein (bovine milk), conalbumin (chicken egg white), myelin basic protein (bovine), hemoglobin (human), leptin (human), creatine phosphokinase (rabbit muscle), α1-acid-glycoprotein (human plasma), albumin (bovine serum).
- Mixture 2: cytochrome C (bovine heart), β-lactoglobulin A (bovine), carbonic anhydrase (bovine erythrocytes), catalase (bovine liver), myoglobin (horse heart), lysozyme (chicken egg white), ribonuclease A (bovine pancreas), transferrin (bovine), α-lactalbumin (bovine), albumin (bovine serum).
- Mixture 3: thyroglobulin (bovine thyroid) and albumin (bovine serum).

Adding albumin to each protein mixture was performed because in each run, there had to be an identical set of peptides to normalize the retention times. The resulting peptide mixtures were then separated using capillary IP-RP-HPLC and subsequently identified by electrospray ionization mass spectrometry (ESI-MS) as described in detail in [129, 106]. The separations were carried out in a capillary/nano HPLC system (Model Ultimate 3000, Dionex Benelux, Amsterdam, The Netherlands) using a 50 x 0.2 mm monolithic poly-(styrene/divinylbenzene) column (Dionex Benelux) and a gradient of 0-40% acetonitrile in 0.05% (v/v) aqueous trifluoroacetic acid in 60 min at 55 °C. The injection volume was 1 μl, and each digest was analyzed in triplicate at a flow rate of 2 μl/min. Online ESI-MS detection was carried out with a quadrupole ion-trap mass spectrometer (Model esquire HCT, Bruker Daltonics, Bremen, Germany).

Identification of Spectra and Normalization of Retention Times

Peptides were identified on the basis of their tandem mass spectra (maximum allowed mass deviations: precursor ions: $\pm$ 1.3 Da, fragment ions: $\pm$ 0.3 Da) using Mascot [87] (version 2.1.03). The database was the Mass Spectrometry Database, MSDB (version 2005-02-27) restricted to chordata. We allowed one missed cleavage as well as charges 1+, 2+ and 3+. The mass values were monoisotopic. The significance level of the significance threshold

score for the peptide hits was 0.05. Since the amino acid sequences of the 19 proteins of our mixtures are known, we could verify the identifications by sequence comparison with the protein sequences. To avoid random verifications, we restricted the peptide length to be equal or greater than six. The whole process led to two datasets for each protein mixture — one which only contained the verified peptides and the other one with all Mascot identifications. We call the datasets containing the verified peptide sequences *vds* and the datasets with all Mascot identifications *ds*. The *vds*s are used to train the predictors and the *ds*s are used to assess the classification performance of the identification process.
We chose two standard peptides which were identified in all of the runs. One of these peptides, which had the amino acid sequence `TCVADESHAGCEK`, elutes very early and the other one, which had the amino acid sequence `MPCTEDYLSLILNR`, elutes very late. We scaled the retention times linearly so that the early eluting peptide got an NRT of 0.1 and the late eluting peptide an NRT of 0.9. All peptides with an NRT below zero or above 1 were removed.

Reimplementation of Existing Methods for Comparison Purposes

For retention time prediction we compared our method with several methods. Therefore we had to reimplement the methods by Klammer *et al.* [55] as well as the methods by Petritis *et al.* [91]. For the methods by Klammer *et al.*, we implemented the same encoding as described in the literature and used the RBF kernel of the LIBSVM [14]. The cross validation was performed with the same parameter ranges as described in the paper ($C \in \{10^{-3}, 10^{-2}, \ldots, 10^{7}\}$ and $\sigma \in \{10^{-6}, 10^{-7}, 10^{-8}\}$). For comparison with the models by Petritis *et al.* we reimplemented the models as described in the literature using Matlab R2007a (The MathWorks, Inc., United States) and the neural networks toolbox version 5.0.2 (The MathWorks, Inc.). This means that for the first model of Petritis *et al.* [90] we had a feedforward neural network with 20 input nodes, two hidden nodes and one output node. The frequencies of the amino acids of the peptides served as input. For the second model of Petritis *et al.* [91] we had 1052 input nodes, 24 hidden nodes, and one output node. The amino acids at the 25 leftmost and the 25 rightmost residues served as input as well as the length and the hydrophobic moment of the peptide as described in [91]. Both models were trained using a backpropagation algorithm.

3.1.4 Results and Discussion

In this section, we present the results for two different application areas of our new kernel function. The first one is peptide sample fractionation prediction in SAX-SPE, and the second one is peptide retention time prediction in IP-RP-HPLC experiments. For peptide sample fractionation prediction, we demonstrate that our method performs better than the established method. In retention time prediction, we show that we obtain good predictions with very little training data. This allows to train our predictor with a dataset

measured in one run to predict retention times of the next runs very accurately. Peptide identification is improved afterwards by filtering out all peptides with a large deviation between observed and predicted retention time.

Performance of Peptide Sample Fractionation Prediction

In order to be able to compare our results with existing methods, we used the same dataset and the same setup as Oh *et al.* [84]. We randomly partitioned our data into a training set and a test set with 120 peptides for training and 30 peptides for testing. Performance was measured by classification success rate (SR), which is the number of successful predictions divided by the number of predictions. The whole procedure was repeated 100 times to minimize random effects. The training was conducted by a 5-fold cross-validation (CV) and the model was trained using the best parameters from the CV and the whole training set.
To compare our new kernel function with established kernels, we used the best four feature combinations of Oh *et al.* [84] and trained an SVM with the polynomial and the RBF kernel for each feature combination. Feature number one is molecular weight, the second is sequence index, the third is length, and the fourth feature is the charge of the peptide. We used the same evaluation setting as described above and in the 5-fold CV the SVM parameter $C \in \{2^{-4} \cdot 2^i | i \in \{0, 2, \ldots, 14\}\}$. For the σ parameter of the RBF kernel, $\sigma \in \{2^{-15} \cdot 2^i | i \in \{0, 1, \ldots, 24\}\}$ and for the degree d of the polynomial kernel, $d \in \{1, 2, 3\}$.
The results are shown in Table 3.1. It seems as if the fourth feature (the charge of the peptide) is the most important factor but molecular weight also seems to improve the prediction performance.

Feature combination	Polynomial kernel	RBF kernel
1, 2, 3, 4	0.78	0.80
1, 2, 3	0.66	0.63
1, 2, 4	0.78	0.80
2, 3, 4	0.75	0.75

Table 3.1: ***Peptide sample fractionation prediction using standard SVMs:*** *This table shows the classification success rates of the different feature combinations for SVMs with the polynomial and the RBF kernel on the dataset of Oh et al. [84]. The features are (1) molecular weight, (2) sequence index (3) length and (4) charge of the peptide calculated as in [84].*

An independent approach which just uses the sequence information of the peptides was performed using the local-alignment kernel by Vert *et al.* [136]. Using the same setup as described above, we used the BLOSUM62 matrix [41] and the kernel function parameters were the following:
$\beta \in \{0.1, 0.2, 0.5, 0.8, 1\}$, $d \in \{1, 3, 5, 7, 9, 11, 13\}$ and $e \in \{1, 3, 5, 7, 9, 11, 13\}$. Nevertheless, the performance of these kernel approaches led to inferior results than the published method by Oh *et al.* [84]. Therefore more appropriate kernel functions are needed, like our new *POBK*, which is explained

in Section 3.1.2. The kernel function has a kernel parameter b which is the border length of the peptide. A small b means that only few border residues of the peptides contribute to the kernel function, and a border length equal to the sequence length would mean that all residues contribute to the kernel function value. To determine the best border length of the *POBK*, we performed the evaluation for all $b \in \{1, \ldots, 30\}$. The evaluation of border length b depicted in Fig. 3.1 shows that for a b greater than 19, the SR does not change significantly, with a slight improvement for $b = 22$. This is why in the following, only the *POBK* for $b = 22$ is considered. To study the relation between border length and the length of the peptides, we plotted a histogram of peptide lengths in Fig. 3.2. It can be seen that with border length 22 all amino acids of the peptides are considered in at least one of the two borders.

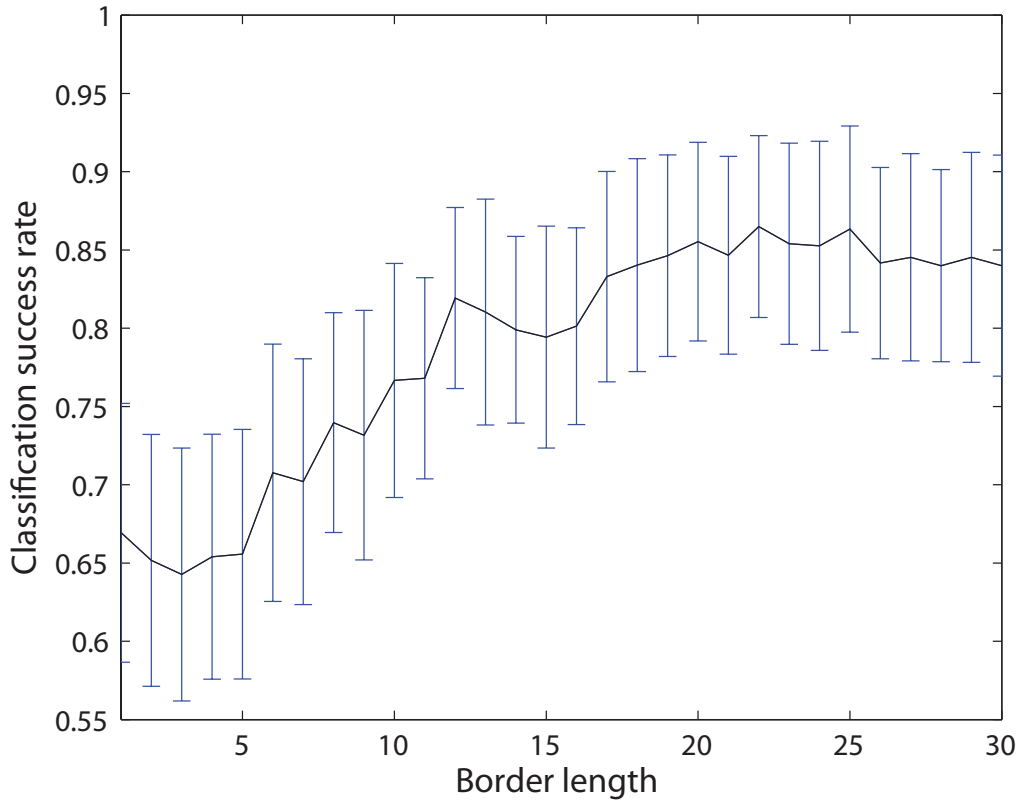

Figure 3.1: ***Border length evaluation of the POBK:*** *This figure shows the evaluation of SR using different border lengths b for the POBK on the dataset of Oh et al. [84].*

A comparison of the SR for different methods can be found in Fig. 3.3. The first two bars represent the SR performance of the best SVMs using standard kernels of Table 3.1. The third bar demonstrates the performance of an SVM with the local-alignment kernel. The fourth bar shows the performance of the best predictor in Oh *et al.*, which is 0.84. The last bar represents the SR of the *POBK* for peptide sample fractionation and retention time prediction. The SR of this method is 0.87, which is significantly better than all other approaches. Since the dataset is very small, there is a significant deviation between performances of different runs. Therefore, Fig. 3.4 shows a boxplot

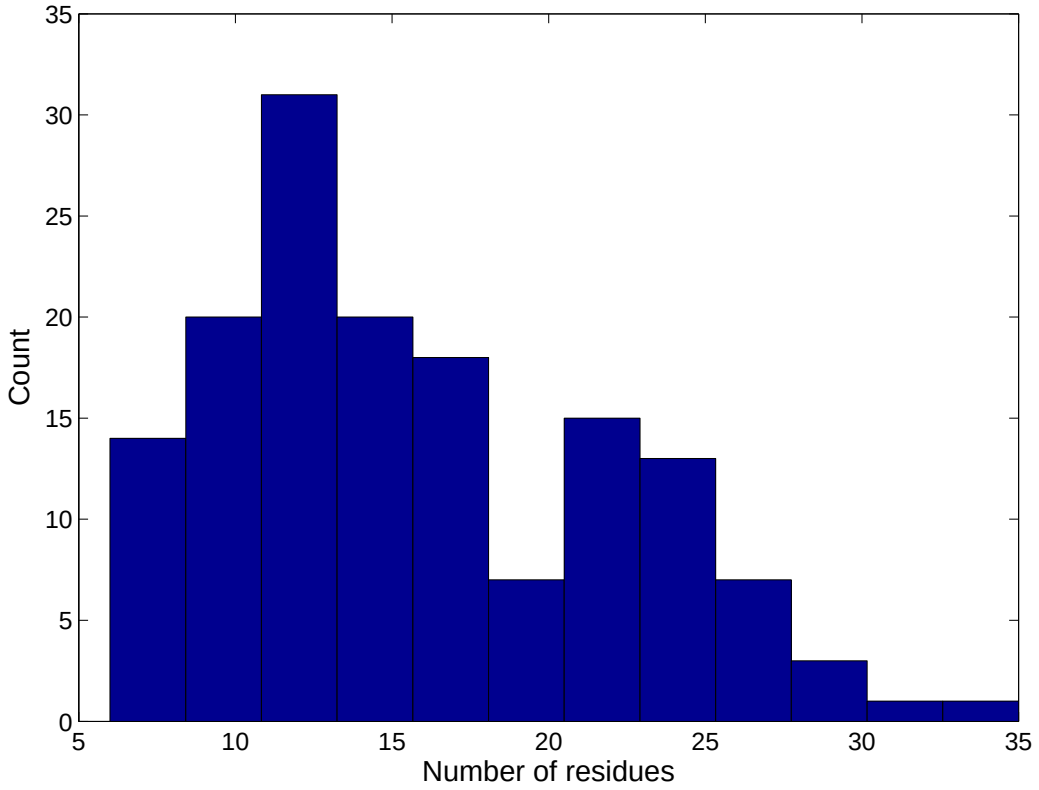

Figure 3.2: ***Length distribution of peptides in dataset:*** *This figure shows a histogram of the peptide lengths of the dataset of Oh et al. [84].*

of the methods, for which we performed the evaluation.

Correctly Predicted Peptides in Peptide Sample Fractionation Prediction

In Oh *et al.* [84] the prediction process with 100 random partitionings was done for the best four predictors, and for every peptide, the whole predictions were stored. Oh *et al.* then classified a peptide by the majority label which had been assigned to the peptide. By this method, they were able to assign 127 of the 150 peptides correctly, which corresponds to an SR of 0.85.
To be able to compare this procedure with our method, we made the assumption, that for a particular peptide, the SVM would make a correct assignment more often. Furthermore, we assumed that if we also stored the predictions for each peptide and each run, we could also get a majority predictor which yields good performance. The evaluation of this procedure shows that we are able to predict 134 peptides correctly in this setting, which is an SR of 0.8933. Fig. 3.5 shows a histogram of the SRs for the different peptides for the method by Oh *et al.* [84] and the SVM with the *POBK*.

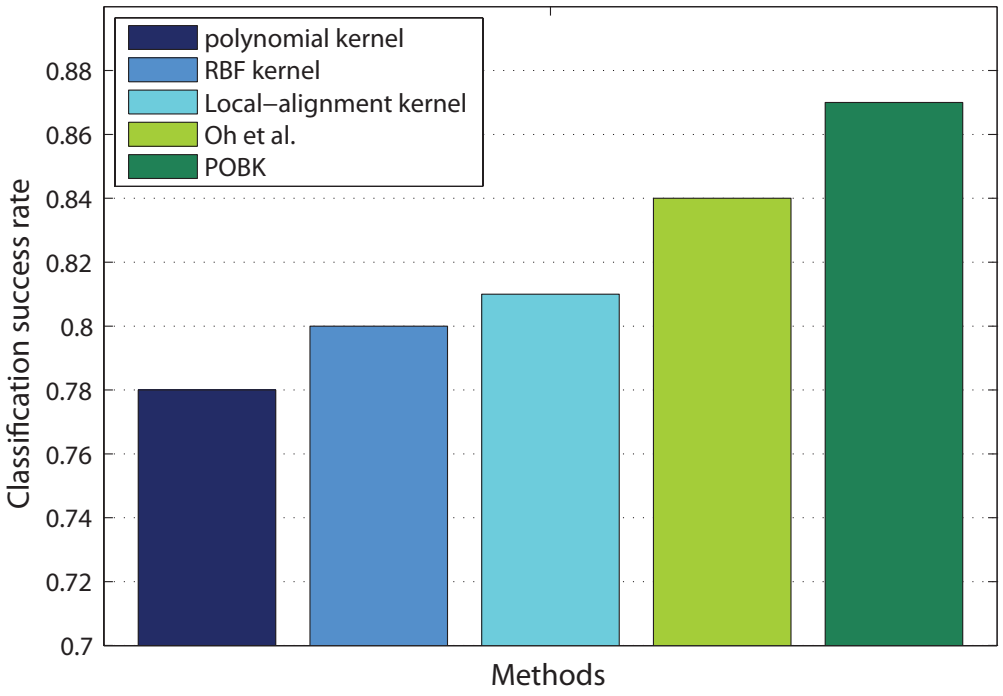

Figure 3.3: ***Performance comparison for peptide sample fractionation prediction:*** *Comparison of classification success rates for different methods predicting peptide sample fractionation on the dataset of Oh et al. [84].*

Evaluation of Model Performance for Peptide Retention Time Prediction

For peptide retention time prediction, we had several goals. The first one was to construct a retention time predictor showing equivalent performance as established methods but requiring just a fraction of the training set size. To demonstrate that our retention time predictor fullfills the desired constraints, we performed a 2-deep CV on the Petritis dataset [91] described in Section 3.1.3. This means that we partitioned the data randomly into ten partitions and performed a CV with the data from nine of the ten partitions to find the best parameters. Later, we trained our model with the best hyperparameters and the data of the nine partitions to evaluate the performance of the predictor on the omitted tenth partition. This was done for every possible combination of the ten partitions and the whole procedure was repeated ten times to minimize random effects.
A plot of the observed normalized retention time against the predicted normalized retention time can be seen in Fig. 3.6 for one of the ten 2-deep CV runs. Since the standard deviation of the Pearson correlation between observed and predicted NRT over the ten runs was 0.0007, this plot is quite representative for the model performance. Petritis *et al.* [91] showed that their method performs better than those of Meek [75], Mant *et al.* [74], Krokhin *et al.* [61] and Kaliszan *et al.* [46], using this dataset for validation. Thus, in Table 3.2, we only compare the performance of our method with the work of Petritis *et al.* [91].

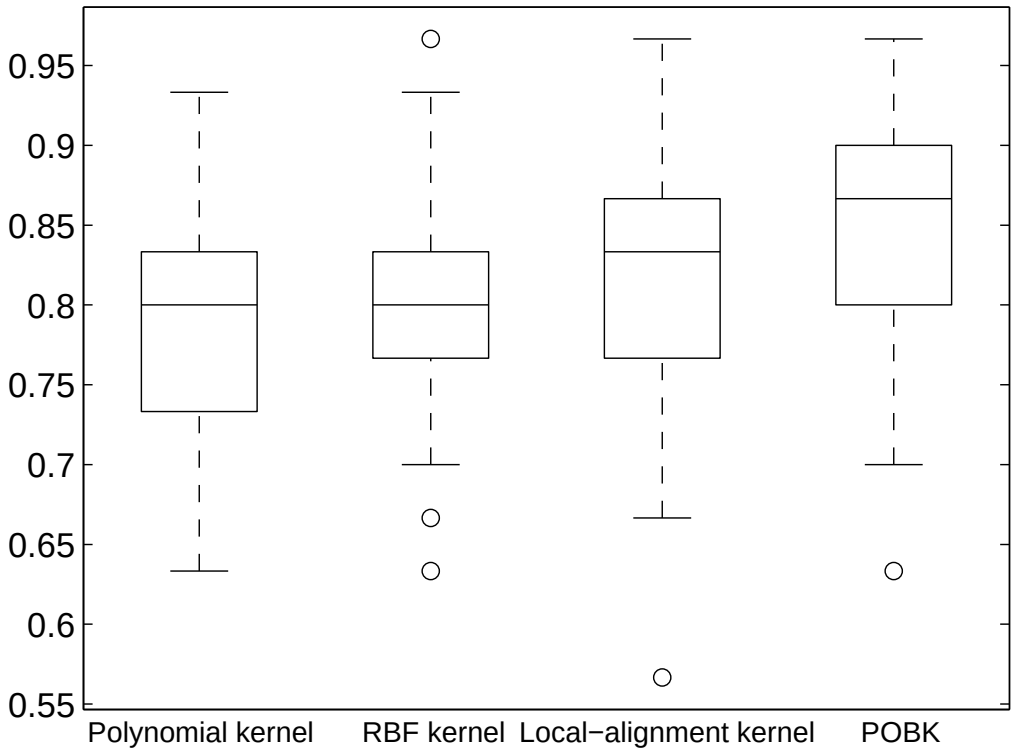

Figure 3.4: ***Boxplot for peptide sample fractionation prediction success rates:*** *Boxplot of classification success rates for different methods predicting peptide sample fractionation on the dataset of Oh et al. [84]. The boxplot is produced with Matlab using standard parameters: The central mark of each box represents the median. The box edges represent the 25th and the 75th percentiles and the whiskers extend to the most extreme data points which are not considered as outliers. Outliers are visualized by circles.*

Method	Number of training sequences	R^2
Meek 1980 [75]	344611	0.816
Mant *et al.* 1988 [74]	344,611	0.833
Krokhin *et al.* 2004 [61]	344,611	0.844
Kaliszan *et al.* 2005 [46]	344,611	0.817
Petritis *et al.* 2003 [90]	344,611	0.870
Petritis *et al.* 2006 [91]	344,611	0.967
This work	**1040**	**0.880**
	200	**0.854**
	100	**0.805**

Table 3.2: ***Comparison of different retention time predictors:*** *This table shows the squared correlation coefficient (R^2) between observed and predicted normalized retention time of established retention time prediction methods presented in [91] on the Petritis test set [91]. These values are compared to our method, the POBK, on the Petritis test set [91]. The second column gives the number of training sequences used. For the last two rows, subsets of the data were chosen randomly so that 100 respectively 200 training peptides were selected.*

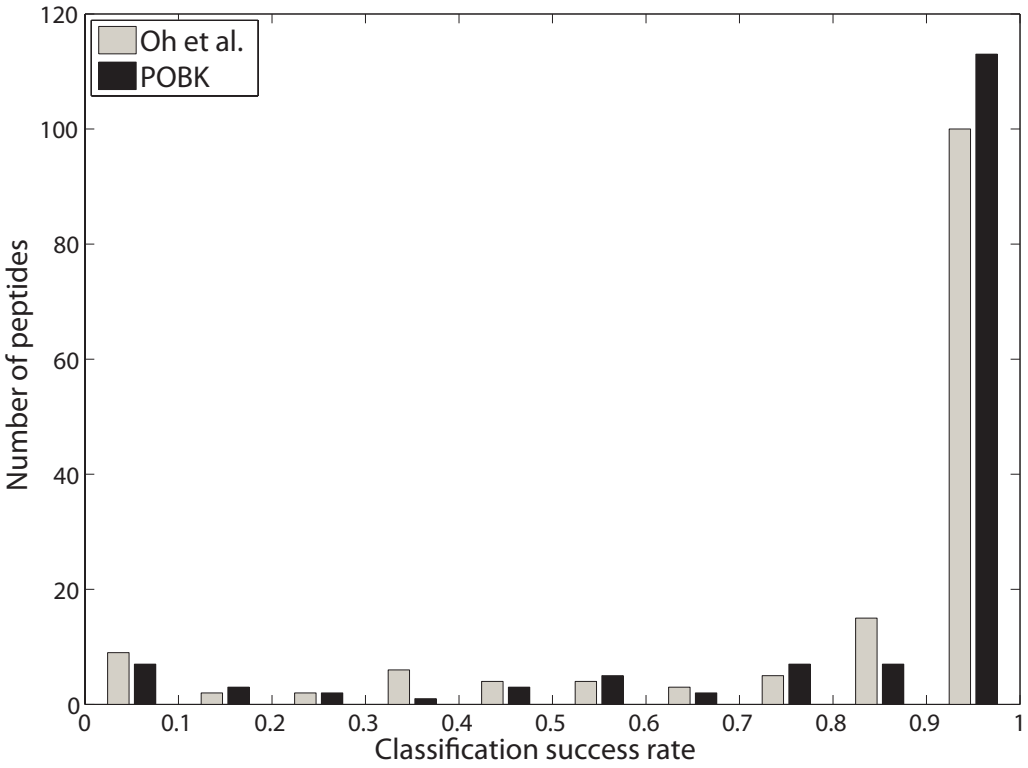

Figure 3.5: ***Histogram of classification success rate:*** *This figure shows a histogram of the SR of particular peptides using the majority classifier on the data set of Oh et al. [84]. This is compared to the ensemble prediction of Oh et al.*

This comparison is somewhat biased since we only had a fraction of the original validation set for training, which means that our training set size was 300 times smaller than that of the other methods. Nevertheless, our method performs better than the model [90] which is used by Strittmater *et al.* [121] in their filtering approach. The only model with a better performance is the artificial neural network with 1052 input nodes and 24 hidden nodes [91]. It is obvious that a model like this needs a very large amount training data. Petritis *et al.* [91] trained their model with more than 344,000 training peptides. Therefore, this type of model is not suitable for retention time prediction for measurements under different conditions or with different machines because it is very time consuming to acquire identification and retention time data for more than 344,000 training peptides before starting the actual measurements.

To demonstrate that our method is robust enough for training on verified data of one single run, we constructed a non-redundant dataset from datasets *vds1* and *vds2*. A detailed description of these datasets can be found in Section 3.1.3. For different training sizes $s \in \{10, 20, \ldots, 170\}$, we randomly selected s peptides for training and 40 peptides for testing. Fig. 3.7 indicates that for the *POBK*, 40 verified peptides are sufficient to train a predictor which has a squared correlation coefficient between observed and predicted normalized retention time greater than 0.9 on the test set. This number is much smaller than the number of verified peptides we get for one run since *vds1* has 144 peptides, *vds2* has 133 peptides and *vds3* has 116. This evalu-

ation shows that with our predictor, it is possible to measure one calibration run with a well-defined and easily accessible peptide mixture prepared from real biological samples to train a predictor, which can then be used to predict retention times for the peptides very accurately. Furthermore, Fig. 3.7 shows a comparison of the *POBK* to the methods introduced by Klammer *et al.* [55] and Petritis *et al.* [90, 91] as described in Section 3.1.3. Our method needs significantly less training data for a good prediction and has also superior performance if all training sequences of our dataset are used. One possible explanation for the low performance of the models from Petritis *et al.* is that their models need a larger amount of training data. This is supported by the fact that they used about 7,000 [90] and about 345,000 [91] training peptides in their studies. To compare our method with the work by Krokhin [60], we used our verified datasets. We trained our model on *vds1* and predicted the retention times for peptides of the union of *vds2* and *vds3*, which were not present in *vds1*. If a peptide occured in *vds2* and in *vds3*, we only kept the peptide identification with the biggest score. For the *POBK*, we performed a 5-fold CV with SVM parameters $C \in \{2^i | i \in \{-9, -8, \ldots, 0\}\}$, $\nu \in \{0.4 \cdot 1.2^i | i \in \{0, 1, 2\}\}$ and $\sigma \in \{0.2 \cdot 1.221055^i | i \in \{0, 1, \ldots, 21\}\}$ to determine the best parameters. Afterwards, we trained our model with the whole training set and the best parameters and calculated the squared correlation between observed and predicted retention time on the test set. This procedure was repeated ten times to minimize random effects. Since there exists a web server for the method by Krokhin [60], we could also compare the observed retention times with the predicted ones on our test sets with this method. To calculate the hydrophobicity parameters a and b of this method, we used our two standard peptides introduced in the Section 3.1.3. Furthermore, we used the 300 Å column since the other columns led to inferior results. As can be seen in Table 3.3, the model by Krokhin performs quite well even though it had been developed on another type of sorbent. Nevertheless, the *POBK* achieves a significantly higher squared correlation coefficient. It should be noted that the web-server by Krokhin is restricted to three different columns. The advantage of our method is that there is not any restriction to a certain type of experimental setup. One only needs a small amount of training peptides and can train a model which can immediately be used for retention time prediction.

It should be mentioned that the *POBK* has a higher squared correlation

Training set	**Test sets**	**POBK**	**Krokhin [60]**
vds1	$(vds2 \cup vds3) \setminus vds1$	0.9570	0.9101
vds2	$(vds1 \cup vds3) \setminus vds2$	0.9564	0.9212
vds3	$(vds1 \cup vds2) \setminus vds3$	0.9521	0.9229

Table 3.3: ***Evaluation of prediction performance for retention time prediction using the POBK:*** *This table shows the performances of the POBK using our verified datasets (introduced in Section 3.1.3). The other columns contain the squared correlation coefficient between the observed normalized retention times and the predicted ones for the POBK and the method by Krokhin [60].*

between observed and predicted retention time on our datasets than on the

testset by Petritis *et al.* This could be due to the fact that Petritis *et al.* performed peptide identification using database search [91]. It is commonly accepted that this results in a significant false positive rate.

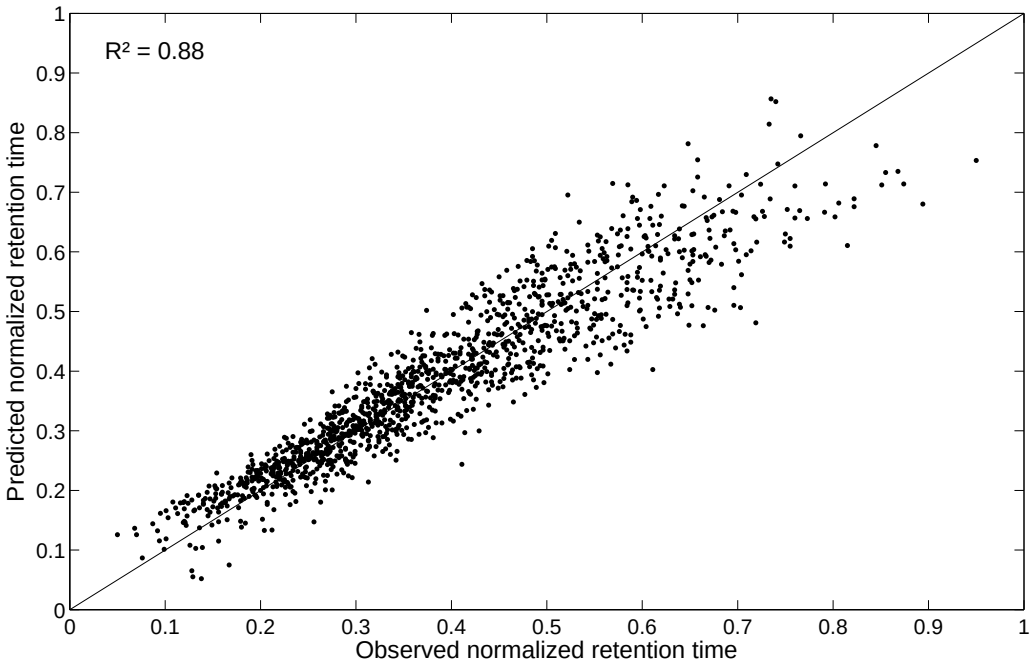

Figure 3.6: ***Example figure for peptide retention time prediction:*** *This plot shows the observed normalized retention time against the predicted normalized retention time for one of ten 2-deep CV runs on the Petritis test set [91]. Since every peptide occurs exactly once in the test set, this plot shows predictions for all of the peptides in the Petritis dataset.*

Improving Peptide Identifications by Using Retention Time Prediction

The second goal for retention time prediction was to elaborate a retention time filter which could be used for improving peptide identifications. In this setting, we trained our learning machine on one of the *vds* (i.e. *vds1*) and predicted the retention times for the remaining *ds* (i.e. *ds2* and *ds3*). The peptides of the training and test sets were made disjoint by removing all identifications of the test set which belonged to spectra having an identification which was also present in the training set. On every training set, we performed a 5-fold CV with SVM parameters $C \in \{2^i | i \in \{-9, -8, \ldots, 0\}\}$, $\nu \in \{0.4 \cdot 1.2^i | i \in \{0, 1, 2\}\}$ and $\sigma \in \{0.2 \cdot 1.221055^i | i \in \{0, 1, \ldots, 21\}\}$. Since the results of the *POBK* for all three datasets in Table 3.3 show nearly the same squared correlation coefficient of about 0.95 between observed and predicted normalized retention times, we restricted ourselves in the following to training our learning machine on *vds3* and evaluated the filtering capability of our filtering approach on *ds1* and *ds2*.
The performance evaluation of our filter model was done by a two-step approach. In the first step, we measured the number of true positives and the

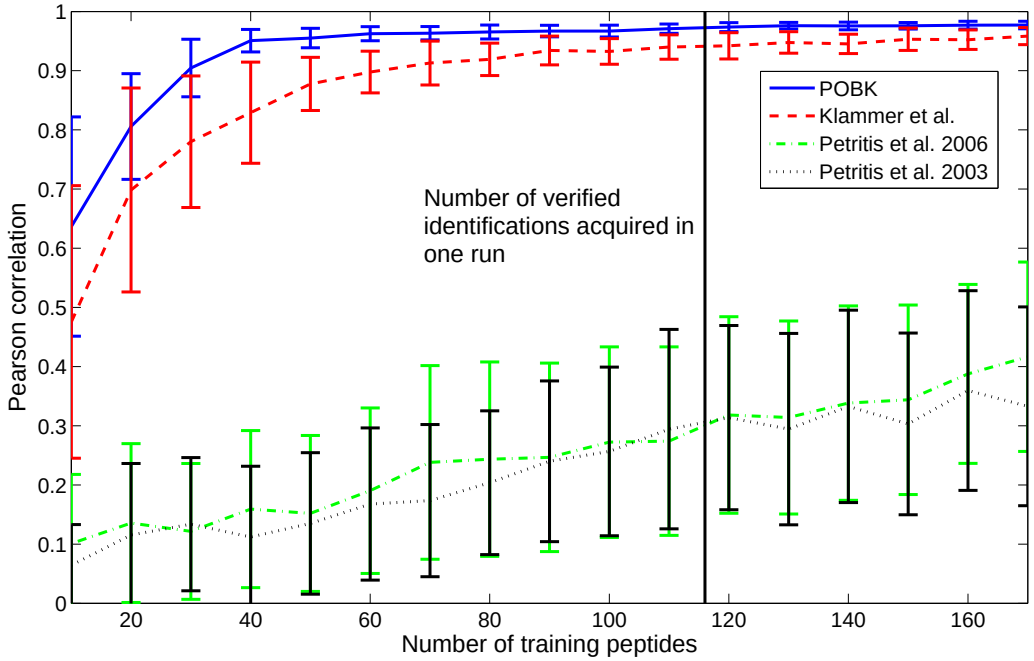

Figure 3.7: ***Learning curve for peptide retention time prediction:*** *This plot shows the Pearson correlation coefficient depending on the number of training samples for the union of vds1 and vds2. For every training sample size, we randomly selected the training peptides and 40 test peptides and repeated this evaluation 100 times. The plot shows the mean correlation coefficients of these 100 runs for every training sample size as well as the standard deviation for the POBK, the methods introduced by Klammer et al. [55] using the RBF kernel, and the methods by Petritis [90, 91]. The vertical line corresponds to the minimal number of distinct peptides in one of our verified datasets which was acquired in one run.*

number of false positives for the identifications returned by the Mascot [87] search engine. This was conducted for different significance values. Mascot provides a significance threshold score for the peptide identification at a given significance level (0.05 in all our studies). In order to be able to compare the identification performance at different levels of certainty we chose different fractions of the significance threshold score. This means for example, that for a fraction of 0.5, all identifications have to have a score which is equal to or greater than half of the significance threshold score. The evaluation was accomplished for varying threshold fractions $t \in \{0.01, 0.02, \ldots, 1\}$. In this setting, we could evaluate the precision. This is the number of true identifications with a score higher than t times the significance threshold divided by the number of spectra having at least one identification with a score higher than t times the significance threshold score. If there was more than one identification with the maximal score for one spectrum, the spectrum was excluded from the evaluation. In the second step, we filtered the data by our retention time model which was trained on the training set and conducted the same evaluation as in the first step. After this, we compared the classification performance of these two evaluations. Fig. 3.8 a) demonstrates the good precision for identifications with high Mascot scores. A threshold fraction equal

to one means that all identifications have a score equal or larger than the significance threshold score given by the Mascot search engine. Nevertheless, even for these identifications, filtering with the retention time filter improves the precision from 89 to 90%. An even greater improvement can be achieved for identifications with smaller scores. If all identifications are constrained to have a score equal or larger than 60% of the significance threshold score, the precision improves from 55 to 77% by using our filter. A precision of 0.77 is still quite good and, as can be seen in Table 3.4, the number of true positives increases from 350 to 557. This means that a significantly larger number of spectra can be identified with an acceptable number of false positives by applying our retention time filtering approach. Fig. 3.8 b) shows that our model is valuable for removing false identifications since many false positives have larger deviations between observed and predicted NRT than allowed and are removed by our filter (threshold fraction of 0.95). Fig. 3.8 c) shows this even more drastically for a threshold fraction of 0.6. The whole evaluation shows that our retention time prediction can be used to improve the level of certainty for high-scoring identifications and also to allow smaller thresholds to find new identifications with an acceptable number of false positives.

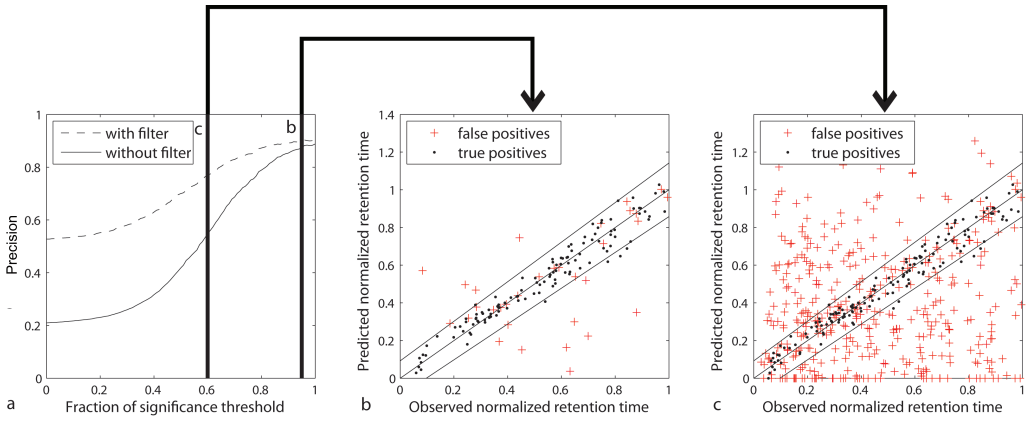

Figure 3.8: ***Visualization of filter performance:*** *This plot shows the improvement in precision one can get by using our retention time filter for a) varying fractions of the significance threshold value, b) all predictions of spectra having a score equal or greater than 95% of the significance threshold value, c) all predictions of spectra having a score equal or greater than 60% of the significance threshold value. The model was trained using the vds3 dataset and the performance was measured on ds1 and ds2. If there was more than one spectrum with the same identification we plotted the mean values of the observed NRT versus the predicted NRT.*

Fraction of threshold	tp	fp	Precision	tp with filter	fp with filter	Precision with filter
0.0	683	2572	0.2098	699	626	0.5275
0.1	682	2460	0.2171	692	602	0.5348
0.2	678	2260	0.2308	683	555	0.5517
0.3	669	1909	0.2595	668	483	0.5804
0.4	654	1410	0.3169	646	380	0.6296
0.5	624	868	0.4182	609	261	0.7000
0.6	575	474	0.5481	557	166	0.7704
0.7	516	235	0.6871	500	103	0.8292
0.8	468	125	0.7892	452	66	0.8726
0.9	420	72	0.8537	404	49	0.8918
1.0	366	46	0.8883	350	38	0.9021

Table 3.4: ***Evaluation of filter performance:*** *This table presents the precisions of the identified spectra for varying fractions of the significance threshold with and without retention time filtering. The model was trained using the vds3 dataset and the performance was measured on ds1 and ds2. In this context, tp stands for the number of true positives (correct hypotheses which are significant at the particular significance level) and fp for the number of false positives (false hypotheses which are significant at the particular significance level). The precision is tp divided by the sum of tp and fp.*

3.1.5 Conclusions

In this section, we introduced a new kernel function which was successfully applied to two problems in computational proteomics, namely peptide sample fractionation by SAX-SPE and high resolution peptide separation by IP-RP-HPLC. Furthermore, we demonstrated that the predicted retention times can be used to build a p-value-based model which is capable of filtering out false identifications very accurately.

Our method performs better than all previously reported peptide sample fractionation prediction methods. For retention time prediction, our method is (to our knowledge) the only learning method which can be trained with a small training size of 40 peptides, while still achieving a high correlation between observed and predicted retention times. This small required training set allows us to imagine the following application which would be very helpful for proteomic experiments. One could identify a well-defined protein mixture before starting the experiments and use the verified peptides for training the predictor. Next, the predictor can be used to predict retention times for all identifications of the following runs. This predicted retention time can then be applied to improve the certainty of the predictions. It can also be used to identify a much larger number of spectra with an acceptable number of false positives. This is achieved by lowering the significance threshold and filtering the identifications by our p-value-based retention time filter. The best σ was usually between five and seven in our experiments. A very small value of σ (e.g., 0.3) would indicate, that positional information is very important and that the positional smearing does not improve prediction results. A very large σ (e.g., 30) would indicate that positional information is not important for the prediction problem. Since the optimal σ was between five and seven, this indicates that the positional smearing is reasonable. The more training sequences are available, the better the positional information is represented. Therefore, the optimal σ is expected to be smaller when more training sequences are available.

Since all our methods are integrated into the OpenMS [122] library, which is open source, every researcher is able to use the presented methods free of charge. Also, we offer the prediction models as tools which are part of the OpenMS proteomics pipeline (TOPP) [58]. These tools can be easily combined with other tools from TOPP building sophisticated applications in computational proteomics. One application is, for example, a simulator for LC-MS maps, called LC-MSsim [113], which was built using OpenMS and TOPP. The RT of the peptides are predicted using an SVM and the *POBK*. Another application is the combination of retention time prediction, prediction of peptide proteotypicity (see Section 3.3), and peptide fragmentation prediction to design scheduled multiple reaction monitoring experiments [3], which we presented at the Proteomic Forum 2009 (manuscript in preparation).

Further research could be pursued to enhance retention time prediction by using multiple kernel learning (MKL) [117] with the 2-norm optimization [57]. Therefore, one could combine the *POBK* and *OBK* for different sigmas and k-mer lengths with other kernels which contribute features that cannot be

directly learnt from the sequence. In preliminary studies, we evaluated the performance of such kernel combinations with the 1-norm optimization but, unfortunately, did not get increased performances. The 1-norm multiple kernel learning tends towards sparse kernel combinations and, therefore, does not lead to better performances in many applications, which could explain the results of our experiments. The 2-norm optimization problem of Kloft *et al.* [57] was presented only very recently. It would be very interesting to evaluate different kernel combinations with this approach to improve retention time prediction.

3.2 Two-Dimensional Chromatographic Separation Prediction

3.2.1 Introduction

We already saw in Section 3.1 that there are many approaches based on machine learning techniques in which a measured parameter such as the chromatographic retention time of a peptide is compared to a predicted one to filter out false spectrum identifications in mass spectrometry-based experiments [121, 55]. In addition to chromatographic retention, there were other properties of the peptides which were used to improve the number of identified spectra. Klammer *et al.* [56] predicted the fragmentation of spectra and through this they could improve the identification process by incorporating the predicted likelihood of a spectrum identification to be correct. Uwaje *et al.* [133] used a database of measured pairs (peptide, pI) to improve peptide identification.

Two-dimensional separations are most commonly used for the analysis of complex samples due to limited peak capacities of only one separation dimension. The most common combinations of chromatographic techniques are strong cation exchange chromatography (SCX) with reversed-phase (RP) or ion-pair reversed-phase (IP-RP) high-performance liquid chromatography (HPLC) [2]. Toll *et al.* [129] and Delmotte *et al.* [20] were able to show that peptide separation on reversed-phase stationary phases using different pH and eluent additives showed significant orthogonality. In the present work, an offline combination of RP-HPLC at pH 10.0 with IP-RP-HPLC at pH 2.1 was used [20]. Although the two separation dimensions are not fully orthogonal, the combination leads to better "peptide identification yield" compared to the classical combination of SCX with RP [20]. This is mainly based on the fact that in this combination the fractions collected from the first-dimension separation contain no salt and can, after concentration, be injected directly into the second separation system.

In this section we significantly extend the applicability of peptide retention prediction [94] to whole proteome analysis by incorporating retention time predictors for both separation dimensions. By doing so, we are able to incorporate essentially four different peptide properties into an identification scheme, namely peptide retention in high-pH reversed chromatography, peptide retention in low-pH ion-pair reversed-phase chromatography, and intact molecular mass and fragmentation pattern of a peptide. This means that we build a model for the first as well as the second separation dimension and then use predicted and observed retention times to build one filter for the first as well as one filter for the second dimension. We show that each filter independently improves the precision of the spectrum identifications, whereas the largest improvement in precision can be achieved by combining the filters. In this way, one can get about 35% more spectrum identifications at the same precision for a standard protein mixture analyzed according to this protocol. In order to show the feasability of this approach to the analysis of whole proteomes, the filtering methods were applied to a whole cell

lysate of the *Sorangium cellulosum* bacteria, which also yielded an increase of about 26% in terms of the number of uniquely identified spectra.

3.2.2 Methods and Data

Experimental Setup

The data sets for the standard mixture and the whole digested proteome were generated with emphasis on high reproducibility in terms of retention times using an actively split capillary HPLC system (Ultiate 3000, Dionex, Germering, Germany). This was done by Andreas Leinenbach, who was then at the laboratory of Prof. Dr. Christian Huber at Saarland University. Separated peptides were detected and identified by electrospray ionization tandem mass spectrometry (ESI-MS/MS) in an ion trap mass spectrometer (HCT Ultra PTM Discovery System, Bruker Daltonics, Bremen, Germany). Two different tryptic peptide mixtures were analyzed: a simple protein digest as a training and validation data set and a tryptic digest of a whole protein extract from *Sorangium cellulosum*. The simple protein mixture consisted of albumin (220 fmol/μl, bovine serum, Sigma Aldrich, St. Louis, MO, USA) and thyroglobulin (410 fmol/μl, bovine thyroid gland, Fluka, Buchs, Switzerland). The proteomic sample was from *Sorangium cellulosum* (So ce56, digest of 690 μg of protein ectract), a soil-dwelling bacterium from the group of myxobacteria. Proteins were digested with trypsin (Promega, Madison, WI, USA) using published protocols [106]. The peptide mixtures were separated using an offline two-dimensional HPLC setup as described in reference [20]. They combined reversed-phase (RP) high-performance liquid chromatography (HPLC) at pH 10.0 with micro ion-pair reversed-phase (IP-RP) HPLC at pH 2.1. Finally, the training data set was used to characterize both separation dimensions. In total, 36 fractions of the simple protein digest (fractions 4 to 39) and 31 fractions from the analysis of *Sorangium cellulosum* (fractions 14 to 44) were analyzed in triplicate in the second dimension.

Peptide Identification and Normalization of Retention Times

We aligned the MS/MS spectra of the standard mixture by the algorithm of Lange *et al.* [65] using standard parameters. This was also done for the MS/MS spectra of *S. cellulosum*. We identified the MS/MS spectra using Mascot (version 2.2) [87] with one missed cleavage, precursor tolerance 1.3 Da, carboxymethyl as fixed modification and deamidated asparagine or glutamine as well as oxidized methionine as variable modifications. For the standard mixture, we searched against the MSDB database, restricted to chordata (vertebrates and relatives). For the *S. cellulosum* spectra we used an in-house database containing all protein sequences of the organism constructed from the published DNA sequence [107]. For both data sets we also searched the spectra against a reverse version of the database. In this way, we could estimate the FDRs and q-values of the spectrum identifications as described in [48] and Section 2.2.5.
All spectrum identifications corresponding to peptides shorter than six amino

acids were filtered out since identifications of shorter length are less reliable and in most cases they cannot be mapped uniquely to protein sequences.

Prediction of Retention Times and Filtering by Retention Times

In this application area, the retention times were predicted with an improved version of the method introduced in Section 3.1. The retention time predictors use ν-SVR, introduced in Section 2.1.4, and the *POBK* to train the predictor. All methods are integrated into the open-source framework for mass spectrometry (OpenMS) [122]. The tools for retention time prediction and filtering are part of the OpenMS Proteomics Pipeline (TOPP) [58]. We now describe the extensions to the methods presented in Section 3.1.
The main advantage of the *POBK* in application to computational proteomics is that it enables the learning machine to learn chemical properties of the data (e.g., composition, sequence length, hydrophobic regions) directly from the amino acid sequence. It was shown that very little training data is needed for ν-SVR in combination with the *POBK* to achieve very accurate retention time prediction models. The kernel operates directly on the sequence data on which every different amino acid is considered as a separate letter in the alphabet. In this section, we extend this alphabet to modified amino acids. For example, a modified methionine with an additional methyl group is treated differently than an oxidized methionine. The method does not rely on any special features because it learns the necessary features for the particular separation process directly from the training data. Therefore, the *POBK* can be applied to a wide range of problems like separation prediction in strong anion-exchange chromatography and reversed-phase chromatography. It can also be used to learn peptide retention behavior under different pH conditions.
A further extension to the method introduced in Section 3.1 is that one does not have to normalize the retention times to the interval between zero and one. Instead, the aligned retention times can be used directly to train the learning machine. The learned retention time models for each dimension are then used to build a retention time filter for the corresponding dimension. The filters are based on a statistical test which measures how likely it is, that the peptide under consideration is a true identification. Therefore, the measured and the predicted retention times are taken into account and the user can specify a certain significance level for the filter.

Evaluation of Precision of the Identifications

Precision (PR) was measured for different subsets of the spectrum identifications. PR is defined as the number of true positives (TP) divided by the sum of the number of true positives and the number of false positives (FP):

$$\mathrm{PR} = \frac{TP}{TP + FP}. \tag{3.16}$$

In our application, the precision is the number of spectra for which the best-scoring identification is correct divided by the total number of identified spectra. Before the evaluation of the precision, we removed all spectra for which the best score was not unique among the identifications for the particular spectrum.

3.2.3 Results and Discussion

Retention Time Prediction at pH 10.0 and pH 2.1

Because fractions of peptides were taken in the first dimension, we can only assign retention windows for peptide elution and take the median of the elution window as the retention time for all peptides contained in a fraction. To show that the method performs well for the prediction of retention times in both dimensions, we performed a nested cross-validation on a subset of the data with high-quality identifications. Therefore, we utilized all spectrum identifications with a q-value lesser than or equal to 0.1 and a peptide length greater than five residues that were a substring of the known protein sequences of the standard mixture. If there were several copies of the same spectrum identification, we took a median of the retention times. Before we measured the performance of the retention time prediction models, we measured the quality of the retention times of the spectrum identifications. To do this, we calculated the standard deviation of the retention times for each peptide that was identified more than once. The average standard deviation was 1.36 min for the retention times at pH 10.0. In the second dimension, where a retention time represents the exact elution time of a peptide, the average standard deviation of retention times was 8.43 s.
The nested CV was performed in the following way: First the spectrum identifications were split randomly into five partitions. On four of the partitions we performed a 5-fold CV to find the best parameters of the learning machine (C, ν, and σ). Therefore, $\nu \in \{0.4 \cdot 1.2^i | i \in \{0, 1, 2\}\}$, and $\sigma \in \{0.2 \cdot 1.221055^i | i \in \{0, 1, \ldots, 21\}\}$. Since it is recommended to have the C values in the range of the maximal label [13], we had $C \in \{0.001, 0.01, 0.1, 1, 10, 100\}$ for the retention times at pH 10.0 and $C \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ for the retention times at pH 2.1 since the retention times at pH 10.0 were measured in minutes and the retention times at pH 2.1 were measured in seconds. Then, we trained on the four partitions with the best parameters of the 5-fold CV and measured the Pearson correlation between the observed and the predicted retention times on the residual fifth partition. This was done for every possible combination of the five partitions to get a mean performance. To exclude random effects introduced by the random partitioning of the data, we repeated the calculations five times with different random partitionings. The average Pearson correlation coefficient between predicted and observed retention times for the evaluation at pH 10.0 is 0.93 and 0.98 at pH 2.1. This means that the prediction of retention times works very well for both dimensions. The better performance for the second-dimension separation at pH 2.1 can be explained by the fact that we only collected fractions at pH 10.0 every minute. Although an exact measurement of the retention

times in the first dimension would increase the performance of the prediction methods, this is experimentally not feasible for off-line two-dimensional peptide separations.

Elimination of False Identifications by Retention Time Filters

To show the applicability of retention time filters, we conducted the following experiment on the standard mixture. We trained the retention time model on all peptides yielding spectra with a q-value lesser than or equal to 0.01. This data set contains 223 unique peptides. The retention times of these peptides in both separation dimensions and their corresponding sequences were utilized to perform SVR with the *POBK* function. Then we used the trained models to predict retention times for both dimensions for the whole data set, similar to Klammer *et al.* [55]. With the two models for retention time prediction, we could build a filter for each dimension as described in Section 3.2.2. Since the model for the first-dimension separation at pH 10.0 is slightly worse than the model for the second dimension at pH 2.1, we set the significance level of the retention time filter of this dimension to 0.01. This means that the probability of filtering out a correct identification is smaller than or equal to 0.01. The significance level for the filter of the second dimension was set to the standard value, which is 0.05. Since we knew which proteins were in the mixture and could therefore distinguish false positives from true positives, we were able to evaluate the performance of the filtering approach. Therefore, we measured the precision as described in Section 3.2.2 on all spectrum identifications having a q-value smaller than or equal to 0.01 and correspondingly for q-values 0.02, 0.03, 0.04, 0.05, 0.1, 0.15, and 0.2. The precision was measured for the data sets without filtering as well as with one of the filters or with both filters in combination. Fig. 3.9 shows that each filter improves the precision for every evaluated subset. Furthermore, it can be seen that the combination of both filters leads to the biggest improvement in precision. The numbers underlying the figure are shown in Table 3.5 which contains additional data for q-value thresholds 0.25, 0.3, ..., 0.5.
The complementarity of both filters is demonstrated by Fig. 3.10, which shows the number of correctly identified spectra with regard to precision. To calculate the underlying values, we took the precision of the different identification sets and evaluated for each data set how many spectra were identified correctly. It can be seen that both filters improve the number of correctly identified spectra. Moreover, the biggest improvement in the number of correctly identified spectra can be achieved for a combination of both filters. For example, at precision 0.94, meaning that 94% of the identifications are correct, one gets 1567 correctly identified spectra by using both filters compared to 1165 spectra without filtering. This corresponds to a 35% increase in peptide identifications at the same level of precision. The same precision of 0.94 is achieved for the spectrum identifications having a q-value lesser than or equal to 0.05 with additional filtering by our two-dimensional retention time filter or for all spectrum identifications with a q-value lesser than or equal to 0.01 without filtering.

q value threshold	unfiltered			filtered in 1st dimension			filtered in 2nd dimension			filtered in both dimensions		
	tp	fp	precision	tp	fp	precision	tp	fp	precision	tp	fp	precision
0.01	1165	70	0.943	1106	58	0.950	1165	64	0.948	1106	56	0.952
0.02	1345	100	0.931	1279	80	0.941	1342	85	0.940	1277	72	0.947
0.03	1468	130	0.919	1395	99	0.934	1464	101	0.935	1393	83	0.944
0.04	1577	159	0.908	1495	115	0.929	1569	117	0.931	1489	91	0.942
0.05	1663	183	0.901	1575	125	0.926	1653	128	0.928	1567	96	0.942
0.10	1962	393	0.833	1852	239	0.886	1942	221	0.898	1836	158	0.921
0.15	2104	598	0.779	1981	329	0.858	2078	292	0.877	1960	198	0.908
0.20	2230	807	0.734	2102	422	0.833	2198	339	0.866	2076	223	0.903
0.25	2315	1097	0.678	2185	553	0.798	2282	415	0.846	2158	267	0.890
0.30	2408	1360	0.639	2268	677	0.770	2366	475	0.833	2233	303	0.881
0.35	2512	1780	0.585	2367	877	0.730	2466	569	0.813	2328	356	0.867
0.40	2595	2562	0.503	2443	1166	0.677	2542	783	0.765	2401	444	0.844
0.45	2665	3368	0.442	2505	1523	0.622	2606	954	0.732	2458	535	0.821
0.50	2723	4044	0.402	2562	1809	0.586	2663	1132	0.702	2513	619	0.802

Table 3.5: ***Overview of precision depending on*** *q*-***value threshold and filtering:*** *This table shows the precision for different subsets of the data. Every row corresponds to one subset. The q-values of the spectrum identifications have to be smaller than or equal to the q-value threshold in the first column. tp stands for the number of true positives (correct hypotheses which are significant at the particular significance level) and fp stands for the number of false positives (false hypotheses which are significant at the particular significance level). The precision is defined as tp / (tp + fp).*

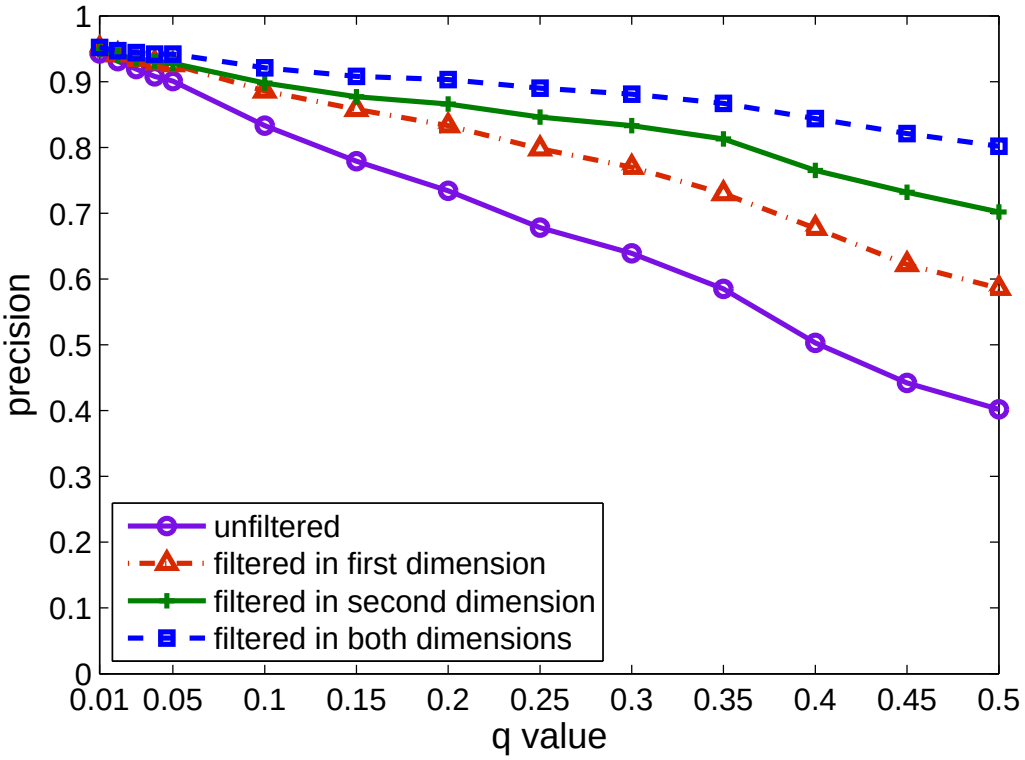

Figure 3.9: ***Comparison of precision depending on the q-value of the identifications with and without filtering:*** *This plot shows the precision for various data sets with and without filtering. At every point all spectrum identifications having a q-value smaller than or equal to the x-axis value are considered.*

To illustrate the filtering capabilities, we plotted the observed retention time against the predicted retention time for the identifications with q-value lesser than or equal to 0.05. Fig. 3.11 shows the performance of the filter for the separation at pH 10.0. It can be seen that the correlation between observed and predicted retention time is quite good for the correct identifications. The lines represent the 99% confidence intervals for the retention times predicted by our model for peptide separation at pH 10.0 (see Section 3.2.2 for details). Furthermore, one can see that there are false identifications which are filtered out only by the filter of the first dimension (crosses without circle). This effect can also be seen in Fig. 3.12, which demonstrates the performance of the filter for the second-dimension separation at pH 2.1. The correlation between observed and predicted retention time is even better than for the first retention time dimension.

Using RT Filters to Improve Identifications in Whole Proteome Analysis

The same protocol as above was applied to the *Sorangium cellulosum* data to obtain more identifications, keeping the precision at the same value. We did not train on all spectra with a q-value smaller than or equal to 0.01 since our learning method does not require such a large amount of training data [94]. Instead, we just used the 600 best-scoring identifications. We then utilized

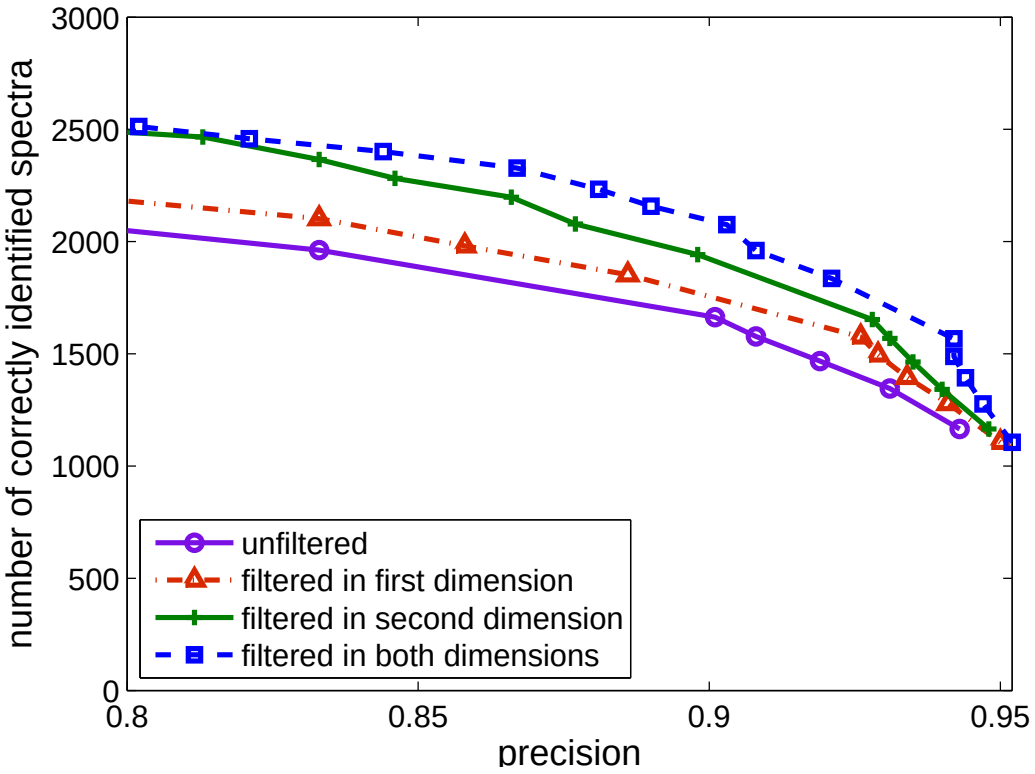

Figure 3.10: ***Comparison of correctly identified spectra with and without filtering:*** *This plot shows the number of correctly identified spectra with and without filtering. From right to left, the points correspond to the different partitions of the data which were evaluated. The first point is for all spectrum identifications with a q-value smaller than or equal to 0.01. The following points are for the q-values 0.02, 0.03, 0.04, 0.05, 0.1, 0.15, ..., 0.5 if the corresponding measured precision was larger than or equal to 0.8. The numbers underlying this figure can be found in Table 3.5 (precision vs tp).*

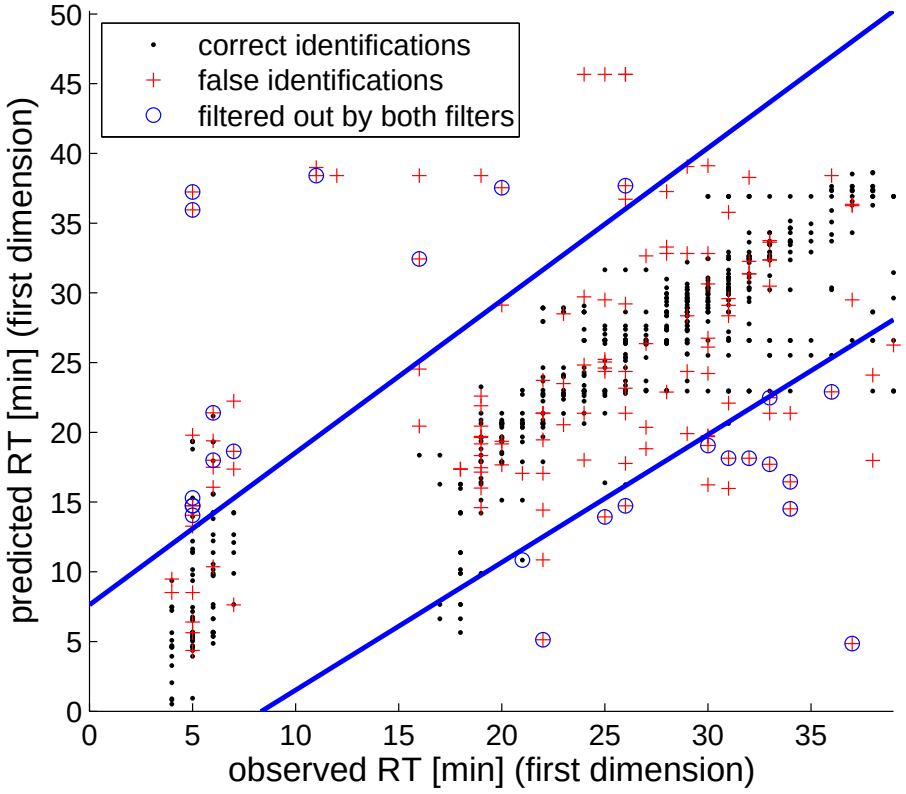

Figure 3.11: ***Filter performance of separation at pH 10.0:*** *This plot shows observed against predicted retention time (first dimension) for all spectrum identifications having a q-value which is lesser or equal to 0.05. The lines show the borders of the filter (at p-value 0.01). Every point which is not between the two lines is filtered out by this filter. Points having an extra circle are also filtered out by the filter of the second dimension (pH 2.1).*

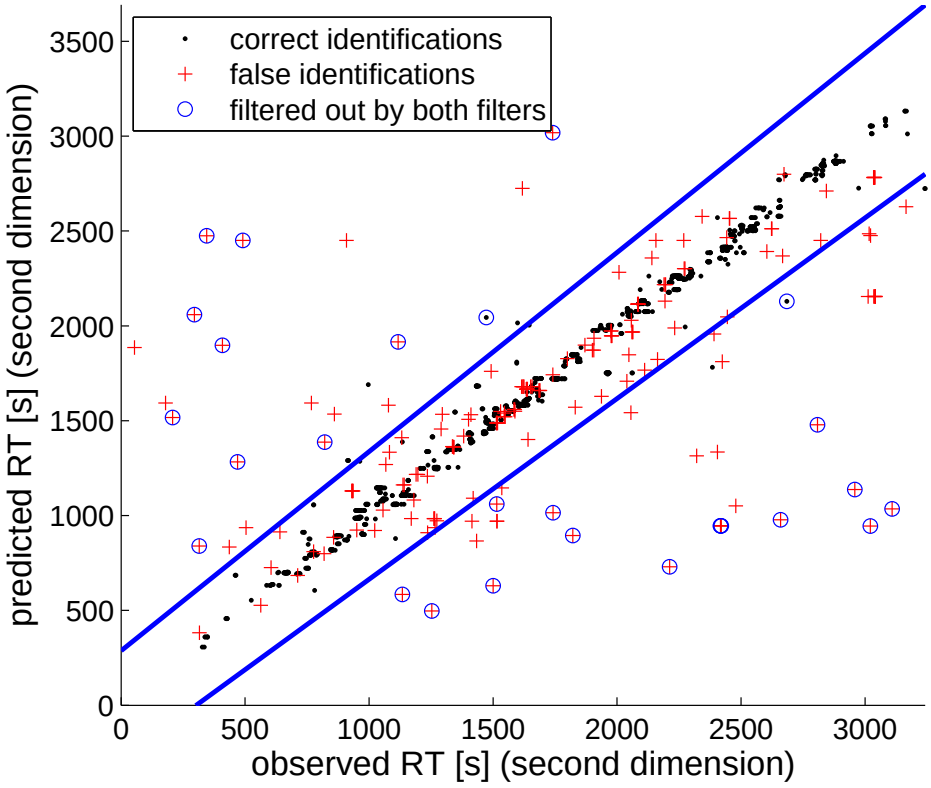

Figure 3.12: ***Filter performance for separation at pH 2.1:*** *This plot shows observed against predicted retention time (second dimension) for all spectrum identifications having a q-value which is lesser or equal to 0.05. The lines show the borders of the filter (at p-value 0.05). Every point which is not between the two lines is filtered out by this filter. Points having an extra circle are also filtered out by the filter of the first dimension (pH 10.0).*

the trained models to predict retention times in both dimensions for the whole data set of mass spectrometrically identified *S. cellulosum* peptides. The study on the standard mixture showed that one can achieve similar precision by choosing all spectrum identifications with a q-value smaller than or equal to 0.01 without filtering or choosing all spectrum identifications with a q-value smaller than or equal to 0.05 and filtering by our two retention time filters. Since, in the whole proteome, we can not directly distinguish between true and false positive identifications, we evaluated the total number of identified spectra and the number of identified unique peptides for these two sets of identification parameters. At a q-value of 0.01 we annotated 21,038 spectra which identified 6,202 unique peptides, and at a q-value of 0.05 with additional RT-filtering we annotated 25,347 spectra, which yielded 7115 unique peptide identifications. This represents an increase in the number of successful peptide identifications by 15% without any loss in the precision of peptide identifications. In this evaluation, peptides with the same amino acid sequence but different post-translational modifications were considered to be different peptide identifications. Furthermore, we looked at the overlap of the unique peptide identifications between the two sets. The majority of identifications are part of both sets. Nevertheless, there are 720 unique peptide identifications in the unfiltered set and 1,633 unique peptide identifications in the filtered set. This means that one can get 1,633 or more than 26% more unique peptide identifications by combining the identifications of these two sets compared to the number of identifications one gets by just taking the unfiltered identifications. The numbers are plotted as a Venn diagram in Fig. 3.13.

3.2.4 Conclusions

We present a new approach to improve the number of correctly identified spectra resulting from mass spectrometry experiments by using experimental data that are inherent to the analytical process. We are able to build retention time predictors for a two-dimensional chromatographic separation using the retention times of peptides identified with high confidence by tandem mass spectrometry. Thus, no additional calibration using standard samples was necessary. The retention time filters were successfully applied to filter out false positive identifications. Moreover, we show that the scoring threshold can be lowered to include more previously false negatives (and to get more correct spectrum identifications) at the same level of precision in terms of correct identifications. This is accomplished by incorporating the retention time predictors into a two-dimensional filter which removes many false positive identifications. Therefore, we can achieve the same rate of precision although the mass spectrometric scoring threshold is smaller. The method was validated on a standard protein mixture. Finally, we applied the same method to the whole proteome analysis of the *Sorangium cellulosum* bacteria. The analysis showed that by using this method we can find about 26% more unique spectrum identifications.
It would be interesting to apply this two-dimensional filtering to data from other two-dimensional chromatographic separation techniques. We already

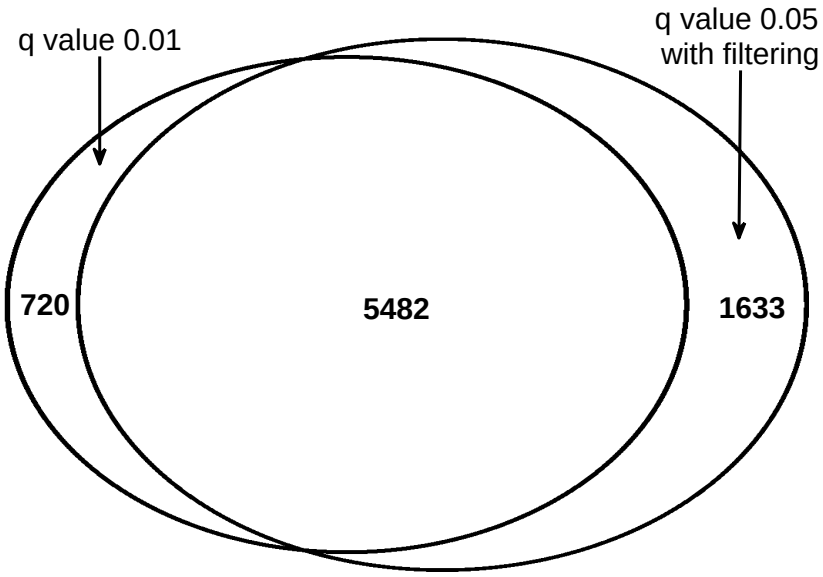

Figure 3.13: ***Increase in unique identifications on the Sorangium cellulosum data set:*** *This plot shows the number of unique spectrum identifications of two sets, for which the precision is estimated to be equal (based on empirical results on standard mixture). In the first set are all unfiltered identifications with a q-value smaller than or equal to 0.01 and in the second set are all spectrum identifications having a q-value smaller than or equal to 0.05 which are not filtered out by any of the two retention time filters. By combining the identifications one gets 7,835 instead of 6,202 unique peptide identifications.*

showed in Section 3.1, that the POBK can be used to predict separation in strong anion exchange chromatography. Therefore, it is very likely, that a two-dimensional filter can be built for data measured by a combination of strong cation exchange (SCX) chromatography and reversed-phase chromatography (see Section 2.2.2).

3.3 Prediction of Proteotypic Peptides

3.3.1 Introduction

The two main goals in computational proteomics are identification and quantitation of all proteins in a protein mixture. Unfortunately, in nearly every mixture, there are highly abundant proteins as well as low-concentration proteins. This creates the problem that high-abundance proteins are identified several times but those in low abundance are often missed. As explained in Section 2.2.4, the highest peaks of the MS1 spectrum are chosen for fragmentation in the second stage of the mass spectrometer. Only those peptides chosen for fragmentation can be identified by the instrument. This is one of the reasons why certain peptides have a higher likelihood of being detected by the instruments [63]. Kuster *et al.* [63] showed that certain peptides of a protein can be identified more often than others of the same protein. They called peptides that are experimentally observable and uniquely identify a protein or protein isoform *proteotypic* peptides. In their study, they suggested that instead of trying to measure all possible peptides, one should concentrate on the proteotypic peptides of the proteins of interest for better reproducibility of the results.
In the same paper, they introduced a database called PeptideAtlas, which was intended as a resource for experimenters to obtain and store peptide identifications. If the database contained measurements for all proteins, one could look-up the proteotypic peptides for the proteins of interest and limit the analysis to this small part of the whole peptide space. Unfortunately, measurements are very time-consuming and costly and the number of different proteins is large. For newly sequenced genomes, the full proteome is still to be presented. Furthermore, peptides which are observable by a certain type of experimental design (e.g., LC-ESI-MS/MS) may be unobserved by another experimental design (e.g., PAGE-MALDI-TOF/TOF).
To be able to measure proteins for which no experimental data of proteotypic peptides is available, computational tools for the prediction of proteotypic peptides are needed. Tang *et al.* [125] presented the first method for the prediction of proteotypic peptides, but methods for prediction of proteotypic peptides were also introduced in the work of Mallick *et al.* [73], Lu *et al.* [70], and Webb-Robertson *et al.* [140]. All methods have in common that they use standard physico-chemical features together with standard learning techniques to build the predictor. Unfortunately, none of the groups compared their method to the methods of any other group. This complicates the choice for the researchers. In this work, we introduce two new predictors of proteotypic peptides based on the *OBK* or the *POBK* and an SVM. We compare the performance of each predictor on the dataset of Mallick *et al.* to an SVM, which uses the same features as introduced by Mallick *et al.* [73] and Lu *et al.* [70] benchmarked in [40]. In this comparison, our methods perform significantly better than the other methods, although they do not contain any specialized features. Furthermore, we investigate which properties of a peptide make it proteotypic. Therefore, we first analyze the different datasets by standard approaches and afterwards, visualize which

amino acids are important for the resulting classifiers. This analysis shows that positive and negative amino acids strongly determine detectability of a peptide. For MALDI measurements, we can also support the hypothesis that aromatic amino acids [71] contribute positively to peptide detectability. Furthermore, we support the hypothesis that an arginine at the C-terminal end of the peptide contributes more to peptide detectability than a lysine for MALDI experiments [59].

3.3.2 Methods and Data

Data

We used four different datasets for performance evaluation of methods for proteotypic peptides prediction. The datasets were introduced by Mallick *et al.* [73] and contain measurements of yeast proteins on four different platforms. For each platform it consists of a set of proteotypic peptides and a set of non-observed peptides, which are neither substrings nor do they contain substrings of proteotypic peptides. Because of the different measurement platforms, the datasets are named as follows:

- ICAT-ESI: This dataset is measured by ICAT labelling with LC-ESI-MS/MS.
- MudPIT-ESI: This dataset is measured by a combination of MudPIT and ESI-MS/MS.
- PAGE-ESI: This dataset is measured by one-dimensional (1D) gel electrophoresis followed by LC-ESI-MS/MS.
- PAGE-MALDI: This dataset is measured by 1D gel electrophoresis followed by MALDI.

All sequences also contain the flanking residue at the N- and C-terminal ends because Mallick *et al.* had better prediction results by including them. This means that the second amino acid of the sequence is the residue at the N-terminal end and the second to last amino acid is the residue at the C-terminal end of the peptide.

Visualization of Important Amino Acids

One common argument against applying machine learning approaches is the lack of interpretability of the results. The machine learning algorithm is in these cases called a "Black Box". Especially for Support Vector Machines this reasoning was often an argument against using string kernels although there exist approaches to elucidate the importance of certain k-mers regarding SVM classification [144, 76, 118]. Since we use the *POBK* and the *OBK*, introduced in Section 3.1.2, with k-mer length one for classification of proteotypic peptides, we visualize the discriminant similar to Meinicke *et al.* [76]. POIMs of Sonnenburg *et al.* [118] just improve the visualization performance for k-mer lengths greater than one.
As introduced in Section 3.1.2, the feature map of the *OBK* is defined as:

$$\Phi(s) = [\mu^L_{\omega_1}(t), \ldots, \mu^L_{\omega_{|\mathcal{A}^k|}}(t), \mu^R_{\omega_1}(t), \ldots, \mu^R_{\omega_{|\mathcal{A}^k|}}(t)]^T$$

in which the ω_i are the different k-mers, L and R are used for the left and the right borders, and the μ_{ω_i} functions are the oligo functions of the corresponding k-mer. One can visualize the training data by weighting them with the α_i from the SVM and summing them for each position and each oligo. If one is, for example, interested in the contribution of the amino acid proline (P) at position five in the left border, the importance value is calculated by:

$$\sum_{i=1}^{n} \alpha_i \mu i^L_P(5).$$

Given a position t and an oligo ω, the importance weight w is calculated by:

$$w^{L|R}_{\omega}(t) = \sum_{i=1}^{n} \alpha_i \mu i^{[L|R]}_{\omega}(t).$$

The weight values can then be computed for every position-oligo combination and the resulting matrix can be visualized by interpreting the weights as color values. The weight matrix for the *POBK* can be computed analogously. A low (high) weight corresponds to a signal, which can be more often found in sequences which are predicted negative (positive). Therefore, the image of the weight matrix allows direct interpretation of the discriminant learned by the SVM.

3.3.3 Results and Discussion

Performance Evaluation of Different Predictors

This evaluation compares the performance of the *OBK* and *POBK* to the performance of the features introduced by Mallick *et al.* [73] and Lu *et al.* [70] presented in reference [40]. For this purpose, we chose an SVM as the classifier and trained it using the *OBK* and *POBK*.
The datasets contain many more negative than positive samples. We chose the datasets exactly as in [40]. To transparently access the performance of the different approaches, we only used balanced datasets for the evaluation. Furthermore, we wanted to compare the performances across datasets. Therefore, all datasets had to contain the same number of samples, which is why the number of training samples from each class is 697: this is the number of positive samples in the smallest dataset. Thus, every evaluation dataset contained 697 positive samples and 697 negative samples. The samples were chosen randomly if a dataset contained more than 697 of them, so as not to introduce a bias in the evaluation due to sampling. This was done ten times to get a mean performance value. The performance was measured by 5-fold cross-validation and the performance measure was the classification rate. The results of the comparison are shown in Table 3.6. It can be seen that the *POBK* performs better than all other methods on nearly all datasets except the PAGE-MALDI dataset, where the features of Lu *et al.* [70] lead to similar performance results.

Dataset	Mallick *et al.*	Lu *et al.*	OBK	POBK
MudPIT-ESI	0.82 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	**0.85** ± 0.01
PAGE-ESI	0.83 ± 0.01	0.84 ± 0.01	0.86 ± 0.01	**0.87** ± 0.01
ICAT-ESI	0.81 ± 0.01	0.83 ± 0.01	0.83 ± 0.01	**0.84** ± 0.01
PAGE-MALDI	0.86 ± 0.01	**0.88** ± 0.01	0.87 ± 0.01	**0.88** ± 0.01

Table 3.6: ***Comparison of classification rates for proteotypicity prediction:*** *This table shows the classification rates and standard deviations of the different approaches for predicting proteotypic peptides. The column labeled Mallick et al. represents the approach with the features of Mallick et al. and the column labeled Lu et al. represents the approach with the features of Lu et al. as presented in [40]. Columns labeled* ***OBK*** *and* ***POBK*** *represent Oligo-Border Kernel and the Paired Oligo-Border Kernel, respectively.*

Analysis of Different Datasets with Standard Statistics

The goal of this study was not only to come up with the best predictor for proteotypic peptides, but also to elucidate how proteotypic peptides differ from non-proteotypic peptides. Therefore, we first analyzed the datasets according to the log of the ratio of the number of positively charged amino acids to the number of negatively charged amino acids. This means that we calculated the ratio for each peptide in each dataset and plotted boxplots for the log of the ratios for the proteotypic peptides and for the non-observed peptides for each dataset. The boxplots can be seen in Figs. 3.14 and 3.15. A general trend is that non-observed peptides contain more positively charged amino acids than negatively charged ones. Furthermore, it can be seen that proteotypic peptides contain more negatively charged amino acids than positively charged amino acids for the ICAT-ESI and the MudPIT-ESI datasets, whereas this trend cannot be observed for the PAGE-ESI and the PAGE-MALDI datasets (median of the log of the ratios is equal to zero).

Analysis of Proteotypic Peptides by Two Sample Logo

To further investigate the properties of proteotypic peptides, we analyzed the datasets with the two sample logo method by Vacic *et al.* [134]. Given a multiple sequence alignment (MSA) for a positive set of sequences and an MSA of negative sequences, the method can be used to find enriched and depleted signals in the positive MSA. Since peptides usually differ in length, we aligned all peptides at the C-terminus and stripped off the flanking residues. This implicitly assumes that signals are distributed equally over the sequence independent of the length of the peptide. This assumption might be too strong, but nevertheless one can gain a first insight into the importance of amino acids at certain positions. The *OBK* and *POBK* are also based on a similar assumption, but by the positional smearing by the parameter σ, the bias to certain positions introduced by the alignment can be reduced. Furthermore, *OBK* and *POBK* consider the alignements from both termini. A general trend which can be found in all two sample logos is that the positively charged amino acids lysine (K) and arginine (R) are depleted in the set of proteotypic peptides. Fig. 3.16 shows this trend for the PAGE-ESI

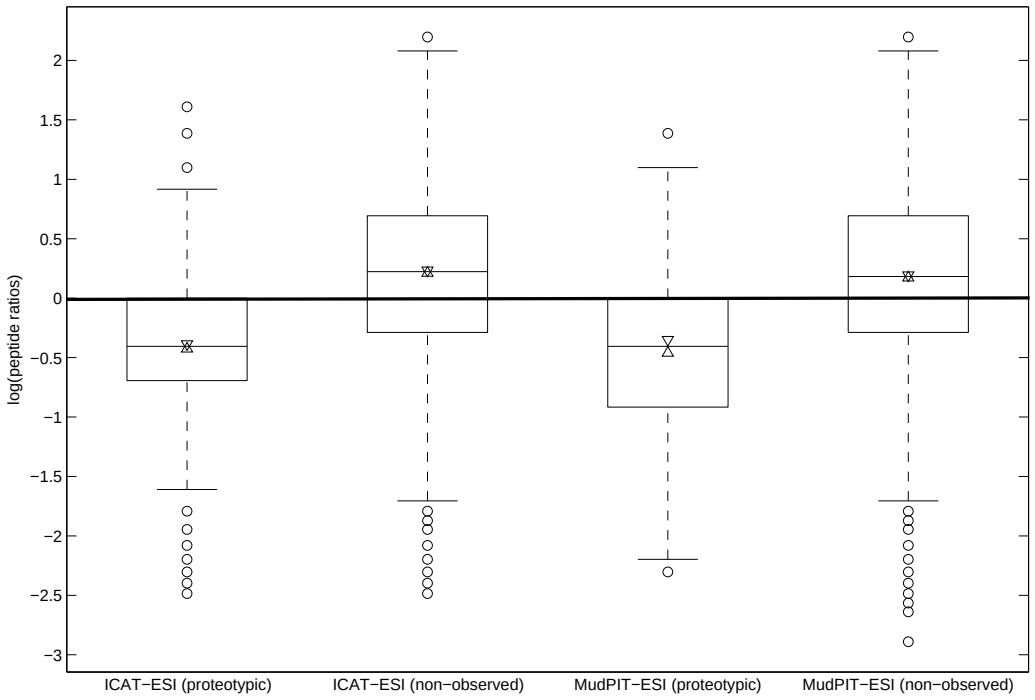

Figure 3.14: ***Boxplot of amino acid ratios for the MudPIT-ESI and the ICAT-ESI datasets:*** *This plot shows a boxplot of the log of the ratios of the number of positively charged amino acids to the number of negatively charged amino acids per peptide for peptides of the MudPIT-ESI and the ICAT-ESI datasets. According to the T-test and the Kolmogoroff-Smirnov test, the distributions of the ratios of the proteotypic and the non-observed peptides are significantly different (p-value* < 0.01*) for both datasets.*

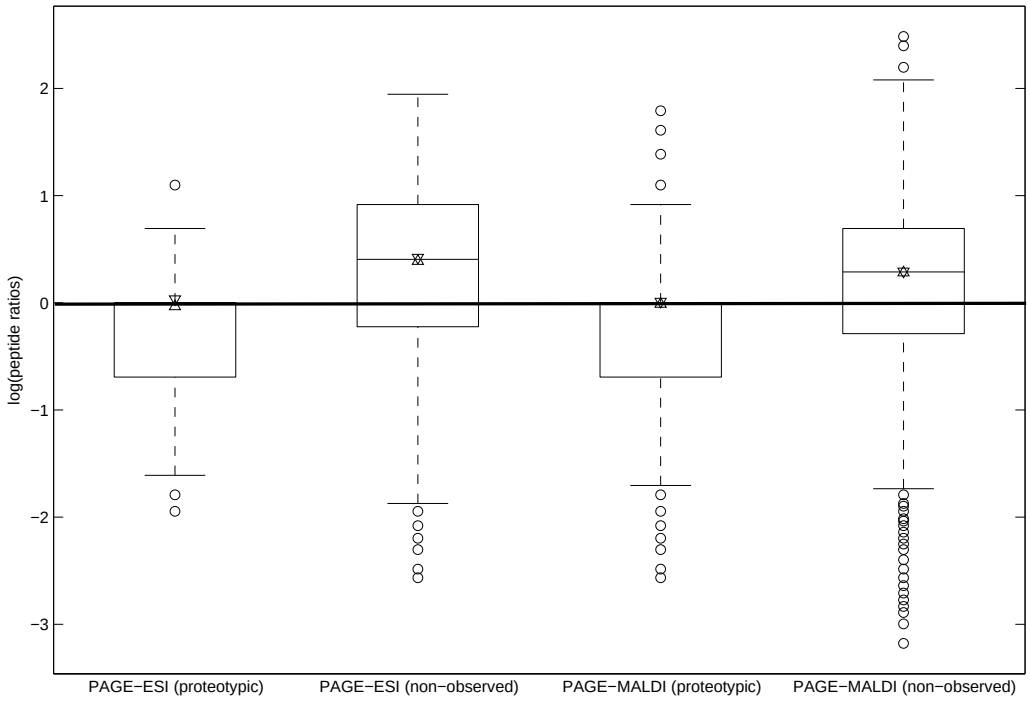

Figure 3.15: ***Boxplot of amino acid ratios for the PAGE-ESI and the PAGE-MALDI datasets:*** *This plot shows a boxplot of the log of the ratios of the number of positively charged amino acids to the number of negatively charged amino acids per peptide for peptides of the PAGE-ESI and the PAGE-MALDI datasets. According to the T-test and the Kolmogoroff-Smirnov test, the distributions of the ratios of the proteotypic and the non-observed peptides are significantly different (p-value* < 0.01*) for both datasets.*

dataset. This could be an artifact of the dataset generation of the non-observed peptides. Therefore, we restrict the following two sample logo analysis to fully tryptic peptides (without missed cleavages). For the ICAT-ESI dataset, it can be seen in Fig. 3.17, that the negatively charged amino acid aspartate (D) is enriched. This observation fits to the analysis in Section 3.3.3, in which we found that there are more negatively charged amino acids for the proteotypic peptides of the ICAT-ESI dataset. This fact cannot be seen for the MudPIT-ESI dataset in Fig. 3.18, although the analysis in Section 3.3.3 suggested the same trend for this dataset. An explanation could be, that the enrichment is not as significant as for the ICAT-ESI dataset. Since we use the two sample logo method with Bonferroni correction, which is very conservative, some weaker enrichments are hidden. The two sample logo without Bonferroni correction in Fig. 3.19 shows an enrichment of negatively charged amino acids for the proteotypic peptides of the MudPIT-ESI dataset. For the PAGE-MALDI dataset in Fig. 3.21 arginine at the C-terminus seems to be highly enriched in the set of proteotypic peptides. This observation was also reported by Krause *et al.* [59]. The authors hypothesized, that this is due to the chemical properties of arginine because of the basicity of the guanidino functionality of the arginine side chain which might result in better ionization in the liquid and/or gas phase. At the C-terminal end of tryptic peptides, there can only be an arginine or lysine. Therefore, an enrichment of arginine at this position implies a depletion of lysine. An enrichment of arginine at the C-terminal end can also be seen for the PAGE-ESI dataset in Fig. 3.20, although the enrichment is not as strong as for the PAGE-MALDI dataset (41.7% compared to 7.5%). Additionally, an enrichment for alanine and valine can be seen.

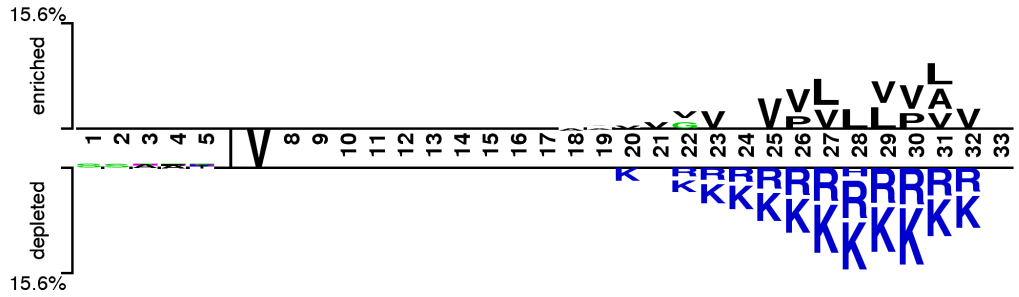

Figure 3.16: ***Two sample logo for PAGE-ESI dataset:*** *This plot shows the two sample logo [134] of the MSAs of the proteotypic and the non-observed peptides for the PAGE-ESI dataset. Amino acids which are enriched in the proteotypic peptides are shown at the top and depleted amino acids are shown at the bottom. The numbers refer to the positions of the amino acids in the peptide and all peptides are aligned to the C-terminal end without flanking residues.*

Visualization of Important Amino Acids

We visualized one of the ten random draws for each dataset with the methods introduced in Section 3.3.2. The visualization results for the *POBK* compared to the *OBK* were in all experiments very similar. An example for the

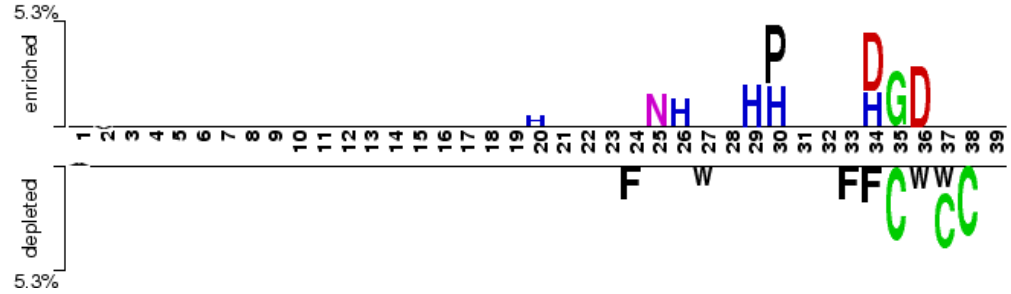

Figure 3.17: ***Two sample logo for ICAT-ESI dataset:*** *This plot shows the two sample logo [134] of the MSAs of the proteotypic and the non-observed peptides for the ICAT-ESI dataset. All peptides are aligned to the C-terminal end without flanking residues. Only fully tryptic peptides are used (no missed cleavages).*

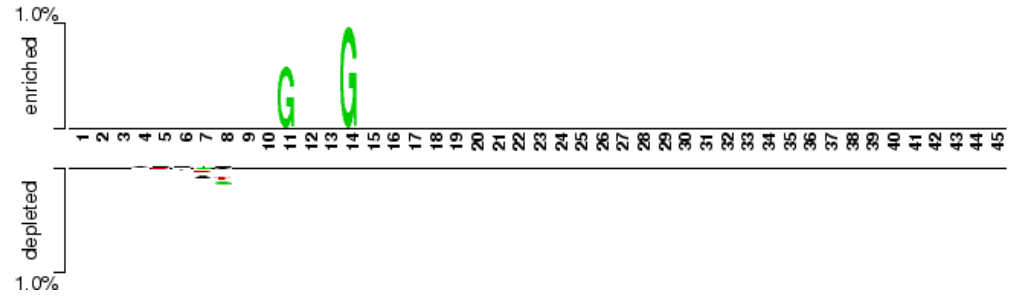

Figure 3.18: ***Two sample logo for MudPIT-ESI dataset:*** *This plot shows the two sample logo [134] of the MSAs of the proteotypic and the non-observed peptides for the MudPIT-ESI dataset. All peptides are aligned to the C-terminal end without flanking residues. Only fully tryptic peptides are used (no missed cleavages).*

Figure 3.19: ***Two sample logo for MudPIT-ESI dataset (without Bonferroni correction):*** *This plot shows the two sample logo [134] of the MSAs of the proteotypic and the non-observed peptides for the MudPIT-ESI dataset without Bonferroni correction. All peptides are aligned to the C-terminal end without flanking residues. Only fully tryptic peptides are used (no missed cleavages).*

MudPIT-ESI dataset can be seen in Figs. 3.22 3.23. Since the visualization of the *OBK* allows investigation of both borders separately, we only present the *OBK* visualizations for the other datasets in Figs. 3.24 3.26, and 3.25. A general trend which can be observed for all datasets, is that arginine or lysine near the peptide ends have very negative weights, with the sole exception of arginine at the C-terminal end for the PAGE-MALDI dataset. Furthermore,

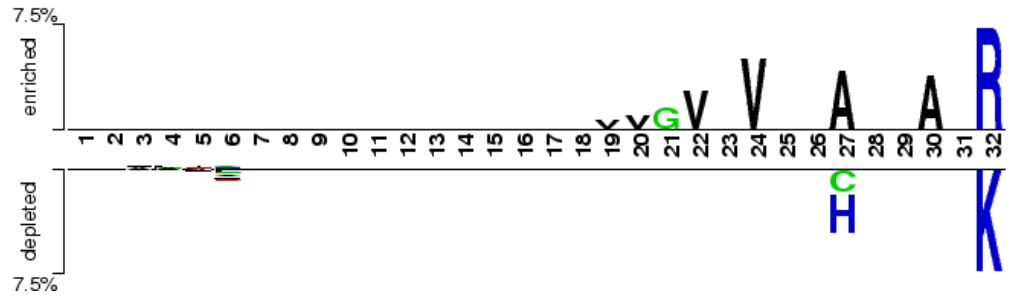

Figure 3.20: ***Two sample logo for PAGE-ESI dataset:*** *This plot shows the two sample logo [134] of the MSAs of the proteotypic and the non-observed peptides for the PAGE-ESI dataset. All peptides are aligned to the C-terminal end without flanking residues. Only fully tryptic peptides are used (no missed cleavages).*

Figure 3.21: ***Two sample logo for PAGE-MALDI dataset:*** *This plot shows the two sample logo [134] of the MSAs of the proteotypic and the non-observed peptides for the PAGE-MALDI dataset. All peptides are aligned to the C-terminal end without flanking residues. Only fully tryptic peptides are used (no missed cleavages).*

aspartate and glutamate near the borders of the peptide seem to be positive for peptide detectability. Since arginine and lysine are positively charged and aspartate and glutamate are negatively charged, this could be a general property and further supports the analysis in Section 3.3.3. Mallick *et al.* [73] also identified positive charge (total count or average) to be one of the five most important features for proteotypicity prediction in each dataset.
The plots for the ICAT-ESI (Fig. 3.24) and the MudPIT-ESI (Fig. 3.22) datasets are very similar, because both measurements use LC-ESI-MS/MS. Furthermore, these datasets show a stronger positive effect for glutamate in the left border than in the right border, which shows that the visualization of the *OBK* can provide more insights than that of the *POBK*. The visualization of the discriminant of the PAGE-ESI dataset shows a weaker positive effect of aspartate. Additionally, the aliphatic amino acids isoleucine, leucine, and valine seem to contribute positively. A very interesting observation for the PAGE-MALDI dataset, shown in Fig. 3.25, is that amino acids with aromatic side chains seem to contribute positively to peptide detectability. The positive effect of aromatic amino acids in MALDI experiments was also presented in [71]. It can be assumed that peptides with aromatic amino acids can interact better with the matrix and are therefore better ionizable. Furthermore, the classifier was able to find the positive arginine signal at the

C-terminal end, which was also found by the two sample logo analysis and reported by Krause *et al.* [59].

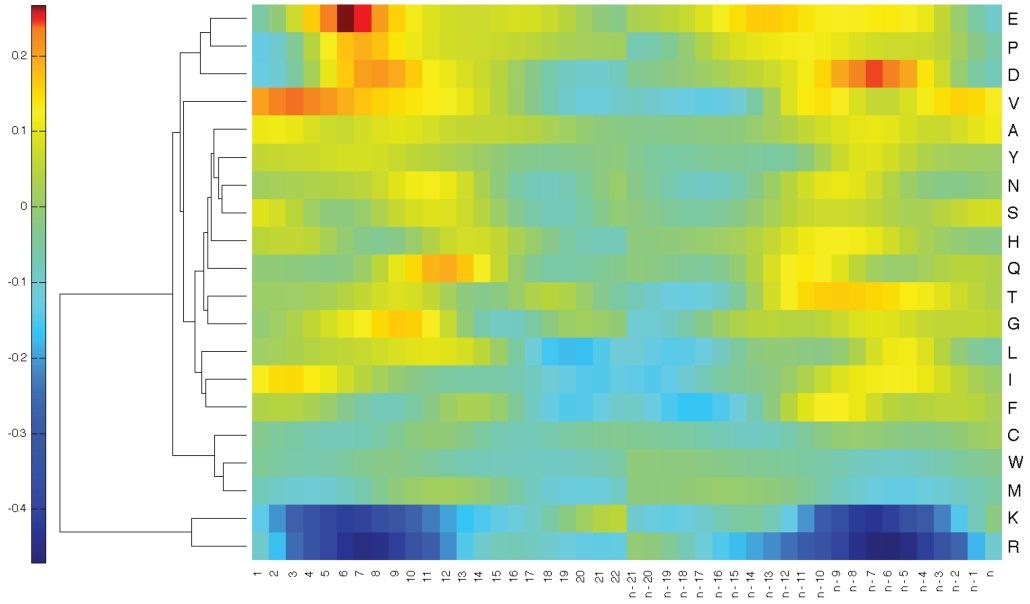

Figure 3.22: ***Visualization of important positions for MudPIT-ESI dataset (OBK):*** *This plot shows the visualization of the importance weights of the OBK classifier calculated as described in Section 3.3.2. The first 22 positions correspond to the primal representation of the left border of the peptide and the remaining positions correspond to the primal representation of the right border of the peptide (n is the position of the amino acid at the C-terminal end of the peptide).*

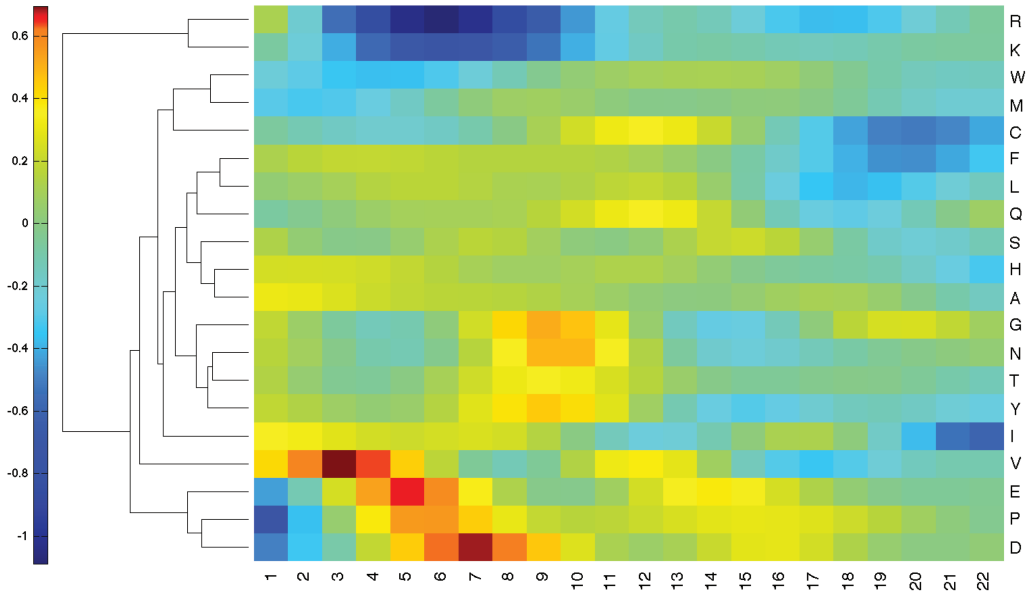

Figure 3.23: ***Visualization of important positions for the MudPIT-ESI dataset (POBK):*** *This plot shows the visualization of the importance weights of the POBK classifier calculated as described in Section 3.3.2. Since the POBK looks at the signals in both borders simultaneously, a positive weight for position i corresponds to the amino acids at position i and position $n - i + 1$.*

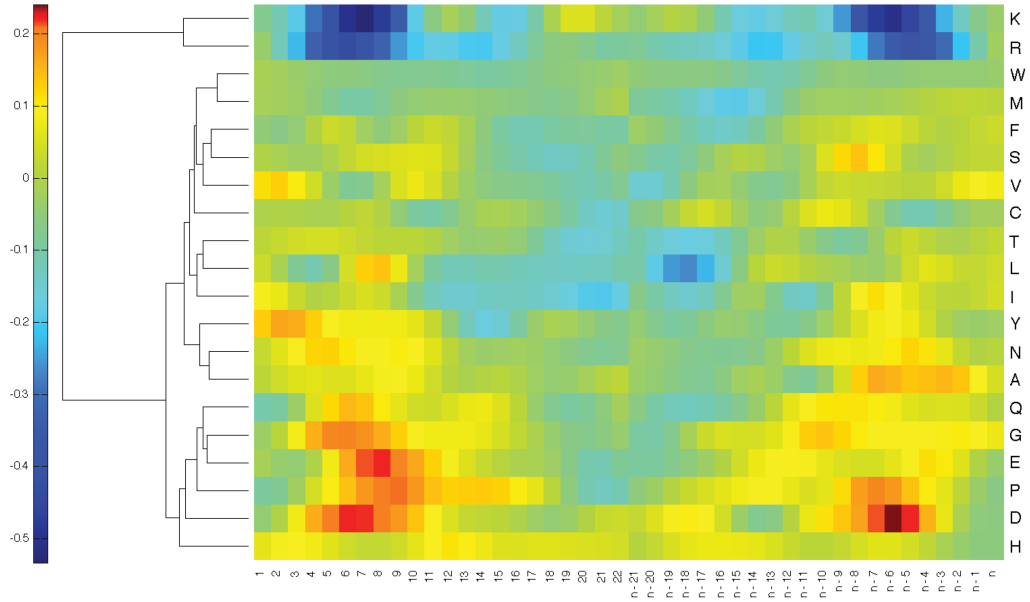

Figure 3.24: ***Visualization of important positions for the ICAT-ESI dataset (OBK):*** *This plot shows the visualization of the importance weights of the OBK classifier calculated as described in Section 3.3.2. The first 22 positions correspond to the primal representation of the left border of the peptide and the remaining positions correspond to the primal representation of the right border of the peptide (n is the position of the amino acid at the C-terminal end of the peptide).*

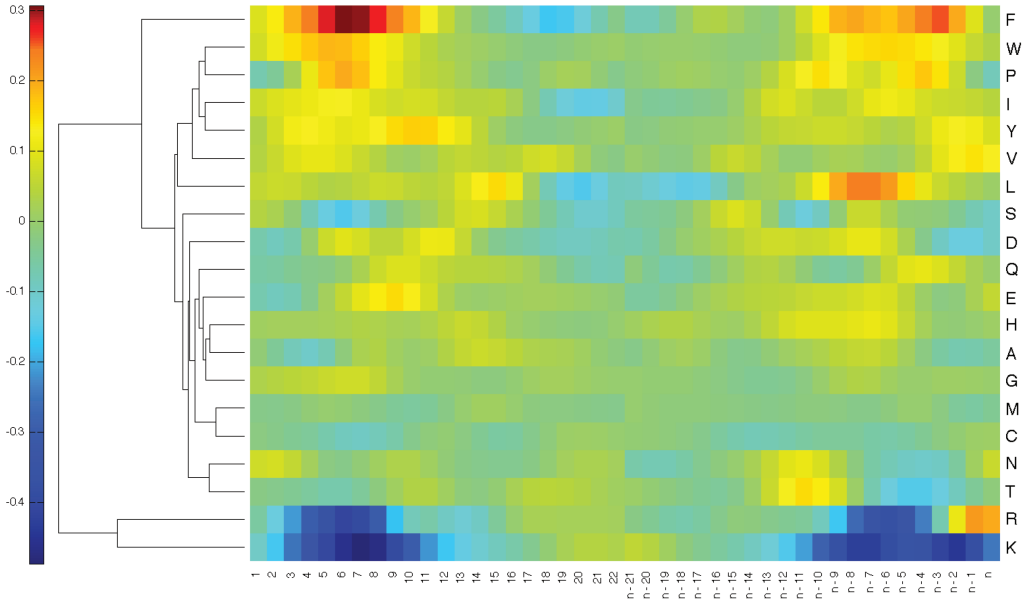

Figure 3.25: ***Visualization of important positions for the PAGE-MALDI dataset (OBK):*** *This plot shows the visualization of the importance weights of the OBK classifier calculated as described in Section 3.3.2. The first 22 positions correspond to the primal representation of the left border of the peptide and the remaining positions correspond to the primal representation of the right border of the peptide (n is the position of the amino acid at the C-terminal end of the peptide).*

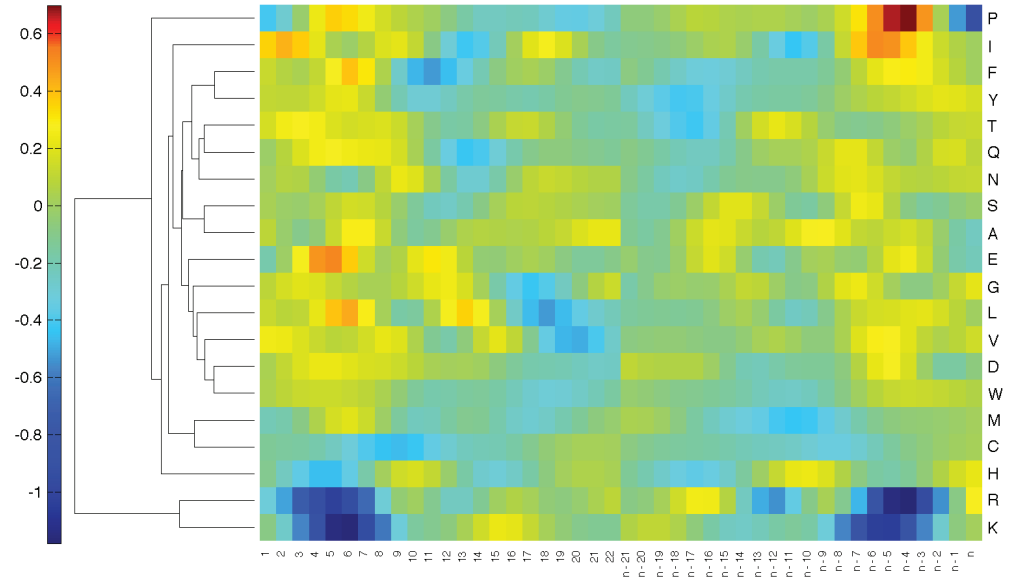

Figure 3.26: ***Visualization of important positions for the PAGE-ESI dataset (OBK):*** *This plot shows the visualization of the importance weights of the OBK classifier calculated as described in Section 3.3.2. The first 22 positions correspond to the primal representation of the left border of the peptide and the remaining positions correspond to the primal representation of the right border of the peptide (n is the position of the amino acid at the C-terminal end of the peptide).*

3.3.4 Conclusions

We show in this section that the *POBK* as well as the *OBK* can be used to predict proteotypic peptides with high accuracy. Our kernel function in combination with a support vector machine performs significantly better than the features of other groups together with a support vector machine and a standard kernel. Furthermore, we show that amino acids have different effects on detectability depending on the experimental setup. Similar to Mallick *et al.* [73], we find that positively charged amino acids help distinguish between proteotypic and non-observed peptides. Additionally, we can show in our analysis that the classifier discovers interesting properties concerning the underlying biochemical mechanisms of the measurement processes. One example are aromatic amino acids, which seem to contribute positively to peptide detectability in PAGE-MALDI experiments. This was also reported by other groups [71]. Another observation is that arginine at the C-terminus seems to increase detectability, compared to lysine, in PAGE-MALDI experiments. This observation is found by computing the two sample logo as well as with our analysis of the learnt classifier and is supported by a recent study [59]. Consequently, our method for peptide detectability has state-of-the-art performance and allows direct interpretation of the learnt classifier to provide interesting insights.

It would be very intersting to extend this binary prediction problem to a regression problem, in which we could predict the degree of proteotypicity. We included the proteotypicity prediction into the LC-MSsim [113], which can be used to simulate LC-MS maps. Therefore, we used the probability estimates of the libSVM to compute the likelihood of a peptide to be proteotypic. To train the proteotypicity predictor, we just used binary data (peptide is proteotypic or not). Extending the proteotypicity prediction to a regression problem would mean that every peptide gets a label representing its proteotypicity. This could be, for example, the intensity of each peptide feature normalized by the amount of protein present in the sample. A peptide feature is the three-dimensional shape (m/z, RT, and intensity) of the peptide measurement, defined by the isotopic distribution as well as the elution profile of the peptide. An accurate predictor for peptide feature intensity could be used to predict the absolute amount of a protein in a sample.

Chapter 4

Applications in Immunomics

4.1 Introduction

The adaptive immune system is one of the most advanced and most important systems in humans. It can direct immune responses according to various kinds of invading microorgansims and even recognize and destroy tumor cells [130]. The main components of the immune system were introduced in Section 2.3. MHCII presents peptides originating from the outside of the cell. There are various different MHCII alleles which have very specific sets of peptides to which they can bind. At present there are more than 750 unique MHCII alleles known [102] (regarded on the protein sequence level), but for less than 3% of them sufficient experimental data to construct a predictor is available. Since every human has at most twelve different MHCII alleles, it is very important for vaccine design to know which peptides can bind to the particular alleles. A good predictor for MHC peptide binding can reduce the number of possible peptides and therefore save a lot of time - and money-consuming wetlab experiments.
In contrast to MHCI, the ends of the binding clefts of the MHCII are open. This is why the length of the binding peptides varies significantly (from 8 to more than 30 amino acids). Nevertheless, analyses of MHCII structures revealed that the part of the peptide responsible for binding to MHCII is usually nine amino acids long. This part is also called binding core of the peptide. For most of the experimental data it is unknown which part of the peptide actually is the binding core, which complicates the problem of MHCII peptide binding prediction compared to MHCI peptide binding prediction. The binding clefts of MHCI are closed at the ends and the binding peptides have a length between eight and twelve. There are various methods for MHCII peptide binding prediction for alleles for which there exists sufficient experimental data. Some of these models are based on positional scoring matrices [9, 80, 96, 99, 116, 124], others use Gibbs samplers [81] or hidden Markov models [82]. Further works have used the ant colony search strategy [49], artificial neural networks [8], partial least squares [36], evolutionary algorithms [95] or support vector machines with standard kernel functions [24, 105, 137]. Very recently Wang *et al.* [138] combined several of these predictors to build a new predictor. There have also been efforts to

improve binding prediction by using structural information [143].
To the best of our knowledge all but two of the models for MHCII peptide binding prediction are based on experimental data for the particular alleles for which the predictions are for. The models of Singh *et al.* [116] and Zaitlen *et al.* [143] are the only methods which were shown to predict binding for alleles without training on them. However, the model by Singh *et al.* is only applicable to 51 alleles [138] which is about 7 % of all known alleles and Zaitlen *et al.* require three-dimensional structures of a similar allele to perform this kind of predictions which limits the number of alleles accessible through the method. Since the experimental data for peptide-MHCII binding is very scarce, we introduce a method to predict peptide binding for alleles, for which few or no experimental data is available. Similar ideas have also recently been introduced for MHCI predictions, although based on different machine learning techniques and for a far simpler problem (MHCI peptides have more or less identical lengths) [21, 44, 79].
We use similarities of the binding pockets of the alleles to build predictors for alleles, which do not need experimental data of the target allele to reach good prediction performance. The similarities are incorporated into the predictions using a specialized kernel function, which is based on the normalized set kernel by Gärtner *et al.* [33]. Therefore, the problem is transformed into a multiple instance learning problem [22]. The predictor is trained using the kernel function and Support Vector Regression (SVR) [110]. Using this method we are for the first time able to build predictors for about two thirds of all MHCII alleles. Assessment of their quality in blind predictions for alleles with known data reveals that the predictions are of sufficient quality for use in vaccine design. Furthermore, we show that our transformation of the problem into the multiple instance learning problem enables us to build predictors which perform equally well or even better than the best methods for MHCII peptide binding prediction.

4.2 Methods and Datasets

4.2.1 Multiple Instance Learning

In standard supervised binary classification, the associated label for every instance out of the sets of training samples is known. The input space $\mathcal{X}$ is usually a Hilbert space. Every instance can be represented as (x_i, y_i) where $x_i \in \mathcal{X}$ and $y_i \in \{-1, 1\}$. We define the set of positive training examples as $S_p = \{(x, y)|x \in \mathcal{X} \wedge y = 1\}$ and the set of negative training examples as $S_n = \{(x, y)|x \in \mathcal{X} \wedge y = -1\}$. In multiple instance learning [22] not every label y_i for every x_i is known. The positive label is only known for sets of instances which are called bags. For every bag X_i with label $+1$ it is only known that at least one instance of X_i is associated with label $+1$. Every instance in a negative bag is associated with label -1. More formally this means that the set of positive bags is $X_p = \{(X_i, 1)|\exists x_j \in X_i : (x_j, y_j) \in S_p\}$. The set of negative bags is $X_n = \{(X_i, -1)|\forall x_j \in X_i : (x_j, y_j) \in S_n\}$. The

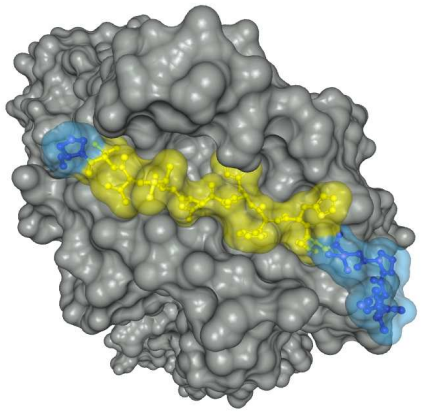

Figure 4.1: ***Binding core of a peptide:*** *This figure shows the structure of an MHCII molecule (grey) together with a bound peptide (blue and yellow). The binding core of the peptide is shown in yellow visualized with BALLView [77]. The PDB ID of the structure is 1BX2.*

multiple instance learning problem is to find the best predictor for predicting the labels of bags.
Kernels for multiple instance learning were introduced by Gärtner *et al.* [33] in 2002. The normalized set kernel (NSK) by Gärtner *et al.* [33] is the following:

$$k(X, X') := \frac{\sum_{x \in X, x' \in X'} k_{\mathcal{X}}(x, x')}{f_{\text{norm}}(X) f_{\text{norm}}(X')} \tag{4.1}$$

with $k_{\mathcal{X}}$ being a kernel on $\mathcal{X}$. Gärtner *et al.* [33] evaluated different normalization functions f_{norm} and showed that averaging ($f_{\text{norm}}(X) = \#X$) and feature space normalization ($f_{\text{norm}}(X) = \sqrt{\sum_{x \in X, x' \in X} k_{\mathcal{X}}(x, x')}$) perform equally well on the datasets studied. Preliminary results on our data also suggest that both methods perform equally well (data not shown). Therefore, in the following only normalization by feature space normalization is considered.
Gärtner *et al.* [33] hypothesized in their paper, that the kernel could also be used for multiple instance regression [25, 98]. In this setting every bag X_i has a label $y_i \in \mathbb{R}$.

4.2.2 Multiple Instance Learning for MHCII Prediction

Since for most of known MHCII binders the binding core is unknown, one cannot directly use the binding core for training a learning machine. Fig. 4.1 shows a structure of an MHCII molecule for which the binding core is known. Unfortunately there are very few such structures available.
Previous work on MHCII prediction [37, 124] suggests that only aliphatic (Ile, Leu, Met, Val) and aromatic (Phe, Trp, Tyr) amino acids in position one

are common. Thus, we represent every putative binder by a bag containing all 9-mers (putative binding cores) with aromatic or aliphatic amino acid at position one. By this, we transformed the data directly into a multiple instance learning problem in which every positive bag has at least one positive binding core. All negative bags just contain false binding cores. Formally, this means that every putative binder s_i of length m is represented by a bag (X_i, y_i).

$$X_i = \{x | x = s_{i_{k,k+1,\ldots,k+8}} \wedge k \geq 1 \wedge k + 8 \leq m \wedge x_1 \in \{\text{Ile}, \text{Leu}, \text{Met}, \text{Val}, \text{Phe}, \text{Trp}, \text{Tyr}\}\}$$

are all putative binding cores and y_i is the binding affinity measured for the putative binder s_i.
In this thesis we introduce two predictors for MHCII binding peptide prediction. The first predictor is just trained on parts of the data of the allele for which the predictions should be made. This predictor is called MHCIISingle in the following. It will be shown that the performance of MHCIISingle is comparable to the best methods in the field. This predictor is particularly useful for alleles, for which sufficient binding data is available.
The second predictor does not need to be trained on data of the allele for which the predictions should be made. Instead, data from other alleles can combined in a way which reflects the similarity of the binding pockets of the target allele to the binding pockets of the other alleles. This predictor will be called MHCIIMulti in the following. Because no data of the allele, for which the predictions should be made is needed, one can even build predictors for alleles with little or no experimentally determined binders.
In this thesis, we use the normalized set kernel with an RBF kernel for MHCIISingle. Furthermore, we introduce a new kernel based on the normalized set kernel for MHCIIMulti.

4.2.3 Feature Encoding

Venkatarajan and Braun [135] evaluated in 2001 different physicochemical properties of amino acids. They performed a dimension reduction by principal component analysis (PCA) on a large set of features from the AAindex database [50] and showed that every amino acid can be represented adequately by a five-dimensional feature vector. This encoding was already used in a recent study on MHC binding by Hertz *et al.* [42] and will be called *PCA encoding* in the following.

4.2.4 Predictions for Alleles with Sufficient Data

For alleles, for which enough experimental data is available, we build predictors which are just trained on binding peptide data for the particular allele. In this setting we use the *normalized set kernel* [33] with $k_{\mathcal{X}}$ being the RBF kernel. $\mathcal{X}$ is the set of all putative binding cores. This means that $\mathcal{X}$ is the set of every possible nine amino acid long peptide sequence in PCA encoding for which the first amino acid is aliphatic or aromatic. This means that

every input vector has length 45. The predictor is trained using this kernel function together with ν-SVR [110].

4.2.5 Combining Allele Information with Peptide Information

Representation of MHCII Alleles

Sturniolo *et al.* [124] showed in 1999 that there is a correspondence between the structures of the binding pockets of the MHCII and the polymorphic residues in this region. They defined certain positions inside the amino acid sequence of the allele sequences and showed that alleles having the same residues at these positions also have similar binding pocket structures. This was done for several alleles and binding pockets for peptide positions 1, 4, 6, 7 and 9 because these positions are assumed to have the largest influence on binding [124].
To represent each allele, we encode every polymorphic residue of the pockets 1, 4, 6, 7, and 9 by PCA encoding and calculate a mean of the encoded vectors for every pocket position. This results in a 25×1 dimensional vector $p = \left(p_1^T, p_4^T, p_6^T, p_7^T, p_9^T\right)^T$ for every allele, which is called *pocket profile vector* in the following. To get the polymorphic residues for alleles that were not defined by Sturniolo *et al.* [124], we used the HLA-DRB1, HLA-DRB3, HLA-DRB4 and HLA-DRB5 alignments of the IMGT/HLA database [102] (release 2.18.0, 09-July-2007).
We computed the sequence logo [19] for an alignment of all HLA-DRB1, HLA-DRB3, HLA-DRB4 and HLA-DRB5 alleles. It is shown in Fig. 4.2. Since the alignments show very good conservation for alleles HLA-DRB1, HLA-DRB3, HLA-DRB4 and HLA-DRB5 at the non-pocket positions, we assume that this procedure is applicable at least for these HLA-DRB alleles which constitute 525 of all 765 unique MHCII alleles (on the protein sequence level), currently contained in the IMGT/HLA database [102].

Similarity Function of MHCII Binding Pockets

Our goal was to get a similarity measure between pocket positions of alleles. Since we have the pocket profile vectors, a natural idea is to take the Pearson correlation between the corresponding positions of the pocket. To get a similarity measure we added one which means that the similarities are in the interval $[0, 2]$. The resulting similarity measure for each pocket $i = 1, 4, 6, 7, 9$ is then

$$\mathrm{sim}_i(p, p') := \mathrm{Pearson}(p_i, p'_i) + 1. \tag{4.2}$$

This function was used in our work to measure similarity between the binding pockets corresponding to peptide position 1, 4, 6, 7 and 9.

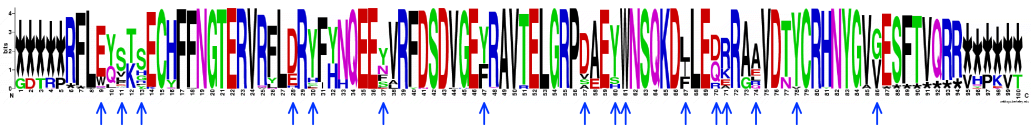

Figure 4.2: ***Conservation of MHCII allele protein sequences:*** *The conservation of the alignments of all HLA-DRB1, HLA-DRB3, HLA-DRB4 and HLA-DRB5 alleles from the IMGT/HLA database [102] is shown in this sequence logo [19]. The arrows pointing at positions of the sequence logo are polymorphic residues, which are used for the binding pocket profile vectors.*

Combining Allele Pocket Profiles with Peptide Information

For MHCI binding peptide prediction there have been two approaches for learning binding prediction models for alleles which do not need experimental data for the target allele [44, 79]. Both methods measure the similarity of the alleles for the whole allele.
Looking at the structural level, one amino acid that does not fit into a binding pocket can change the whole binding affinity of the peptide. Therefore, we want to enforce similarity of peptides at a certain position of the binding core if their binding pockets for the respective position are very similar. Thus, we use similarities between the binding pockets directly in our kernel function to be able to account for these cases, too.
We now define the kernel $k_{\mathrm{pw-RBF}}$, which is defined on $\mathcal{A} \times \mathcal{X}$. $\mathcal{A}$ is the set of all possible pocket profile vectors and $\mathcal{X}$ is again the set of all possible nine amino acid long peptide sequences in PCA encoding. Let $p = \left(p_1^T, p_4^T, p_6^T, p_7^T, p_9^T\right)^T$ be the *pocket profile vector* of peptide sequence s. Let $x = (x_1^T, x_2^T, ..., x_9^T)^T$ be a putative binding core of sequence s, for which every x_i is the PCA encoding of the amino acid at position i in the putative binding core. Let p' and x' be defined analogously for peptide sequence s'. In MHCIISingle the inner kernel function of the normalized set kernel is a standard RBF kernel:

$$k_{\mathrm{RBF}}(x, x') = \exp^{-\frac{\|x-x'\|^2}{2\sigma^2}} . \tag{4.3}$$

As mentioned above, the kernel function should be able to weight positions according to the similarity of the alleles. Therefore, we use a positionally-weighted RBF-kernel:

$$k_{\mathrm{pw-RBF}}((p, x), (p', x')) = \exp^{-\frac{w_1 \times \|x_1 - x_1'\|^2 + w_2 \times \|x_2 - x_2'\|^2 + ... + w_9 \times \|x_9 - x_9'\|^2}{2\sigma^2}} . \tag{4.4}$$

In our setting the weights are determined using the sim function, which was mentioned above:

$$w_i := \mathrm{sim}_i(p, p') \ \forall i = 1, 4, 6, 7, 9 \tag{4.5}$$

Since the other positions are not as important for binding, we set the weights w_2, w_3, w_5 and w_8 (which correspond to peptide positions 2, 3, 5 and 8) to 0.5.
In this work $k_{\mathrm{pw-RBF}}$ is used as the inner kernel function of the *normalized set kernel* [33] in conjunction with ν-SVR [110] for MHCIIMulti.

Positive Semi-Definiteness of NSK with Positionally-Weighted RBF Kernel

Gärtner *et al.* [33] showed that the normalized set kernel is positive semi-definite if and only if the inner kernel function is positive semi-definite. This directly means, that the NSK in conjunction with the standard RBF kernel is positive semi-definite. For the combination of NSK with the positionally-weighted RBF kernel ($k_{\text{pw-RBF}}$), we just have to show that $k_{\text{pw-RBF}}$ is positive semi-definite. Our proof is very similar to Li and Jiang [67] who used the Schoenberg Theorem [108]:

Theorem 4.1 (SCHOENBERG THEOREM)**.** *Let $\mathcal{X}$ be a space in which a distance function $d(x, y)$ is defined subject to the following conditions:*

1. $d(x, y) = d(y, x) \geq 0$
2. $d(x, x) = 0$

for all $x, y \in \mathcal{X}$. The function $\exp^{(-d^p(x,y))}$ is positive definite if $0 < p \leq 2$ and not positive definite if $p > 2$.

Theorem 4.2 (Positive Semi-Definiteness of $k_{\text{pw-RBF}}$)**.** *The positionally-weighted RBF kernel with*

$$k_{\text{pw-RBF}}((p, x), (p', x')) = \exp^{-\frac{w_1 \times \|x_1 - x_1'\|^2 + w_2 \times \|x_2 - x_2'\|^2 + \ldots + w_9 \times \|x_9 - x_9'\|^2}{2\sigma^2}}$$

as defined in 4.2.5 is positive semi-definite.

Proof: Since $p = 1$, it is sufficient to show that requirements 1. and 2. of the Schoenberg theorem hold for $d(x, x') = \frac{w_1 \times \|x_1 - x_1'\|^2 + w_2 \times \|x_2 - x_2'\|^2 + \ldots + w_9 \times \|x_9 - x_9'\|^2}{2\sigma^2}$:

1.$(d(x, x') = d(x', x) \geq 0)$:

$$\begin{aligned} d(x, x') = &\ \tfrac{1}{2\sigma^2} \left(w_1 \times \|x_1 - x_1'\|^2 + \ldots + w_9 \times \|x_9 - x_9'\|^2\right) \\ = &\ \tfrac{1}{2\sigma^2} \left(w_1 \times \|x_1' - x_1\|^2 + \ldots + w_9 \times \|x_9' - x_9\|^2\right) \quad = d(x', x) \end{aligned} \tag{4.6}$$

Furthermore, $d(x, x') \geq 0 \ \forall\, x, x' \in \mathcal{X}$ since all summands are positive (the w_i are between zero and two). This means that $d(x, x') = d(x', x) \geq 0 \ \forall\, x, x' \in \mathcal{X}$.

2.$(d(x, x) = 0)$:

This immeadiately follows from the definition:

$d(x, x) = \frac{1}{2\sigma^2} \left(w_1 \times \|x_1 - x_1\|^2 + \ldots + w_9 \times \|x_9 - x_9\|^2\right) = 0.$ ■

Training Choices for MHCIIMulti

We design a procedure to get the largest possible training set, in which the similarities of the target allele to the other alleles are reflected in the number of training samples from the particular alleles. The idea is that training samples from more similar alleles should enable better predictions for the target allele than distant ones. To compute similarities between alleles, we calculate the Pearson correlation between alleles using the pocket profile vectors. Let p_i and p_j be the pocket profile vectors of alleles i and j. The similarity between these vectors is the Pearson correlation of p_i and p_j scaled linearly to $[0, 1]$. This value is called $allelesim_{i,j}$ in the following. Let n_i be the number of sequences of allele i. For a particular target allele j, the procedure is the following:

```
For every allele i ≠ j
    Compute the maximal number t_i such that allelesim_{i,j} × t_i ≤ n_i.
Choose the minimum of all t_i, which is now called t*.
For every allele i ≠ j
    Choose t* × allelesim_{i,j} peptide sequences randomly from allele i
    and assign them to the training set.
```

Since for the benchmark dataset the binding affinities are not distributed uniformly, we partition the data into three parts ($\left[0, \frac{1}{3}\right]$, $\left]\frac{1}{3}, \frac{2}{3}\right]$ and $\left]\frac{2}{3}, \max\right]$). We then randomly choose from these partitions such that we have the same number of samples from each partition.

Nearest Neighbor Predictor MHCIISingle$^{\mathrm{NN}}$

To show that the $k_{\mathrm{pw-RBF}}$ kernel function of MHCIIMulti really improves the MHCII binding prediction if data from various alleles is combined we introduce MHCIISingle$^{\mathrm{NN}}$. This predictor is the MHCIISingle predictor of the nearest neighbor allele. The nearest neighbor allele j of allele i is the allele for which $allelesim_{i,j}$ is maximal $\forall j \neq i$.

Aggregating Predictor MHCIIMulti*

We choose the number of training samples per allele according to the similarities of the alleles as described above. Therefore, we do not use all peptides that are available. Since we do not want to miss important peptides we build aggregating predictors over ten random draws of the training sets. The idea is similar to bagging [7]. The only difference is that we do not need bootstrapping, since we have enough data for the training alleles. The aggregating versions of the predictor MHCIIMulti is called MHCIIMulti*. The whole workflow can be seen in a UML activity diagram in Fig. 4.3.

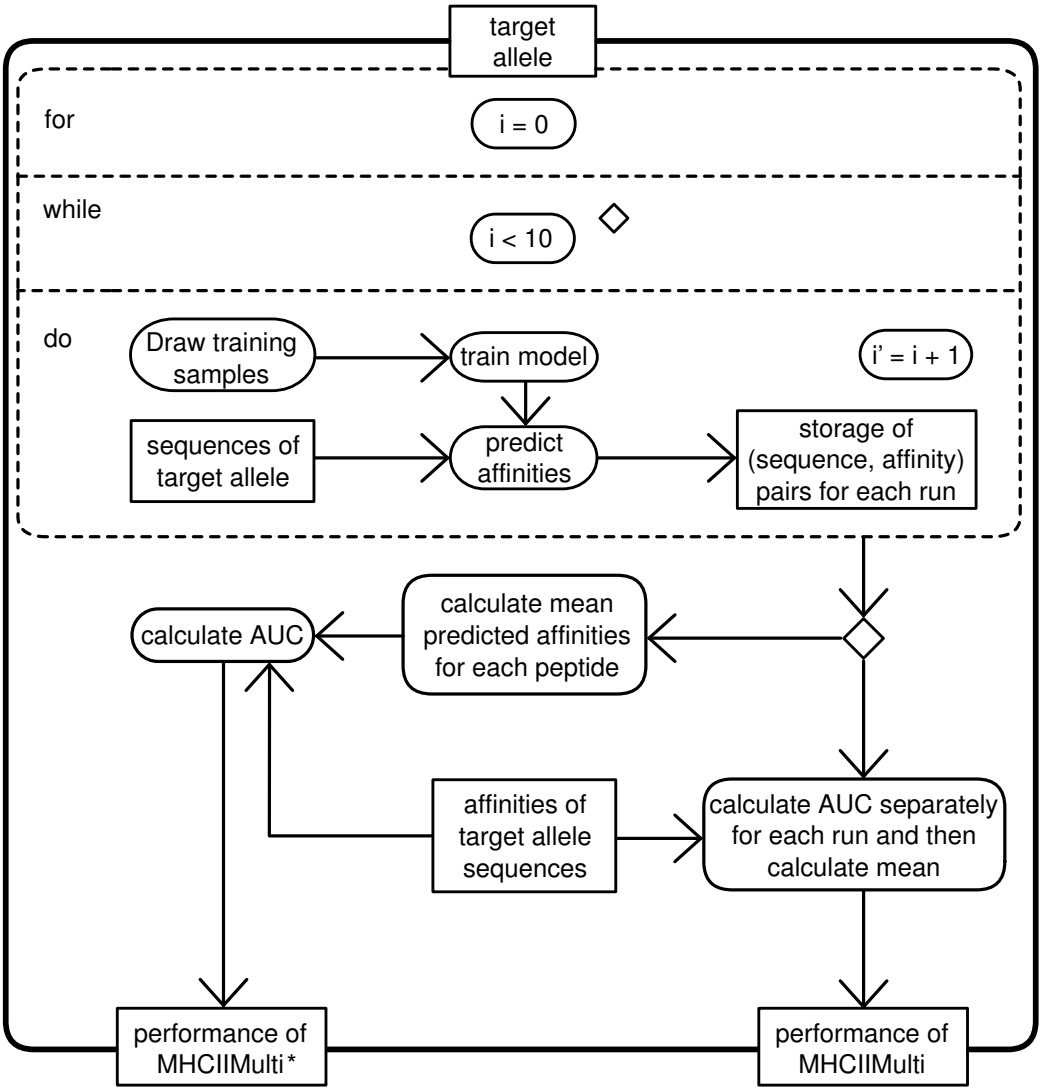

Figure 4.3: ***UML activity diagram of performance evaluation for LOAO predictors:*** *This UML activity diagram shows the workflow of the performance evaluation of the leave one allele out predictors MHCIIMulti and MHCIIMulti*.*

4.2.6 Data

We show the performance of our predictors on an MHCII benchmark dataset, introduced by Wang *et al.* [138]. This dataset contains peptide binding data, measured in the laboratory of Wang *et al.* [138], for 14 human alleles as well as three mouse alleles. Binding affinities of the benchmark dataset were given as IC_{50} values, which is defined as the concentration at which 50% of the MHCII molecules are bound. The smaller the IC_{50} value, the better is the binder. There are many peptides with very high IC_{50} values. Since the cutoff for binders is between 500 and 1,000 nM there is not a big difference between a non-binder with IC_{50} value of 10,000 or 20,000 nM. Therefore, we transformed the IC_{50} values like Nielsen *et al.* [80] to the interval $[0, 1]$. Let a_i be the binding affinity of peptide i. The log-transformed binding affinity a'_i is defined as $a'_i := 1 - \log_{50000}(a_i)$. Like Nielsen *et al.* [80], we set the $a'_i < 0$ to zero, which is needed for all peptides with IC_{50} value larger than 50,000 nM. In the following, the dataset will be called $D_{\text{benchmark}}$.
Peptide sequences for which no binding core could be found (aliphatic or aromatic amino acid at position one) were excluded from all evaluations. This was the case for less than 3% of peptides (270 out of all 9478). Out of these peptides only 64 peptides are considered binders (IC_{50} value smaller than 1,000 nM [138] which is equal to a log-transformed value greater than 0.3616). Since the whole dataset contains 6,475 binders in total, this means that our assumption that every binder has to have a binding core with an aliphatic or aromatic amino acid at position one just misses 64 out of 6,475 binders which is under 1%.

4.3 Results

In this section we compare our predictors to other state-of-the-art methods. In particular we compare our performance to the results of Wang *et al.* [138] who performed a large scale evaluation on MHC class II prediction methods. We show on their benchmark dataset that our predictor MHCIISingle, which is trained on parts of the target allele dataset, performs equally well or better than all other methods. Furthermore, we show that MHCIIMulti* can predict binding for alleles without using any training data of the target allele and achieves performances that are comparable to the best predictors trained on binding data of the target allele.

4.3.1 Performance on Single Allele Datasets

Wang *et al.* [138] recently compared the performances of state-of-the-art predictors for MHCII binding. We show a comparison to the top four methods of their evaluation. All performances are measured in area under the ROC curve. Wang *et al.* [138] measured the performance of the ARB method [9] by 10-fold cross-validation. The performance of the other methods was evaluated using available webservers. The authors justified

this procedure by the fact that they measured the performance on unpublished data, which had been measured in their labs. Therefore, it is unlikely that any of these methods (except the ARB method) was trained on parts of this dataset. To compare the performance of MHCIISingle to this evaluation, we performed a 10-fold cross-validation using parameter ranges $C \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$, $\nu \in \{0.2 * 1.4^i | i = 1, 2, ..., 5\}$ and $\sigma \in \{0.0625, 0.125, 0.25, ..., 16\}$. Table 4.1 shows that MHCIISingle outperforms all other single methods. The column "Consensus" corresponds to the consensus approach of Wang *et al.* [138] in which the best three predictors for the particular allele are combined to achieve higher accuracy. One could assume that with MHCIISingle as one of these three predictors the accuracy will improve, since the performance of MHCIISingle is comparable to the consensus approach.
A further improvement can be achieved by incorporating binding data from other alleles. Therefore, we performed a CV with MHCIIMulti and the same parameters as above. Additionally, we chose data from other alleles of $D_{\text{benchmark}}$. To minimize random effects, we performed this procedure ten times and listed the mean performance in Table 4.1. It can be seen that the incorporation of extra data improves the performances on many alleles (8 out of 14). Especially for the two alleles, for which MHCIISingle performed worst, a significant improvement can be achieved by MHCIIMulti. The worse performance on allele HLA-DRB1*0101 can be explained by the number of samples in $D_{\text{benchmark}}$ since there are 3,882 peptides. For alleles for which there exist such a big amount of training data the incorporation of binding data from different alleles does not improve the predictions which is in accordance with what one would expect. The bad performance on allele HLA-DRB3*0101 can be explained by the fact that the eleventh residue of the beta chain ($\beta 11$) of the MHCII molecule of this allele has an arginine, which reaches into pocket number four (and therefore influences binding), although it is located in pocket number six [85]. Since the arginine at $\beta 11$ is exclusive to the DR52a [85] alleles with the sole exception of DRB1*1446, it can be assumed that this effect is limited to this small number of alleles (37).

4.3.2 Performance of Leave-Allele-Out Predictors

To show that MHCIIMulti performs well, although it is not trained on any data of the target allele, we conducted the following experiment. The training samples were chosen as described in the Section 4.2. We then performed a validation on the training set to determine the best hyperparameters ($C \in \{0.01, 0.1, 1, 10\}$, $\nu \in \{0.2 * 1.4^i | i = 1, 2, ..., 5\}$ and $\sigma \in \{0.0625, 0.125, 0.25, ..., 4\}$). The binding data of the alleles were stored in separate partitions. The best hyperparameters were found by training on all but one of these partitions and measuring the performance on the left-out partition. With the best hyperparameters of the validation we trained our predictors with the whole training set. We then measured the area under the ROC curve performance on the target allele. The whole process was repeated ten times to minimize random effects. The mean area under the ROC

MHCII type	# peptides	ARB	PROPRED	SMM-align	Consensus	MHCIISingle	MHCIIMulti
DRB1*0101	3882	0.76	0.74	0.77	0.79	**0.81**	0.75
DRB1*0301	502	0.66	0.65	0.69	0.72	**0.73**	0.72
DRB1*0401	512	0.67	0.69	0.68	0.69	0.67	**0.78**
DRB1*0404	449	0.72	0.79	0.75	**0.80**	0.79	**0.80**
DRB1*0405	457	0.67	0.75	0.69	0.72	**0.83**	0.79
DRB1*0701	505	0.69	0.78	0.78	0.83	0.82	**0.90**
DRB1*0802	245	0.74	0.77	0.75	**0.82**	0.76	0.79
DRB1*0901	412	0.62	-	0.66	**0.68**	0.64	0.66
DRB1*1101	520	0.73	0.80	0.81	0.80	0.85	**0.87**
DRB1*1302	289	**0.79**	0.58	0.69	0.73	0.74	0.73
DRB1*1501	520	0.70	0.72	0.74	0.72	0.72	**0.75**
DRB3*0101	420	0.59	-	0.68	-	**0.72**	0.57
DRB4*0101	245	0.74	-	0.71	0.74	**0.79**	0.78
DRB5*0101	520	0.70	0.79	0.75	0.79	0.81	**0.90**
Mean		0.71	0.73	0.73	0.76	0.76	**0.77**

Table 4.1: ***Performance comparison on benchmark dataset:*** *The performance of our predictors is compared to the best four methods presented in [138]. The performance of MHCIISingle, MHCIIMulti and ARB are measured by 10-fold cross validation. All other methods are trained on binding data of the target allele which was not contained in the benchmark dataset. MHCIIMulti uses additional training data from the other alleles of the benchmark dataset.*

MHCII type	MHCIISingle^{NN}	MHCIIMulti	MHCIIMulti*
DRB1*0101	**0.74**	0.64	0.69
DRB1*0301	0.61	0.69	**0.70**
DRB1*0401	0.64	0.76	**0.78**
DRB1*0404	0.70	0.79	**0.82**
DRB1*0405	**0.78**	0.76	0.77
DRB1*0701	0.72	0.89	**0.91**
DRB1*0802	0.71	0.77	**0.79**
DRB1*0901	0.57	0.64	**0.65**
DRB1*1101	0.79	0.87	**0.90**
DRB1*1302	0.62	0.68	**0.69**
DRB1*1501	0.66	0.75	**0.77**
DRB3*0101	0.51	**0.54**	**0.54**
DRB4*0101	**0.77**	0.69	0.72
DRB5*0101	0.73	0.89	**0.92**
Mean	0.68	0.74	**0.76**

Table 4.2: ***Leave-allele-out prediction on benchmark dataset:*** *The performance of our predictors MHCIIMulti and* MHCIIMulti* *are shown which are not trained on any data of the target allele. Instead, the predictors are trained on data from the other alleles of* $D_{\text{benchmark}}$*. Additionally, the performance of the nearest neighbor predictor MHCIISingle*^{NN} *is shown, which is trained on data of the most similar allele to the target allele.*

curve over the ten runs is given in Table 4.2. For the aggregating predictor MHCIIMulti*, which was introduced in the Section 4.2, we calculated the mean prediction labels for every test sample over the ten runs. Afterwards, we measured the area under the ROC curve for these labels.
It can be seen in Table 4.2 that the predictors MHCIIMulti and especially MHCIIMulti* perform quite well on $D_{\text{benchmark}}$ although they were not trained on any binding data of the target allele. One can hypothesize that this performance could also be reached for other alleles, for which no binding data is available, since we did not use any data of the target allele. These predictors perform even better than MHCIISingle on some alleles which shows that the method is not just valuable for new alleles but also for predictions for alleles for which there exists binding data. The performance of the nearest neigbor predictor MHCIISingle^{NN} is worse than the performance of MHCIIMulti. The fact that MHCIIMulti and MHCIIMulti* outperform the nearest neighbor predictor underlines that our new kernel function, which takes the similarities of the alleles into account, is very valuable for this kind of predictions.

4.3.3 Implementation

All methods were implemented in C++. We used LIBSVM [14] for support vector learning. The predictions for all alleles are integrated into Epi-

ToolKit [30], available at `http://www.epitoolkit.org/mhciimulti`.

4.4 Discussion

The proposed method is a novel approach for predicting MHC class II binding peptides for alleles lacking experimental data and thus opens up new alleys for the design of peptide-based therapeutic or prophylactic vaccines. Obviously, a conclusive validation of predicitons for alleles without experimental data is difficult. The leave-one-allele out predictions presented here indicate, however, that the method performs very well. One could object that restricting the first amino acid of the binding core to aromatic and aliphatic amino acids is a strong assumption. Nevertheless, if one selects all putative binding peptides of the 9,478 peptide sequences in $D_{\text{benchmark}}$ for which no binding core with an aromatic or aliphatic residue at position one can be found, a predictor which just predicts 0 (non-binder) would have 0.7630 classification rate on these peptides. In other words, only for 270 peptides or 2.85% out of the 9,478 peptides no binding core with aromatic or aliphatic residue at position one can be found and only 64 out of these are considered as binders (log-transformed binding affinity greater than 0.3616). This is why we think that the heuristic is very well applicable and reflects a general property for MHCII binding. This is also supported by previous work [37, 124]. Moreover, the restriction to these binding cores is one of the key parts of this work because if one selected all 9-mers as binding cores this would add a lot of noise to the bags and the positional weighting of the binding core would not have a big effect since nearly every residue of a peptide (except the residues at the ends) would be at every position in one of the instances in the bag. Ultimately, only experimental testing or structure-based studies will reveal whether some of the rarer alleles might deviate from this behavior on the first position.

To improve peptide-MHCII affinity prediction it would be interesting to use other multiple instance learning approaches. Kwok *et al.* [64] presented marginalized multi-instance kernels in 2007 at the International Joint Conference on Artificial Intelligence. In this approach, the kernel $k(X_i, X_j)$ does not weight every $k_{\mathcal{X}}(x_i, x_j)$ equally, as the NSK does. Instead, it weights the inner kernel function by the similarity of the (estimated) labels of the single instances. In their paper, Kwok *et al.* state that their marginalized multi-instance kernels could be combined with the regularization framework they presented at ICML 2006 [15]. A combination of this framework and the marginalized multi-instance kernels should lead to improved results for peptide-MHCII binding affinity prediction.

Building predictors for alleles for which no experimental data exists belongs to the field of transfer learning. Bickel *et al.* [4] very recently presented a sophisticated approach for transfer learning. It would be very interesting to combine this approach with the marginalized multi-instance kernels.

Chapter 5

Conclusions and Discussion

"My mind seems to have become a kind of machine for grinding laws out of large collections of facts,..."
- Charles Darwin, The Autobiography of Charles Darwin, 1881

In biology, there are always interesting traits to discover. Unfortunately, there does not exist for each object of interest an expert machine like Charles Darwin. Therefore, we are interested in designing learning machines which are able to learn general rules from the data of biological entities. One of the main questions to ask before building a learning machine is: "which properties of the data should be learnt by the learning algorithm?" In this step it is usually beneficial if expert knowledge of the particular domain is available. If, for example, our task was to predict whether a person is able to speak, it would be very reasonable to account for the age of the person inside the learning algorithm. Unfortunately, it is often not that easy to find meaningful parts of the data – which help in predicting the property of interest, and expert knowledge is often not available. Thus, in many applications of machine learning, scientists put all the "appropriate" features into the learning algorithm and perform feature selection during the learning process.
In this thesis, we focus on kernel-based approaches for machine learning. Applying a certain kernel to an application area also requires some prior knowledge. If, for example, the position of particular k-mers is very important and one chooses the spectrum kernel [66], which is not position-aware, it will be hard to come up with any reasonable results. In most cases, the assumptions one makes by choosing the kernel are milder than by choosing features. We introduce the paired oligo-border kernel in this thesis, which assumes that all interesting properties of the peptides are represented by the amino acids. We do not directly put into the assumption that aromatic amino acids at certain positions are indicative for proteotypic peptide prediction in PAGE-MALDI experiments, but nevertheless, the SVM with the *POBK* is able to find and exploit this feature. Thus the approach is more general than deciding on a specific set of features beforehand. This leads to a wider applicability of our kernel function.
We show in Sections 3.1 and 3.2 that the *POBK* can be used to predict chromatographic separation very accurately. For SAX-SPE behavior prediction, our method performs better than all other methods. For retention time

prediction in reversed-phase chromatography, our method performs better than all but one of the available methods. The only method with better performance requires about 345,000 training peptides which are not easily measured before being able to use the predictor. Our method just needs a fraction of this amount of training data (40 - 200 peptides) although it achieves nearly the same performance. We show that a good predictor for chromatographic behavior is very valuable for peptide identification by applying the predicted retention time in a p-value-based filter. This filter allows us to lower mass spectrometric scoring thresholds, filter out false identifications, and get more correctly identified spectra while keeping the same precision. We furthermore show that our method is applicable under different chromatography conditions in Section 3.2 in which we predict retention times for chromatographic separations at different pH values. Since we are able to build accurate predictors for both dimensions, we can build filters for both retention time dimensions. As both separations are nearly orthogonal [20], it is even more unlikely that a false peptide identification is not filtered out by one of the two filters. We can show that the combined filters yield the largest increase in the number of correctly identified spectra at comparable precision.

The good performance of the *POBK* and the *OBK* at predicting proteotypic peptides, shown in Section 3.3, is further evidence that it is generally applicable to computational proteomics problems. We show that our kernels perform better than other methods using the features of Mallick *et al.* [73] or Lu *et al.* [70] on a comparative benchmark. Furthermore, the visualization of the resulting classifier allows gaining interesting insights into the biochemical processes which are involved during the whole measurement process.

For predicting peptide-MHCII affinity, we put in another mild assumption to be able to use kernel-based learning machines. This assumption was that there has to exist a reasonable binding core in every peptide for which we want to perform the prediction. This mild assumption is enough to transform the prediction problem into a multiple instance learning problem for which kernel approaches exist [33]. For our positionally-weighted RBF kernel, we put in the further assumption that MHCII molecules with similar pockets should also bind similar amino acids at the particular positions. This enables us to build predictors for about two thirds of all known MHCII alleles, instead of just about 6% for which there existed peptide-MHCII binding affinity predictors previously.

It would be interesting to use multiple kernel learning [117] together with the *POBK* and *OBK* for different sigmas and k-mer lengths using the 2-norm optimization [57]. We performed experiments with the 1-norm optimization but unfortunately did not get better results. This could be explained by the fact that 1-norm multiple kernel learning tends towards sparse kernel combinations and therefore does not lead to better performances in many applications. The 2-norm optimization problem of Kloft *et al.* [57] was presented only very recently. It would be very interesting to see whether kernel combinations using the 2-norm optimization lead to increased performances in retention time prediction. One could even try to add other kernels which

contribute features which cannot be directly learnt from the sequence. This should enable even higher performances.
For peptide-MHCII affinity prediction, it would be interesting to use other multiple instance learning approaches. Kwok *et al.* [64] presented a kernel which does not weight every instance in the bag equally as the NSK does. We also performed experiments with the method of Bunescu *et al.* [10] but achieved performances comparable to the NSK for multiple instance classification (data not presented in this thesis). We could not use the approach of Bunescu *et al.* for the binding affinity prediction problem since it cannot be applied in multiple instance regression. Kwok *et al.* [64] state in their paper that their marginalized multi-instance kernels could be combined with the regularization framework they presented at ICML 2006 [15]. It would, therefore, be interesting to see whether this combination of methods leads to better results.
Building predictors for alleles for which no experimental data exists belongs to the field of transfer learning. Bickel *et al.* [4] very recently presented a sophisticated approach for transfer learning. It would be very interesting to combine these methods with the marginalized multi-instance kernels.
Beyond the problems tackled in this thesis, there are many open problems in computational biology, for which kenel-based machine learning should deliver accurate results. The problem of spectrum intensity prediction [56], for example, could also be approached by structured output prediction [45] using appropriate kernels.

Bibliography

[1] Ruedi Aebersold and Matthias Mann. Mass spectrometry-based proteomics. *Nature*, 422(6928):198–207, Mar 2003.

[2] A. J. Alpert and P. C. Andrews. Cation-exchange chromatography of peptides on poly(2-sulfoethyl aspartamide)-silica. *J Chromatogr*, 443:85–96, Jun 1988.

[3] Leigh Anderson and Christie L Hunter. Quantitative mass spectrometric multiple reaction monitoring assays for major plasma proteins. *Mol Cell Proteomics*, 5(4):573–588, Apr 2006.

[4] Steffen Bickel, Christoph Sawade, and Tobias Scheffer. Transfer learning by distribution matching for targeted advertising. In *NIPS '08*, pages 105–112, 2008.

[5] K. Bieman. Mass spectrometry. *Ann Rev Biochem*, 32:755–780, 1963.

[6] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. *Introduction to Statistical Learning Theory*, volume 3176 of *Lecture Notes in Artificial Intelligence*, pages 169–207. Springer, 2004.

[7] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[8] V. Brusic, G. Rudy, G. Honeyman, J. Hammer, and L. Harrison. Prediction of MHC class II-binding peptides using an evolutionary algorithm and artificial neural network. *Bioinformatics*, 14(2):121–130, 1998.

[9] Huynh-Hoa Bui, John Sidney, Bjoern Peters, Muthuraman Sathiamurthy, Sinichi Asabe, and et al. Automated generation and evaluation of specific MHC binding predictive tools: ARB matrix applications. *Immunogenetics*, 57(5):304–314, 2005.

[10] Razvan C. Bunescu and Raymond J. Mooney. Multiple instance learning for sparse positive bags. In *Proceedings of the 24th international conference on Machine learning*, pages 105–112, Corvalis, Oregon, 2007. ACM.

[11] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Min Knowl Discov*, 2(2):121–167, 1998.

[12] William R. Cannon, Danny Taasevigen, Douglas J. Baxter, and Julia Laskin. Evaluation of the influence of amino acid composition on the propensity for collision-induced dissociation of model peptides using molecular dynamics simulations. *Journal of the American Society for Mass Spectrometry*, 18(9):1625–1637, September 2007.

[13] Athanassia Chalimourda, Bernhard Schölkopf, and Alex J. Smola. Experimentally optimal ν in support vector regression for different noise models and parameter settings. *Neural Networks*, 18(2):205–205, March 2005.

[14] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[15] Pak-Ming Cheung and James T. Kwok. A regularization framework for multiple-instance learning. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 193–200, New York, NY, USA, 2006. ACM.

[16] Roman M. Chicz, Robert G. Urban, William S. Lane, Joan C. Gorga, Lawrence J. Stern, Dario A. A. Vignali, and Jack L. Strominger. Predominant naturally processed peptides bound to HLA-DR1 are derived from MHC-related molecules and are heterogeneous in size. *Nature*, 358(6389):764–768, August 1992.

[17] Corinna Cortes and Vladimir Vapnik. Support vector networks. In *Machine Learning*, pages 273–297, 1995.

[18] Robertson Craig and Ronald C Beavis. Tandem: matching proteins with tandem mass spectra. *Bioinformatics*, 20(9):1466–1467, Jun 2004.

[19] Gavin E Crooks, Gary Hon, John-Marc Chandonia, and Steven E Brenner. WebLogo: a sequence logo generator. *Genome Res*, 14(6):1188–1190, 2004.

[20] N. Delmotte, M. Lasaosa, A. Tholey, E. Heinzle, and C.G. Huber. Two-dimensional reversed-phase x ion-pair reversed-phase hplc: An alternative approach to high-resolution peptide separation for shotgun proteome analysis. *J Proteome Res*, 6(11):4363–4373, 2007.

[21] David DeLuca, Barbara Khattab, and Rainer Blasczyk. A modular concept of hla for comprehensive peptide binding prediction. *Immunogenetics*, 59(1):25–35, 2007.

[22] Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif Intell*, 89(1-2):31–71, 1997.

[23] Bruno Domon and Ruedi Aebersold. Challenges and opportunities in proteomics data analysis. *Mol Cell Proteomics*, 5(10):1921–1926, Oct 2006.

[24] Pierre Dönnes and Oliver Kohlbacher. SVMHC: a server for prediction of MHC-binding peptides. *Nucleic Acids Res*, 34 (Web Server issue):W194–W197, 2006.

[25] Daniel R. Dooly, Qi Zhang, Sally A. Goldman, and Robert A. Amar. Multiple-instance learning of real-valued data. *J Machine Learn Res*, 3:651–678, 2002.

[26] Jacek P Dworzanski, A. Peter Snyder, Rui Chen, Haiyan Zhang, David Wishart, and Liang Li. Identification of bacteria using tandem mass spectrometry combined with a proteome database and statistical scoring. *Anal Chem*, 76(8):2355–2366, Apr 2004.

[27] Pehr Edman. Method for Determination of the Amino Acid Sequence in Peptides. *Acta Chem. Scand.*, 4:283–293, 1950.

[28] Joshua E Elias and Steven P Gygi. Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry. *Nat Meth*, 4(3):207–214, March 2007.

[29] Jimmy K. Eng, Ashley L. McCormack, and John R. Yates. An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *Journal of the American Society for Mass Spectrometry*, 5(11):976–989, November 1994.

[30] Magdalena Feldhahn, Philipp Thiel, Mathias M. Schuler, Nina Hillen, Stefan Stevanović, and et al. EpiToolKit–a web server for computational immunomics. *Nucleic Acids Res*, pages (advanced access, doi:10.1093/nar/gkn229), 2008.

[31] Ari Frank and Pavel Pevzner. Pepnovo: de novo peptide sequencing via probabilistic network modeling. *Anal Chem*, 77(4):964–973, Feb 2005.

[32] Ari Frank, Stephen Tanner, Vineet Bafna, and Pavel Pevzner. Peptide sequence tags for fast database search in mass-spectrometry. *J Proteome Res*, 4(4):1287–1295, 2005.

[33] Thomas Gärtner, Peter A. Flach, Adam Kowalczyk, and Alex J. Smola. Multi-instance kernels. In Claude Sammut and Achim G. Hoffmann, editors, *ICML*, pages 179–186. Morgan Kaufmann, 2002.

[34] Lewis Y Geer, Sanford P Markey, Jeffrey A Kowalak, Lukas Wagner, Ming Xu, Dawn M Maynard, Xiaoyu Yang, Wenyao Shi, and Stephen H Bryant. Open mass spectrometry search algorithm. *J Proteome Res*, 3(5):958–964, 2004.

[35] Alexander V Gorshkov, Irina A Tarasova, Victor V Evreinov, Mikhail M Savitski, Michael L Nielsen, Roman A Zubarev, and Mikhail V Gorshkov. Liquid chromatography at critical conditions: comprehensive approach to sequence-dependent retention time prediction. *Anal Chem*, 78(22):7770–7777, Nov 2006.

[36] Pingping Guan, Irini A Doytchinova, Christianna Zygouri, and Darren R Flower. MHCPred: A server for quantitative prediction of peptide-MHC binding. *Nucleic Acids Res*, 31(13):3621–3624, 2003.

[37] J. Hammer, C. Belunis, D. Bolin, J. Papadopoulos, R. Walsky, J. Higelin, W. Danho, F. Sinigaglia, and Z. A. Nagy. High-affinity binding of short peptides to major histocompatibility complex class II molecules by anchor combinations. *Proc Natl Acad Sci U S A*, 91(10):4456–4460, 1994.

[38] Xuemei Han, Mi Jin, Kathrin Breuker, and Fred W. McLafferty. Extending Top-Down Mass Spectrometry to Proteins with Masses Greater Than 200 Kilodaltons. *Science*, 314(5796):109–112, 2006.

[39] David Heckerman, Dan Geiger, and David M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, September 1995.

[40] Till Helge Helwig. Maschinelles lernen zur vorhersage proteotypischer peptide. *Bachelor Thesis*, 2008.

[41] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*, 89(22):10915–10919, Nov 1992.

[42] Tomer Hertz and Chen Yanover. Pepdist: A new framework for protein-peptide binding prediction based on learning peptide distance functions. *BMC Bioinformatics*, 7(Suppl 1):S3, 2006.

[43] 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 Trans Comput Biol Bioinformatics*, 4(2):216–226, 2007.

[44] Laurent Jacob and Jean-Philippe Vert. Efficient peptide-MHC-I binding prediction for alleles with few known binders. *Bioinformatics*, 24(3):358–366, 2008.

[45] Thorsten Joachims. Structured output prediction with support vector machines, 2006.

[46] Roman Kaliszan, Tomasz Baczek, Anna Cimochowska, Paulina Juszczyk, Kornelia Wisniewska, and Zbigniew Grzonka. Prediction of high-performance liquid chromatography retention of peptides with the use of quantitative structure-retention relationships. *Proteomics*, 5(2):409–415, 2005.

[47] Lukas Käll, Jesse D Canterbury, Jason Weston, William Stafford Noble, and Michael J MacCoss. Semi-supervised learning for peptide identification from shotgun proteomics datasets. *Nat Methods*, 4(11):923–925, Nov 2007.

[48] Lukas Käll, John D. Storey, Michael J. MacCoss, and William Stafford Noble. Assigning significance to peptides identified by tandem mass spectrometry using decoy databases. *J Proteome Res*, 7(1):29–34, 2008.

[49] Oleksiy Karpenko, Jianming Shi, and Yang Dai. Prediction of MHC class II binders using the ant colony search strategy. *Artif Intell Med*, 35(1-2):147–156, 2005.

[50] S. Kawashima, H. Ogata, and M. Kanehisa. AAindex: Amino acid index database. *Nucleic Acids Res*, 27(1):368–369, 1999.

[51] Shuichi Kawashima, Piotr Pokarowski, Maria Pokarowska, Andrzej Kolinski, Toshiaki Katayama, and Minoru Kanehisa. AAindex: amino acid index database, progress report 2008. *Nucl Acids Res*, 36(suppl_1):D202–205, 2008.

[52] Michael Kinter and Nicholas E. Sherman. *Protein Sequencing and Identification Using Tandem Mass Spectrometry*. John Wiley & Sons, 2000.

[53] Michael Kinter and Nicholas E. Sherman. *Protein Sequencing and Identification Using Tandem Mass Spectrometry*, page 15. John Wiley & Sons, 2000.

[54] Michael Kinter and Nicholas E. Sherman. *Protein Sequencing and Identification Using Tandem Mass Spectrometry*, page 32. John Wiley & Sons, 2000.

[55] A.A. Klammer, X. Yi, M.J. MacCoss, and W.S. Noble. Improving tandem mass spectrum identification using peptide retention time prediction across diverse chromatography conditions. *Anal Chem*, 79(16):6111–6118, 2007.

[56] Aaron A Klammer, Sheila M Reynolds, Jeff A Bilmes, Michael J MacCoss, and William Stafford Noble. Modeling peptide fragmentation with dynamic bayesian networks for peptide identification. *Bioinformatics*, 24(13):i348–i356, Jul 2008.

[57] Marius Kloft, Ulf Brefeld, Pavel Laskov, and Sören Sonnenburg. Non-sparse multiple kernel learning. In *NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, 2008.

[58] Oliver Kohlbacher, Knut Reinert, Clemens Gropl, Eva Lange, Nico Pfeifer, Ole Schulz-Trieglaff, and Marc Sturm. TOPP–the OpenMS proteomics pipeline. *Bioinformatics*, 23(2):e191–197, 2007.

[59] Eberhard Krause, Holger Wenschuh, and Peter R. Jungblut. The dominance of arginine-containing peptides in maldi-derived tryptic mass fingerprints of proteins. *Analytical Chemistry*, 71(19):4160–4165, 1999.

[60] Oleg V Krokhin. Sequence-specific retention calculator. algorithm for peptide retention prediction in ion-pair rp-hplc: application to 300- and 100-a pore size c18 sorbents. *Anal Chem*, 78(22):7785–7795, Nov 2006.

[61] O.V. Krokhin, R. Craig, V. Spicer, W. Ens, K. G. Standing, R. C. Beavis, and J. A. Wilkins. An Improved Model for Prediction of Retention Times of Tryptic Peptides in Ion Pair Reversed-phase HPLC: Its Application to Protein Peptide Mapping by Off-Line HPLC-MALDI MS. *Mol Cell Proteomics*, 3(9):908–919, 2004.

[62] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*, pages 481–492. University of California Press, 1951.

[63] Bernhard Kuster, Markus Schirle, Parag Mallick, and Ruedi Aebersold. Scoring proteomes with proteotypic peptide probes. *Nat Rev Mol Cell Biol*, 6(7):577–583, Jul 2005.

[64] James T. Kwok and Pak-Ming Cheung. Marginalized multi-instance kernels. In *IJCAI '07*, 2007.

[65] Eva Lange, Clemens Gröpl, Ole Schulz-Trieglaff, Andreas Leinenbach, Christian Huber, and Knut Reinert. A geometric approach for the alignment of liquid chromatography mass spectrometry data. *Bioinformatics*, 23(13):i273–281, 2007.

[66] Christina Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: a string kernel for svm protein classification. *Pac Symp Biocomput*, pages 564–575, 2002.

[67] Haifeng Li and Tao Jiang. A class of edit kernels for SVMs to predict translation initiation sites in eukaryotic mRNAs. In *RECOMB*, pages 262–271, 2004.

[68] Thomas Lingner and Peter Meinicke. Remote homology detection based on oligomer distances. *Bioinformatics*, 22(18):2224–2231, 2006.

[69] Andrew J. Link, Jimmy Eng, David M. Schieltz, Edwin Carmack, Gregory J. Mize, David R. Morris, Barbara M. Garvik, and John R. Yates. Direct analysis of protein complexes using mass spectrometry. *Nat Biotech*, 17(7):676–682, July 1999.

[70] Peng Lu, Christine Vogel, Rong Wang, Xin Yao, and Edward M Marcotte. Absolute protein expression profiling estimates the relative contributions of transcriptional and translational regulation. *Nat Biotech*, 25(1):117–124, February 2007.

[71] E. Giralt M.-L. Valero and D. Andreu. An evaluation of some structural determinants for peptide desorption in MALDI-TOF mass spectrometry. In *Peptides 1996*, pages 855–856, Kingswinford, UK, 1998. Mayflower Scientific Ltd.

[72] Michael J MacCoss, Christine C Wu, and John R Yates. Probability-based validation of protein identifications using a modified sequest algorithm. *Anal Chem*, 74(21):5593–5599, Nov 2002.

[73] Parag Mallick, Markus Schirle, Sharon S Chen, Mark R Flory, Hookeun Lee, Daniel Martin, Jeffrey Ranish, Brian Raught, Robert Schmitt, Thilo Werner, Bernhard Kuster, and Ruedi Aebersold. Computational prediction of proteotypic peptides for quantitative proteomics. *Nat Biotech*, 25(1):125–131, February 2007.

[74] C. T. Mant, T. W. Burke, J. A. Black, and R. S. Hodges. Effect of peptide chain length on peptide retention behaviour in reversed-phase chromatography. *J Chromatogr*, 458:193–205, Dec 1988.

[75] James L. Meek. Prediction of Peptide Retention Times in High-Pressure Liquid Chromatography on the Basis of Amino Acid Composition. *PNAS*, 77(3):1632–1636, 1980.

[76] Peter Meinicke, Maike Tech, Burkhard Morgenstern, and Rainer Merkl. Oligo kernels for datamining on biological sequences: a case study on prokaryotic translation initiation sites. *BMC Bioinformatics*, 5(1):169, 2004.

[77] Andreas Moll, Andreas Hildebrandt, Hans-Peter Lenhof, and Oliver Kohlbacher. Ballview: a tool for research and education in molecular modeling. *Bioinformatics*, 22(3):365–366, Feb 2006.

[78] Roger E Moore, Mary K Young, and Terry D Lee. Qscore: an algorithm for evaluating sequest database search results. *J Am Soc Mass Spectrom*, 13(4):378–386, Apr 2002.

[79] Morten Nielsen, Claus Lundegaard, Thomas Blicher, Kasper Lamberth, Mikkel Harndahl, Sune Justesen, Gustav Røder, Bjoern Peters, Alessandro Sette, Ole Lund, and Søren Buus. NetMHCpan, a method for quantitative predictions of peptide binding to any HLA-A and -B locus protein of known sequence. *PLoS ONE*, 2(8):e796, 2007.

[80] Morten Nielsen, Claus Lundegaard, and Ole Lund. Prediction of MHC class II binding affinity using SMM-align, a novel stabilization matrix alignment method. *BMC Bioinformatics*, 8:238, 2007.

[81] Morten Nielsen, Claus Lundegaard, Peder Worning, Christina Sylvester Hvid, Kasper Lamberth, Søren Buus, Soren Brunak, and Ole Lund. Improved prediction of MHC class I and class II epitopes using a novel Gibbs sampling approach. *Bioinformatics*, 20(9):1388–1397, 2004.

[82] Hideki Noguchi, Ryuji Kato, Taizo Hanai, Yukari Matsubara, Hiroyuki Honda, Vladimir Brusic, and Takeshi Kobayashi. Hidden markov model-based prediction of antigenic peptides that interact with MHC class II molecules. *J Biosci Bioeng*, 94(3):264–270, 2002.

[83] A.B.J. Novikoff. On convergence proofs on perceptrons. In *Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622, 1962.

[84] Cheolhwan Oh, Stanislaw H Zak, Hamid Mirzaei, Charles Buck, Fred E Regnier, and Xiang Zhang. Neural network prediction of peptide separation in strong anion exchange chromatography. *Bioinformatics*, 23(1):114–118, Jan 2007.

[85] Christian S. Parry, Jack Gorski, and Lawrence J. Stern. Crystallographic structure of the human leukocyte antigen DRA, DRB3*0101: Models of a directional alloimmune response and autoimmunity. *Journal of Molecular Biology*, 371(2):435–446, August 2007.

[86] Junmin Peng, Joshua E Elias, Carson C Thoreen, Larry J Licklider, and Steven P Gygi. Evaluation of multidimensional chromatography coupled with tandem mass spectrometry (LC/LC-MS/MS) for large-scale protein analysis: the yeast proteome. *J Proteome Res*, 2(1):43–50, 2003.

[87] D. N. Perkins, D. J. Pappin, D. M. Creasy, and J. S. Cottrell. Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis*, 20(18):3551–3567, Dec 1999.

[88] Bjoern Peters, Huynh-Hoa Bui, Sune Frankild, Morten Nielson, Claus Lundegaard, Emrah Kostem, Derek Basch, Kasper Lamberth, Mikkel Harndahl, Ward Fleri, Stephen S Wilson, John Sidney, Ole Lund, Soren Buus, and Alessandro Sette. A community resource benchmarking predictions of peptide binding to MHC-I molecules. *PLoS Comput Biol*, 2(6):e65, Jun 2006.

[89] Bjoern Peters, John Sidney, Phil Bourne, Huynh-Hoa Bui, Soeren Buus, and et al. The immune epitope database and analysis resource: from vision to blueprint. *PLoS Biol*, 3(3):e91, 2005.

[90] Konstantinos Petritis, Lars J Kangas, Patrick L Ferguson, Gordon A Anderson, Ljiljana Pasa-Tolic, Mary S Lipton, Kenneth J Auberry, Eric F Strittmatter, Yufeng Shen, Rui Zhao, and Richard D Smith. Use of artificial neural networks for the accurate prediction of peptide liquid chromatography elution times in proteome analyses. *Anal Chem*, 75(5):1039–1048, Mar 2003.

[91] Konstantinos Petritis, Lars J Kangas, Bo Yan, Matthew E Monroe, Eric F Strittmatter, Wei-Jun Qian, Joshua N Adkins, Ronald J Moore, Ying Xu, Mary S Lipton, David G Camp, and Richard D Smith. Improved peptide elution time prediction for reversed-phase liquid chromatography-ms by incorporating peptide sequence information. *Anal Chem*, 78(14):5026–5039, Jul 2006.

[92] Nico Pfeifer and Oliver Kohlbacher. Multiple Instance Learning Allows MHC Class II Epitope Predictions Across Alleles. *Algorithms in Bioinformatics*, pages 210–221, 2008.

[93] Nico Pfeifer, Andreas Leinenbach, Christian G. Huber, and Oliver Kohlbacher. Improving peptide identification in proteome analysis by a two-dimensional retention time filtering approach. *Journal of Proteome Research, accepted*.

[94] Nico Pfeifer, Andreas Leinenbach, Christian G. Huber, and Oliver Kohlbacher. Statistical learning of peptide retention behavior in chromatographic separations: a new kernel-based approach for computational proteomics. *BMC Bioinformatics*, 8(1):468, 2007.

[95] Menaka Rajapakse, Bertil Schmidt, Lin Feng, and Vladimir Brusic. Predicting peptides binding to MHC class II molecules using multi-objective evolutionary algorithms. *BMC Bioinformatics*, 8:459, 2007.

[96] H. G. Rammensee, T. Friede, and S. Stevanović. MHC ligands and peptide motifs: first listing. *Immunogenetics*, 41(4):178–228, 1995.

[97] G. Rätsch, S. Sonnenburg, and B. Schölkopf. Rase: recognition of alternatively spliced exons in c.elegans. *Bioinformatics*, 21(1):369–377, 2005.

[98] Soumya Ray and David Page. Multiple instance regression. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 425–432, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[99] Pedro A. Reche, John-Paul Glutting, Hong Zhang, and Ellis L. Reinherz. Enhancement to the RANKPEP resource for the prediction of peptide binding to MHC molecules using profiles. *Immunogenetics*, 56(6):405–419, 2004.

[100] Thomas J. Kindt Richard A. Goldsby and Barbara A. Osborne. *Kuby Immunology*. W.H. Freeman & Company, 5 edition, 2002.

[101] Thomas J. Kindt Richard A. Goldsby and Barbara A. Osborne. *Kuby Immunology*, page 169. W.H. Freeman & Company, 5 edition, 2002.

[102] James Robinson, Matthew J Waller, Peter Parham, Natasja de Groot, Ronald Bontrop, and et al. IMGT/HLA and IMGT/MHC: sequence databases for the study of the major histocompatibility complex. *Nucleic Acids Res*, 31(1):311–314, 2003.

[103] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[104] Rovshan G Sadygov and John R Yates. A hypergeometric probability model for protein identification and validation using tandem mass spectral data and protein sequence databases. *Anal Chem*, 75(15):3792–3798, Aug 2003.

[105] Jesper Salomon and Darren Flower. Predicting class II MHC-peptide binding: a kernel based approach using similarity scores. *BMC Bioinformatics*, 7(1):501, 2006.

[106] Christian Schley, Remco Swart, and Christian G Huber. Capillary scale monolithic trap column for desalting and preconcentration of peptides and proteins in one- and two-dimensional separations. *J Chromatogr A*, 1136(2):210–220, Dec 2006.

[107] Susanne Schneiker, Olena Perlova, Olaf Kaiser, Klaus Gerth, Aysel Alici, Matthias O Altmeyer, Daniela Bartels, Thomas Bekel, Stefan Beyer, Edna Bode, Helge B Bode, Christoph J Bolten, Jomuna V Choudhuri, Sabrina Doss, Yasser A Elnakady, Bettina Frank, Lars Gaigalat, Alexander Goesmann, Carolin Groeger, Frank Gross, Lars Jelsbak, Lotte Jelsbak, Jorn Kalinowski, Carsten Kegler, Tina Knauber, Sebastian Konietzny, Maren Kopp, Lutz Krause, Daniel Krug, Bukhard Linke, Taifo Mahmud, Rosa Martinez-Arias, Alice C McHardy, Michelle Merai, Folker Meyer, Sascha Mormann, Jose Munoz-Dorado, Juana Perez, Silke Pradella, Shwan Rachid, Gunter Raddatz, Frank Rosenau, Christian Ruckert, Florenz Sasse, Maren Scharfe, Stephan C Schuster, Garret Suen, Anke Treuner-Lange, Gregory J Velicer, Frank-Jorg Vorholter, Kira J Weissman, Roy D Welch, Silke C Wenzel, David E Whitworth, Susanne Wilhelm, Christoph Wittmann, Helmut Blocker, Alfred Puhler, and Rolf Muller. Complete genome sequence of the myxobacterium sorangium cellulosum. *Nat Biotech*, 25(11):1281–1289, November 2007.

[108] I. J. Schoenberg. Metric spaces and positive definite functions. *Trans Amer Math Soc*, 44(3):522–536, 1938.

[109] Bernhard Schölkopf, Ralf Herbrich, and Alex Smola. A generalized representer theorem. *Computational Learning Theory*, pages 416–426, 2001.

[110] Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Computation*, 12(5):1207–1245, 2000.

[111] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[112] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Kernel Methods in Computational Biology*. MIT press, New York, NY, USA, 2004.

[113] Ole Schulz-Trieglaff, Nico Pfeifer, Clemens Gropl, Oliver Kohlbacher, and Knut Reinert. LC-MSsim - a simulation software for liquid chromatography mass spectrometry data. *BMC Bioinformatics*, 9(1):423, 2008.

[114] Alessandro Sette and John Fikes. Epitope-based vaccines: an update on epitope identification, vaccine design and delivery. *Curr Opin Immunol*, 15(4):461–470, Aug 2003.

[115] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

[116] H. Singh and G. P. Raghava. ProPred: prediction of HLA-DR binding sites. *Bioinformatics*, 17(12):1236–1237, 2001.

[117] Sören Sonnenburg, Gunnar Rätsch, Christin Schäfer, and Bernhard Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, 2006.

[118] Sören Sonnenburg, Alexander Zien, Petra Philips, and Gunnar Rtsch. POIMs: positional oligomer importance matrices–understanding support vector machine-based signal detectors. *Bioinformatics*, 24(13):i6–14, Jul 2008.

[119] Ingo Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Trans Inform Theory*, 51(1):128–142, 2005.

[120] John D. Storey and Robert Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences of the United States of America*, 100(16):9440–9445, 2003.

[121] Eric F Strittmatter, Lars J Kangas, Konstantinos Petritis, Heather M Mottaz, Gordon A Anderson, Yufeng Shen, Jon M Jacobs, David G Camp, and Richard D Smith. Application of peptide lc retention time information in a discriminant function for peptide identification by tandem mass spectrometry. *J Proteome Res*, 3(4):760–769, 2004.

[122] Marc Sturm, Andreas Bertsch, Clemens Gröpl, Andreas Hildebrandt, Rene Hussong, Eva Lange, Nico Pfeifer, Ole Schulz-Trieglaff, Alexandra Zerck, Knut Reinert, and Oliver Kohlbacher. OpenMS - an open-source software framework for mass spectrometry. *BMC Bioinformatics*, 9(1):163, 2008.

[123] Marc Sturm and Oliver Kohlbacher. Toppview: An open-source viewer for mass spectrometry data. *Journal of Proteome Research, accepted*.

[124] T. Sturniolo, E. Bono, J. Ding, L. Raddrizzani, O. Tuereci, and et al. Generation of tissue-specific and promiscuous HLA ligand databases using DNA microarrays and virtual HLA class II matrices. *Nat Biotechnol*, 17(6):555–561, 1999.

[125] Haixu Tang, Randy J Arnold, Pedro Alves, Zhiyin Xun, David E Clemmer, Milos V Novotny, James P Reilly, and Predrag Radivojac. A computational approach toward label-free protein quantification using predicted peptide detectability. *Bioinformatics*, 22(14):e481–e488, Jul 2006.

[126] Adi L Tarca, Vincent J Carey, Xue-wen Chen, Roberto Romero, and Sorin Drăghici. Machine learning and its applications to biology. *PLoS Comput Biol*, 3(6):e116–, June 2007.

[127] J. A. Taylor and R. S. Johnson. Sequence database searches via de novo peptide sequencing by tandem mass spectrometry. *Rapid Commun Mass Spectrom*, 11(9):1067–1075, 1997.

[128] Maike Tech, Nico Pfeifer, Burkhard Morgenstern, and Peter Meinicke. Tico: a tool for improving predictions of prokaryotic translation initiation sites. *Bioinformatics*, 21(17):3568–3569, Sep 2005.

[129] Hansjörg Toll, Reiner Wintringer, Ulrike Schweiger-Hufnagel, and Christian G Huber. Comparing monolithic and microparticular capillary columns for the separation and analysis of peptide mixtures by liquid chromatography-mass spectrometry. *J Sep Sci*, 28(14):1666–1674, Sep 2005.

[130] S. L. Topalian. MHC class II restricted tumor antigens and the role of CD4+ T cells in cancer immunotherapy. *Curr Opin Immunol*, 6(5):741–745, 1994.

[131] Nora C. Toussaint, Pierre Dönnes, and Oliver Kohlbacher. A mathematical framework for the selection of an optimal set of peptides for epitope-based vaccines. *PLoS Comput Biol*, 4(12):e1000246, 12 2008.

[132] Nora C Toussaint and Oliver Kohlbacher. Optitope–a web server for the selection of an optimal set of peptides for epitope-based vaccines. *Nucleic Acids Res*, May 2009.

[133] Nkemdilim C. Uwaje, Nikola S. Mueller, Giuseppina Maccarrone, and Christoph W. Turck. Interrogation of MS/MS search data with an pI Filter algorithm to increase protein identification success. *ELECTROPHORESIS*, 28(12):1867–1874, 2007.

[134] Vladimir Vacic, Lilia M Iakoucheva, and Predrag Radivojac. Two sample logo: a graphical representation of the differences between two sets of sequence alignments. *Bioinformatics*, 22(12):1536–1537, Jun 2006.

[135] Mathura S. Venkatarajan and Werner Braun. New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physical-chemical properties. *Journal of Molecular Modeling*, 7(12):445–453, 2001.

[136] J.-P. Vert, H. Saigo, and T. Akutsu. *Local alignment kernels for biological sequences*, pages 131–154. Kernel Methods in Computational Biology. MIT Press, 2004.

[137] Ji Wan, Wen Liu, Qiqi Xu, Yongliang Ren, Darren R Flower, and Tongbin Li. SVRMHC prediction server for MHC-binding peptides. *BMC Bioinformatics*, 7:463, 2006.

[138] Peng Wang, John Sidney, Courtney Dow, Bianca Mothé, Alessandro Sette, and Bjoern Peters. A systematic assessment of MHC class II peptide binding predictions and evaluation of a consensus approach. *PLoS Comput Biol*, 4(4):e1000048, 2008.

[139] Michael P. Washburn, Dirk Wolters, and John R. Yates. Large-scale analysis of the yeast proteome by multidimensional protein identification technology. *Nat Biotech*, 19(3):242–247, March 2001.

[140] Bobbie-Jo M Webb-Robertson, William R Cannon, Christopher S Oehmen, Anuj R Shah, Vidhya Gurumoorthi, Mary S Lipton, and Katrina M Waters. A support vector machine model for the prediction of proteotypic peptides for accurate mass and time proteomics. *Bioinformatics*, 24(13):1503–1509, Jul 2008.

[141] C. M. Whitehouse, R. N. Dreyer, M. Yamashita, and J. B. Fenn. Electrospray interface for liquid chromatographs and mass spectrometers. *Anal Chem*, 57(3):675–679, Mar 1985.

[142] J. R. Yates. Mass spectrometry and the age of the proteome. *J Mass Spectrom*, 33(1):1–19, Jan 1998.

[143] Noah Zaitlen, Manuel Reyes-Gomez, David Heckerman, and Nebojsa Jojic. Shift-invariant adaptive double threading: Learning MHC II - peptide binding. In *RECOMB*, pages 181–195, 2007.

[144] Xiang H-F Zhang, Katherine A Heller, Ilana Hefter, Christina S Leslie, and Lawrence A Chasin. Sequence information for the splicing of human pre-mrna identified by support vector machine classification. *Genome Res*, 13(12):2637–2650, Dec 2003.

[145] Alexander Zien and Cheng Soon Ong. Multiclass multiple kernel learning. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 1191–1198, New York, NY, USA, 2007. ACM.

[146] Alexander Zien, Gunnar Rätsch, Sebastian Mika, Bernhard Schölkopf, Thomas Lengauer, and Klaus-Robert Müller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, 2000.

Appendix A

Abbreviations

AI	Artificial Intelligence
CID	Collision-Induced Dissociation
CTL	Cytotoxic T Lymphocyte
CV	Cross-Validation
ESI	Electro Spray Ionization
HLA	Human Leukocyte Antigen
HPLC	High Performance Liquid Chromatography
ICAT	Isotope-Coded Affinity Tag
IP	Ion-Pair
LC	Liquid Chromatography
MALDI	Matrix-Assisted Laser Desorption/Ionization
MHC	Major Histocompatibility Complex
MHCI	Major Histocompatibility Complex Class I
MHCII	Major Histocompatibility Complex Class II
MS	Mass Spectrometry
MSA	Multiple Sequence Alignment
MSDB	Mass Spectrometry Database
MudPIT	Multidimensional Protein Identification Technology
NSK	Normalized Set Kernel
OBK	Oligo-Border Kernel
PAGE	Polyacrylamide Gel Electrophoresis

PDB ID	Protein Data Bank Identifier
POBK	Paired Oligo-Border Kernel
R	Pearson Correlation Coefficient
RBF	Radial Basis Function
RKHS	Reproducing Kernel Hilbert Space
RP	Reversed-Phase
RT	Retention Time
SAX	Strong Anion Exchange
SPE	Solid Phase Extraction
SVM	Support Vector Machine
SVR	Support Vector Regression
SR	Classification Success Rate
TOF	Time-of-Flight
UML	Unified Modeling Language
WD	Weighted Degree

Appendix B

Publications

B.1 Published Manuscripts

1. O. Schulz-Trieglaff, N. Pfeifer, C. Gröpl, O. Kohlbacher and K. Reinert. **LC-MSsim - a simulation software for Liquid Chromatography Mass Spectrometry data**. BMC Bioinformatics 2008, 9:423

2. N. Pfeifer and O. Kohlbacher. **Multiple Instance Learning Allows MHC Class II Epitope Predictions across Alleles**. Proceedings of WABI 2008, Lecture Notes in Computer Science, 2008, 5251:210-221.

3. M. Sturm, A. Bertsch, C. Gröpl, A. Hildebrandt, R. Hussong, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, A. Zerck, K. Reinert, O. Kohlbacher. **OpenMS - An open-source software framework for mass spectrometry**. BMC Bioinformatics 2008, 9:163

4. N. Pfeifer, A. Leinenbach, C. G. Huber and O. Kohlbacher. **Statistical learning of peptide retention behavior in chromatographic separations: A new kernel-based approach for computational proteomics**. BMC Bioinformatics 2007, 8:468

5. 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 2007, Vol. 4, No. 2:216-226

6. O. Kohlbacher, K. Reinert, C. Gröpl, E. Lange, N. Pfeifer, O. Schulz-Trieglaff and M. Sturm. **TOPP - The OpenMS Proteomics Pipeline**. Bioinformatics 2007 23 (ECCB 2006 Conference Proceedings), e177 - e183

7. M. Tech, N. Pfeifer, B. Morgenstern, P. Meinicke. **TICO: a tool for improving predictions of prokaryotic translation initiation sites**. Bioinformatics 2005 21, 3568 - 3569

B.2 Accepted Manuscripts

1. N. Pfeifer, A. Leinenbach, C. G. Huber and O. Kohlbacher. **Improving Peptide Identification in Proteome Analysis by a Two-Dimensional Retention Time Filtering Approach**. Journal of Proteome Research 2009, accepted, pre-print available at `http://dx.doi.org/10.1021/pr900064b`

Appendix C

Contributions

At all research topics I always talked about the latest ideas to my supervisor Prof. Dr. Oliver Kohlbacher. These discussions always lead to new ideas or directions.

1. **Section 3.1- A New Kernel for Chromatographic Separation Prediction**.

   The main ideas of this work were presented in a journal article in 2007 [94]. A short introduction is published in Reference [122] and preliminary ideas were presented in Reference [58]. The contributions, as mentioned in the paper were: OK and CH designed the experiment and the study. AL was responsible for the experimental data generation. NP developed and implemented the theoretical methods and performed the data evaluation.

2. **Section 3.2- Two-Dimensional Chromatographic Separation Prediction**.

   This work is accepted in a similar form at the Journal of Proteome Research [93]. The contributions were the same as in the work presented in Section 3.1.

3. **Section 3.3- Prediction of Proteotypic Peptides**.

   This work started in 2008, when Ole Schulz-Trieglaff asked me whether we could predict proteotypic peptides with OpenMS. The first evaluations were published in [113] and included into a simulator for LC-MS maps called LC-MSsim [113]. After the publication we wanted to assess the performance of different approaches more transparently. This was pursued in the bachelor thesis of Till Helge Helwig [40], which I supervised. The evaluation of the *POBK* and *OBK* as presented in this thesis were not presented in Reference [40]. The visualization results presented in this thesis are also not yet presented anywhere else.

4. **Chapter 4- Applications in Immunomics**.

   Parts of this work were presented at WABI 2008 [92]. The introduction of the aggregating predictor is not published anywhere else and the proof of the positive semi-definiteness of the positionally-weighted RBF kernel is also new. We only presented a sketch of the proof in Reference [92].

Index