2008

Machine learning approaches for epitope prediction

Yasser Mohamed El-manzalawy
Iowa State University

Machine learning approaches for epitope prediction

by

Yasser EL-Manzalawy

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Iowa State University

Ames, Iowa

2008

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my advisor, Dr. Vasant Honavar, for his constant encouragement, mentoring, and support. This work would not have been possible without the constructive and insightful discussions that I have had with him over the past few years. I gratefully thank him for being always accessible and allowing me to interrupt him numerous times, regardless of how busy he was, to get his invaluable feedback on some recent results or some new directions that I might explore.

My earnest thanks also goes to my program of study committee members for serving on my supervising committee and for making constructive suggestions. My sincere thanks and appreciation goes to Dr. Drena Dobbs for encouraging me to work on Bioinformatics related problems, being always available to answer my biological questions, and for her collaborations and invaluable comments and suggestions in most of my written manuscripts. I would like to thank Dr. David Fernández-Baca for his useful comments and his interesting lectures about scoring matrices and hidden Markov models, which greatly benefited me in my research. Many thanks to Dr. Dimitris Margaritis for helping me in my first semester at Iowa State University and for his constant encouragement. I would also like to express my gratitude to Dr. Ahmed Kamal for serving on my supervising committee and for his encouragement and support.

Many thanks go to the members of the Artificial Intelligence Research Lab and Dobbs-Honavar group for many rewarding research discussions and helpful comments. Special thanks goes to Adrian Silvescu, Cornelia Caragea, and Oksana Yakhnenko for useful discussions and suggestions. I am very grateful to Fadi Towfic for proofreading many of my manuscripts and parts of this dissertation. Thanks are also due to Lanette Woodard and Linda Dutton for their assistance and guidance in many matters.

Finally, I would like to express my gratitude to my Master thesis advisor, Dr. Hani Harb, for pointing me toward the right path to starting a career in academia and for his encouragement and support. I would also like to take this opportunity to thank Dr. Mohamed Zaki for aiding me in obtaining a doctoral fellowship from the Egyptian Government and for his constant encouragement and support during my few years at Iowa State University.

I can not end without thanking my family, on whose constant encouragement and love I have relied during my Ph.D. years and otherwise.

This research was supported in part by a doctoral fellowship from the Egyptian Government to Yasser El-Manzalawy and a grant from the National Institutes of Health (GM066387) to Vasant Honavar.

ABSTRACT

The identification and characterization of epitopes in antigenic sequences is critical for understanding disease pathogenesis, for identifying potential autoantigens, and for designing vaccines and immune-based cancer therapies. As the number of pathogen genomes fully or partially sequenced is rapidly increasing, experimental methods for epitope mapping would be prohibitive in terms of time and expenses. Therefore, computational methods for reliably identifying potential vaccine candidates (i.e., epitopes that invoke strong response from both T-cells and B-cells) are highly desirable.

Machine learning offers one of the most cost-effective and widely used approaches to developing epitope prediction tools. In the last few years, several advances in machine learning research have emerged. We utilize recent advances in machine learning research to provide epitope prediction tools with improved predictive performance. First, we introduce two methods, BCPred and FBCPred, for predicting linear B-cell epitopes and flexible length linear B-cell epitopes, respectively, using string kernel based support vector machine (SVM) classifiers. Second, we introduce three scoring matrix methods and show that they are highly competitive with a broad class of machine learning methods, including SVM, in predicting major histocompatibility complex class I (MHC-I) binding peptides. Finally, we formulate the problems of qualitatively and quantitatively predicting flexible length major histocompatibility complex class II (MHC-II) peptides as multiple instance learning and multiple instance regression problems, respectively. Based on this formulation, we introduce MHCMIR, a novel method for predicting MHC-II binding affinity using multiple instance regression.

The development of reliable epitope prediction tools is not feasible in the absence of high quality data sets. Unfortunately, most of the existing epitope benchmark data sets are comprised of epitope sequences that share high degree of similarity with other peptide sequences in the same data set. We demonstrate the pitfalls of these commonly used data sets for evaluating the performance of machine learning approaches to epitope prediction. Finally, we propose a similarity reduction procedure that is more stringent than currently used similarity reduction methods.

CHAPTER 1. GENERAL INTRODUCTION

Introduction

The immune system is a network of cells, tissues, and organs that work together to defend organisms against attacks by pathogens (e.g., bacteria, viruses, parasites, and fungi). The immune system is composed of two major subdivisions, the innate immune system and the adaptive immune system. The innate immune system is our first line of defense against invading pathogens, while the adaptive immune system acts as a second line of defense and also affords protection against re-exposure to the same pathogen. The adaptive immune system has two arms: B-cells which secret antibodies and can neutralize pathogens outside the cells; and T-cells which eliminate infected or malfunctioning cells and provide help to other immune system responses.

Adaptive immunity relies on the capability of immune cells to distinguish between self cells and foreign cells (e.g., bacteria and virally infected host cells). The process of recognizing non-self cells is called antigen presentation. An antigen is anything that can trigger an adaptive immune system response. B-cells recognize antigens on the surface of the pathogen in their naive form, while T-cells recognize their cognate antigen in a processed form as a peptide in the context of an MHC molecule. Peptides presented by MHC class I (MHC-I) molecules are derived from proteasomal degradation of intracellular proteins and their lengths range from 8 to 11 amino acids. Peptides presented by MHC class II (MHC-II) molecules are derived from endogenous proteins or intracellular pathogens and the binding peptides range from 11 to 30 residues in length. The recognition of Peptide-MHC complexes aids killer T-cells in identifying and destroying abnormal or foreign cells. Peptides that can complete this pathway are called T-cell epitopes. The identification and characterization of B-cell epitopes and MHC binding peptides is critical for understanding disease pathogenesis, for identifying potential autoantigens, and for designing vaccines and immune-based cancer therapies. As the number of pathogen genomes fully or partially sequenced is rapidly increasing, experimental methods for epitope mapping would be prohibitive in terms of time and expenses. Therefore, computational methods for reliably identifying potential vaccine candidates (i.e., epitopes that invoke strong response from both T-cells and B-cells) are highly desirable.

Several prediction methods have been proposed in the literature for predicting B-cell epitopes and MHC binding peptides (a survey of these methods is provided in the next section). However, several recent studies have pointed the limitation of existing methods in reliably identifying potential epitopes (Blythe and Flower, 2005; Greenbaum et al., 2007; EL-Manzalawy et al., 2008c; Wang et al., 2008; Gowthaman and Agrewala, 2008; EL-Manzalawy et al., 2008a). Therefore, more efforts are urgently needed to introduce epitope prediction tools with a better predictive performance, and hence, more reliability.

Machine learning offers one of the most cost-effective, and hence widely used approaches to developing epitope prediction tools. In the last few years, several advances in the machine learning research have emerged. For example, an extensive number of customized kernels have been proposed and shown more effective than general purpose kernels (e.g., polynomial and radial bias function kernels) (Lodhi et al., 2002; Saigo et al., 2004; Gärtner et al., 2003), multiple instance learning (MIL) (Dietterich et al., 1997; Gartner et al., 2002; Chen et al., 2006) has been proposed as an approach for learning from ambiguous or partially labeled data, developing classifiers that explicitly optimize the area under ROC curve (AUC) has been shown to be more useful than optimizing the classification error in cases where the AUC is the performance metric of interest (Yan et al., 2003; Brefeld and Scheffer, 2005), and ensemble learning methods (e.g., boosting and bagging) have been shown effective in improving the performance of a single learning algorithm (Freund and Schapire, 1996; Breiman, 1996). More details about these machine learning methods are given in the following chapters.

In this dissertation, we utilize recent advances in the machine learning research to provide epitope prediction tools with improved predictive performance. First, we introduce two methods, BCPred and FBCPred, for predicting linear B-cell epitopes and flexible length linear B-cell epitopes, respectively, using string kernel based support vector machine (SVM) classifiers. Second, we introduce three scoring matrix methods and show that they are highly competitive with a broad class of machine learning methods, including SVM, in predicting major histocompatibility complex class I (MHC-I) binding peptides. Finally, we formulate the problems of qualitatively and quantitatively predicting flexible length major histocompatibility complex class II (MHC-II) peptides as multiple instance learning and multiple instance regression problems, respectively. Based on this formulation, we introduce MHCMIR, a novel method for predicting MHC-II binding affinity using multiple instance regression.

The development of reliable epitope prediction tools is not feasible in the absence of high quality data sets. Unfortunately, most of the existing epitope benchmark data sets are comprised of epitope sequences that share high degree of similarity with other peptide sequences in the same data set. We demonstrate the pitfalls of these commonly used data sets for evaluating the performance of machine learning approaches to epitope prediction. Finally, we propose a similarity reduction procedure that is more stringent than currently used similarity reduction methods.

Related work

Predicting B-cell epitopes

B-cell epitopes can be classified into two types: linear (continuous) epitopes and conformational (discontinuous) epitopes. Linear epitopes are short peptides, corresponding to a contiguous amino acid sequence fragment of a protein (Barlow et al., 1986; Langeveld et al., 2001). In contrast, conformational epitopes are composed of amino acids that are not contiguous in primary sequence, but are brought into close proximity within the folded protein structure. Although it is believed that a large majority of B-cell epitopes are discontinuous (Walter, 1986), experimental epitope identification has focused primarily on linear B-cell epitopes (Flower, 2007).

Several studies have reported correlations between certain physicochemical properties of amino acids and the locations of linear B-cell epitopes within protein sequences (Pellequer et al., 1991; Parker and Guo, 1986; Karplus and Schulz, 1985; Emini et al., 1985; Pellequer et al., 1993), and several epitope prediction methods based on physicochemical properties of amino acids have been proposed. For example, hydrophilicity, flexibility, turns, or solvent accessibility propensity scales were used in the methods of Parker and Guo (1986), Karplus and Schulz (1985), Pellequer et al. (1993) and Emini et al. (1985), respectively. PREDITOP (Pellequer and Westhof, 1993), PEOPLE (Alix, 1999), BEPITOPE (Odorico and Pellequer, 2003), and BcePred (Saha and Raghava, 2004) predict linear B-cell epitopes based on groups of physicochemical properties instead of a single property.

Recently, Blythe and Flower (2005) performed an exhaustive assessment of 484 amino acid propensity scales, combined with ranges of profile parameters, to examine the correlation between propensity scale-based profiles and the location of linear B-cell epitopes in a set of 50 proteins. They reported that for predicting B-cell epitopes based on amino acid sequence information, even the best combinations of amino acid propensities performed only marginally better than random. They concluded that the reported performance of such methods in the literature is likely to have been overly optimistic, in part due to the small size of the data sets on which the methods had been evaluated.

Motivated by Blythe and Flower (2005) results and the increasing availability of experimentally identified linear B-cell epitopes, several studies have attempted to improve the accuracy of linear B-cell epitope prediction methods using machine learning approaches. BepiPred (Larsen et al., 2006) combines two amino acid propensity scales and a Hidden Markov Model (HMM) trained on linear epitopes to yield a slight improvement in prediction accuracy relative to techniques that rely on analysis of amino acid physicochemical properties. ABCPred (Saha and Raghava, 2006b) uses artificial neural networks for predicting linear B-cell epitopes. Both feed-forward and recurrent neural networks were evaluated on a *non-redundant* data set of 700 B-cell epitopes and 700 non-epitope peptides, using 5-fold cross validation tests. Input sequence windows ranging from 10 to 20 amino acids, were tested and the best performance, 66% accuracy, was obtained using a recurrent neural network trained on peptides 16 amino acids in length. In the method of Söllner and Mayer (2006), each epitope is represented using a set of 1487 features extracted from a variety of propensity scales, neighborhood matrices, and respective probability and likelihood values. Of two machine learning methods tested, decision trees and a nearest-neighbor method combined with feature selection, the latter was reported to attain an accuracy of 72% on a data set of 1211 B-cell epitopes and 1211 non-epitopes, using a 5-fold cross validation test (Söllner and Mayer, 2006). Chen et al. (2007) observed that certain amino acid pairs (AAPs) tend to occur more frequently in B-cell epitopes than in non-epitope peptides. Using an AAP propensity scale based on this observation, in combination with a support vector machine (SVM) classifier, they reported prediction accuracy of 71% on a data set of 872 B-cell epitopes and 872 non-B-cell epitopes, estimated using 5-fold cross validation. In addition, Chen et al. (2007) demonstrated an improvement in the prediction accuracy, 72.5%, when the APP propensity scale is combined with turns, accessibility, antigenicity, hydrophilicity, and flexibility propensity scales. Söllner et al. (2008) analyzed a subgroup of linear B-cell epitopes, protective epitopes, and showed that protective linear B-cell epitopes are more likely to be located in regions with high antigenicity, low sequence variability, and absence of post-translational modification patterns.

Because the number of available antigen-antibody complexes in protein data bank (PDB) is limited, only few methods for predicting conformational B-cell epitopes using structure information have been proposed. Conformational epitope prediction (CEP) (Kulkarni-Kale et al., 2005) utilizes an algorithm for predicting conformational and continuous B-cell epitope using accessibility of residues in the antigen and spatial distance cut-off. DiscoTope (Haste Andersen et al., 2006) is a method based on a weighted linear combination of sequentially smoothed epitope log-odd ratios propensity scale and some structure-based information (number of intramolecular C_α atom contacts for each residue). Even in the case of linear B-cell epitopes, however, antibody-antigen interactions are often conformation-dependent. The conformation-dependent aspect of antibody binding complicates the problem of B-cell epitope prediction, making it less tractable than T-cell epitope prediction.

Predicting T-cell epitopes

T-cells, a major type of the immune system cells, play a central role in the cell-mediated immunity (Janeway et al., 2004). Cytotoxic T-cells attack cells that have certain foreign or abnormal molecules on their surfaces. They have also been implicated in transplant rejection. Helper T-cells, or CD4+ T-cells, coordinate immune responses by communicating with other cells. Once activated, they divide rapidly and secrete cytokines that regulate the immune response. T-cells are also targets of HIV infection, with the loss of CD4+ T-cells being associated with the appearance of AIDS symptoms. Regulatory T-cells are believed to be crucial for the maintenance of immunological tolerance. T-cells epitopes are short linear peptides that are generated by the cleavage of antigenic proteins. The identification of T-cell epitopes in protein sequences is important for understanding disease pathogenesis, for identifying potential autoantigens, and for designing vaccines and immune-based cancer therapies. Predicting whether a given peptide will bind to a specific major histocompatibility complex (MHC) molecule (and the binding affinity) is an important step in identifying potential T-cell epitopes.

There are two classes of MHC molecules: MHC class I (MHC-I) molecules that are characterized by short binding peptides, usually consisting of 9 amino acid residues; and MHC class II (MHC-II) molecules that bind to peptides of variable length. MHC-II binding peptides typically vary from 11 to 30 amino acids in length, although shorter and longer MHC-binding peptides are not entirely uncommon (Rammensee et al., 1995). MHC-II molecules allow variable length peptides to bind because the binding groove of MHC-II molecule is open at both ends. However, it has been reported that a 9-mer core region is essential for MHC-II binding activity of peptides (Madden, 1995; Rammensee et al., 1995). Because the precise location of the 9-mer core region of the MHC-II binding peptide is unknown, predicting MHC-II binding peptides is more challenging than predicting MHC-I binding peptides.

A variety of methods for predicting MHC-I binding peptides from amino acid sequence information have been proposed. Examples of these MHC-I prediction methods include methods based on: scoring matrices (Parker et al., 1994; Rammensee et al., 1999; Reche et al., 2004; Bui et al., 2005; Peters and Sette, 2005); hidden Markov models (HMM) (Mamitsuka, 1998); additive method (Hattotuwagama et al., 2004); artificial neural networks (ANN) (Buus et al., 2003); support vector machine (SVM) (Donnes and Kohlbacher, 2006); support vector regression (SVR) (Liu et al., 2006). These methods can be categorized into two major types: i) qualitative methods (e.g., (Reche et al., 2004; Donnes and Kohlbacher, 2006)), which predict whether a test peptide is an MHC-I binder or non-binder; ii) quantitative methods (e.g., (Hattotuwagama et al., 2004; Liu et al., 2006; Peters and Sette, 2005)), which predicts the value of the binding affinity (e.g., IC50) of a test peptide.

Similarly, computational methods for predicting MHC-II can be categorized into:

- Qualitative MHC-II binding peptide prediction methods. Examples of such methods include: (i) methods that use a position weight matrix to model ungapped multiple sequence alignment of MHC binding peptides (Reche et al., 2004; Singh and Raghava, 2001; Nielsen et al., 2004; Rajapakse et al., 2007; Nielsen et al., 2007), or rely on Hidden Markov Models (HMMs) (Mamitsuka, 1998; Noguchi et al., 2002); (ii) supervised machine learning methods based on Artificial Neural Networks (ANN) (Nielsen et al., 2003; Buus et al., 2003) or Support Vector Machines (SVMs) (Donnes and Kohlbacher, 2006; Bhasin and Raghava, 2004; Cui et al., 2006a; Salomon and Flower, 2006); and (iii) semi-supervised machine learning methods (Murugan and Dai, 2005; Hertz and Yanover, 2006).

- Quantitative MHC-II binding peptide prediction methods. Examples of such methods include PLS-ISC (Doytchinova and Flower, 2003), MHCPred (Hattotuwagama et al., 2004), SVRMHC (Liu et al., 2006), ARB (Bui et al., 2005), and NetMHCII (Nielsen et al., 2007).

Most of the currently available MHC-II binding prediction methods focus on identifying a putative 9-mer MHC-II binding core region, e.g., based on the degree of match with a 9-mer MHC-II binding motif, typically constructed using one of the motif finding algorithms. For example, MEME (Bailey and Elkan, 1995), Gibbs sampling (Lawrence et al., 1993), matrix optimization techniques (MOTs) (Singh and Raghava, unpublished data), evolutionary algorithms (Fonseca and Fleming, 1993), Mont Carlo (MC) search (Metropolis et al., 2004), and linear programming (Bennett and Mangasarian, 1992) form the basis of MHC-II binding peptide prediction methods RankPEP (Reche et al., 2004), Gibbs (Nielsen et al., 2004), HLA-DR4Pred (Bhasin and Raghava, 2004), MOEA (Rajapakse et al., 2007), NetMHCII (Nielsen et al., 2007), and LP (Murugan and Dai, 2005), respectively. The success of these MHC-II prediction methods in identifying MHC-II peptides relies on the effectiveness of the corresponding motif-finding methods in recognizing the motif that characterizes the 9-mer core of MHC-II binding peptides. An inherent limitation of such MHC-II prediction methods is their inability to make use of any potentially useful signals that lie outside the 9-mer core region (Chang et al., 2006; Nielsen et al., 2007). Moreover, Wang et al. (2008) showed that existing MHC-II prediction tools lack consistency in identifying the 9-mer binding cores.

Recently, two methods (Cui et al., 2006a; Salomon and Flower, 2006) for predicting flexible length MHC-II peptides have been proposed. Both methods use the entire sequences of MHC-II peptides (as opposed to only the 9-mer cores) for training MHC-II binding peptide predictors. The first method (Cui et al., 2006a) maps a variable length peptide into a fixed length feature vector obtained from sequence-derived structural and physicochemical properties of the peptide. The second method (Salomon and Flower, 2006) uses a sequence kernel that defines the pair-wise similarity of variable-length peptides as the average score of all possible local alignments between the corresponding amino acid sequences.

Dissertation organization

The major aim of this dissertation is to improve the predictive performance, and hence, the reliability of epitope prediction tools. We focus on machine learning methods trained using only amino acid sequence information for training the predictors. To facilitate our major aim, we divided it into three sub-aims: i) improving the predictive performance of linear B-cell epitope prediction methods (Chapters 2 and 3); ii) improving the predictive performance of MHC-I binding peptide prediction methods (Chapter 4); iii) improving the predictive performance of MHC-II binding peptide prediction methods (Chapters 5 and 6). The following is the outline of this dissertation.

Chapter 1: We present the epitope prediction problem, related work, and the outline of the dissertation.

Chapter 2: We propose BCPred, a novel method for predicting linear B-cell epitopes using the subsequence kernel. We show that the predictive performance of BCPred (AUC = 0.76) outperforms 11 SVM-based classifiers developed and evaluated in our experiments as well as our implementation of the amino acid pair (AAP) propensity method (Chen et al., 2007) (AUC = 0.70). The results have been published in the Journal of molecular recognition (EL-Manzalawy et al., 2008d). Yasser EL-Manzalawy conceived the study, implemented the algorithms, analyzed the results, and prepared the manuscript. Drena Dobbs and Vasant Honavar contributed to the design of experiments, interpretation of results, and preparation of the manuscript.

Chapter 3: We extend the work presented in Chapter 2 by introducing the problem of predicting flexible length linear B-cell epitopes. We explore two machine learning approaches for predicting flexible length linear B-cell epitopes. The first approach utilizes four sequence kernels for determining a similarity score between any arbitrary pair of variable length sequences. The second approach utilizes four different methods of mapping a variable length sequence into a fixed length feature vector. Based on our results, we propose FBCPred, a method for predicting flexible length linear B-cell epitopes using string kernels. The results have been published in the 7th International Conference on Computational Systems Bioinformatics (CSB'08) (EL-Manzalawy et al., 2008b). Yasser EL-Manzalawy conceived the study, implemented the algorithms, analyzed the results, and prepared the manuscript. Drena Dobbs and Vasant Honavar contributed to the design of experiments, interpretation of results, and preparation of the manuscript.

Chapter 4: We introduce three scoring matrix based methods: i) modified position specific scoring matrix (MPSSM), a method for computing a PSSM from both binding and non-binding training peptides; ii) area under ROC curve (AUC) optimized matrix method (AOMM), a method for finding a scoring matrix that maximizes the AUC over the training data; iii) SMM-

Bin, a method for qualitative via quantitative (QVQ) prediction using the Stabilized Matrix Method (SMM) (Peters and Sette, 2005). We compare these methods with 10 different MHC-I prediction methods covering major approaches for predicting MHC-I binding peptides. The results of our experiments show that advanced scoring matrix based machine learning methods (e.g., AOMM, SMM) are highly competitive with a broad class of machine learning methods for predicting MHC-I peptides. Yasser EL-Manzalawy conceived the study, implemented the algorithms, analyzed the results, and prepared the manuscript. Vasant Honavar contributed to the design of experiments, interpretation of results, and preparation of the manuscript.

Chapter 5: We demonstrate that the previously reported similarity reduction methods applied to MHC-II data sets may not eliminate highly similar peptides, i.e., peptides that share $> 80\%$ sequence identity still pass the similarity test. We propose a two-step similarity reduction procedure that is much more stringent than those currently in use for similarity reduction with MHC-II benchmark data sets. We introduce three similarity-reduced MHC-II benchmark data sets derived from MHCPEP (Brusic et al., 1998), MHCBN (Bhasin et al., 2003), and IEDB (Peters et al., 2005) databases and utilize them in our experiments to show that the performance of three MHC-II binding peptide prediction methods estimated using data sets of unique peptides with that obtained using their similarity-reduced counterparts show that the former can be rather optimistic relative to the performance of the same methods on similarity-reduced counterparts of the same data sets. Furthermore, our results demonstrate that conclusions regarding the superiority of one method over another drawn on the basis of performance estimates obtained using commonly used data sets of unique peptides are often contradicted by the observed performance of the methods on the similarity-reduced versions of the same data sets. These results underscore the importance of using similarity-reduced data sets in rigorously comparing the performance of alternative MHC-II peptide prediction methods. The results have been published in PLoS ONE Journal (EL-Manzalawy et al., 2008a). Yasser EL-Manzalawy conceived the study, implemented the algorithms, analyzed the results, and prepared the manuscript. Drena Dobbs and Vasant Honavar contributed to the design of experiments, interpretation of results, and preparation of the manuscript.

Chapter 6: We formulate the problems of qualitatively and quantitatively predicting flexible length MHC-II peptides as multiple instance learning and multiple instance regression problems, respectively. Based on this formulation, we introduce MHCMIR, a novel method for predicting MHC-II binding affinity using multiple instance regression. We present results of experiments using a benchmark data set covering 13 HLA-DR and three H2-IA alleles that show that MHCMIR is competitive with the state-of-the-art methods for predicting MHC-II binding peptides. Yasser EL-Manzalawy conceived the study, implemented the algorithms, analyzed the results, and prepared the manuscript. Drena Dobbs and Vasant Honavar contributed to the design of experiments, and interpretation of results, and preparation of the manuscript.

Chapter 7: We conclude with a summary of the dissertation, the contributions, and future work.

CHAPTER 2. PREDICTING LINEAR B-CELL EPITOPES USING STRING KERNELS

A paper published in the Journal of Moleculer Recognition

Yasser EL-Manzalawy, Drena Dobbs, Vasant Honavar

Abstract

The identification and characterization of B-cell epitopes play an important role in vaccine design, immunodiagnostic tests, and antibody production. Therefore, computational tools for reliably predicting linear B-cell epitopes are highly desirable. We evaluated Support Vector Machine (SVM) classifiers trained utilizing five different kernel methods using 5-fold cross validation on a *homology-reduced* data set of 701 linear B-cell epitopes, extracted from Bcipep database, and 701 non-epitopes, randomly extracted from SwissProt sequences. Based on the results of our computational experiments, we propose BCPred, a novel method for predicting linear B-cell epitopes using the subsequence kernel. We show that the predictive performance of BCPred (AUC = 0.758) outperforms 11 SVM-based classifiers developed and evaluated in our experiments as well as our implementation of AAP (AUC = 0.7), a recently proposed method for predicting linear B-cell epitopes using amino acid pair antigenicity. Furthermore, we compared BCPred with AAP and ABCPred, a method that uses recurrent neural networks, using two data sets of *unique* B-cell epitopes that had been previously used to evaluate ABCPred. Analysis of the data sets used and the results of this comparison show that conclusions about the relative performance of different B-cell epitope prediction methods drawn on the basis of experiments using data sets of *unique* B-cell epitopes are likely to yield overly

optimistic estimates of performance of evaluated methods. This argues for the use of carefully *homology-reduced* data sets in comparing B-cell epitope prediction methods to avoid misleading conclusions about how different methods compare to each other. Our *homology-reduced* data set and implementations of BCPred as well as the APP method are publicly available through our web-based server, BCPREDS, at: `http://ailab.cs.iastate.edu/bcpreds/`.

Introduction

B-cell epitopes are antigenic determinants that are recognized and bound by receptors (membrane-bound antibodies) on the surface of B lymphocytes (Pier et al., 2004). There are many different types of B-cell receptors, but each B-cell produces only one type. When a B-cell receptor binds its cognate antigen, the B-cell is stimulated to undergo proliferation. This involves the generation of two types of cells, effector or plasma B-cells, which produce and secrete soluble antibodies, and memory B-cells, which remain in the organism and can proliferate rapidly if re-exposed to antigen. Hence, understanding the sequence and structural features of B-cell epitopes is critical both for the design of effective vaccines and for the development of sensitive diagnostic tests.

B-cell epitopes can be classified into two types: linear (continuous) epitopes and conformational (discontinuous) epitopes. Linear epitopes are short peptides, corresponding to a contiguous amino acid sequence fragment of a protein (Barlow et al., 1986; Langeveld et al., 2001). In contrast, conformational epitopes are composed of amino acids that are not contiguous in primary sequence, but are brought into close proximity within the folded protein structure. Although it is believed that a large majority of B-cell epitopes are discontinuous (Walter, 1986), experimental epitope identification has focused primarily on linear B-cell epitopes (Flower, 2007). Even in the case of linear B-cell epitopes, however, antibody-antigen interactions are often conformation-dependent. The conformation-dependent aspect of antibody binding complicates the problem of B-cell epitope prediction, making it less tractable than T-cell epitope prediction. Therefore, the development of reliable computational methods for predicting linear B-cell epitopes is an important challenge in bioinformatics and computational biology (Greenbaum et al., 2007).

Several studies have reported correlations between certain physicochemical properties of amino acids and the locations of linear B-cell epitopes within protein sequences (Pellequer et al., 1991; Parker and Guo, 1986; Karplus and Schulz, 1985; Emini et al., 1985; Pellequer et al., 1993), and several epitope prediction methods based on physicochemical properties of amino acids have been proposed. For example, hydrophilicity, flexibility, turns, or solvent accessibility propensity scales were used in the methods of Parker and Guo (1986), Karplus and Schulz (1985), Pellequer et al. (1993) and Emini et al. (1985), respectively. PREDITOP (Pellequer and Westhof, 1993), PEOPLE (Alix, 1999), BEPITOPE (Odorico and Pellequer, 2003), and BcePred (Saha and Raghava, 2004) predict linear B-cell epitopes based on groups of physicochemical properties instead of a single property.

Recently, Blythe and Flower (2005) performed an exhaustive assessment of 484 amino acid propensity scales, combined with ranges of profile parameters, to examine the correlation between propensity scale-based profiles and the location of linear B-cell epitopes in a set of 50 proteins. They reported that for predicting B-cell epitopes based on amino acid sequence information, even the best combinations of amino acid propensities performed only marginally better than random. They concluded that the reported performance of such methods in the literature is likely to have been overly optimistic, in part due to the small size of the data sets on which the methods had been evaluated.

Motivated by Blythe and Flower (2005) results and the increasing availability of experimentally identified linear B-cell epitopes, several studies have attempted to improve the accuracy of linear B-cell epitope prediction methods using machine learning approaches. BepiPred (Larsen et al., 2006) combines two amino acid propensity scales and a Hidden Markov Model (HMM) trained on linear epitopes to yield a slight improvement in prediction accuracy relative to techniques that rely on analysis of amino acid physicochemical properties. ABCPred (Saha and Raghava, 2006b) uses artificial neural networks for predicting linear B-cell epitopes. Both feed-forward and recurrent neural networks were evaluated on a *non-redundant* data set of 700 B-cell epitopes and 700 non-epitope peptides, using 5-fold cross validation tests. Input sequence windows ranging from 10 to 20 amino acids, were tested and the best performance, 66% accuracy, was obtained using a recurrent neural network trained on peptides 16 amino acids in length. In the method of Söllner and Mayer (2006), each epitope is represented using a set of 1487 features extracted from a variety of propensity scales, neighborhood matrices, and respective probability and likelihood values. Of two machine learning methods tested, decision trees and a nearest-neighbor method combined with feature selection, the latter was reported to attain an accuracy of 72% on a data set of 1211 B-cell epitopes and 1211 non-epitopes, using a 5-fold cross validation test (Söllner and Mayer, 2006). Chen et al. (2007) observed that certain amino acid pairs (AAPs) tend to occur more frequently in B-cell epitopes than in non-epitope peptides. Using an AAP propensity scale based on this observation, in combination with a support vector machine (SVM) classifier, they reported prediction accuracy of 71% on a data set of 872 B-cell epitopes and 872 non-B-cell epitopes, estimated using 5-fold cross validation. In addition, Chen et al. (2007) demonstrated an improvement in the prediction accuracy, 72.5%, when the APP propensity scale is combined with turns, accessibility, antigenicity, hydrophilicity, and flexibility propensity scales.

In this report, we present BCPred, a method for predicting linear B-cell epitopes using an SVM machine learning method. Although the performance of SVM-based classifiers largely depends on the selection of the kernel function, there are no theoretical foundations for choosing good kernel functions in a data-dependent way. Therefore, one objective of this study was to explore a class of kernel methods, namely string kernels, in addition to the widely used radial bias function (RBF) kernel. Our choice of string kernels was motivated by their successful application in a number of bioinformatics classification tasks, including protein remote homology detection (Leslie et al., 2002, 2004; Zaki et al., 2005), protein structure prediction (Rangwala et al., 2006), protein binding site prediction (Wu et al., 2006), and major histocompatibility complex (MHC) binding peptide prediction (Salomon and Flower, 2006). In addition, we introduce the subsequence kernel (SSK), which has been successfully used in text classification (Lodhi et al., 2002), but has been under-explored in macromolecular sequence classification applications. Our empirical results demonstrate superior performance of SSK over other string kernels and the RBF kernel. Hence, we employed the SSK in building SVM classifiers for our proposed linear B-cell epitope prediction method, BCPred.

A second goal of this study was to determine how existing methods for linear B-cell epitope prediction compare with each other and with BCPred. At present, little is known about the relative performance of different methods, due to the lack of published direct comparisons using standard benchmark data sets. Unfortunately, neither the data set used by Söllner and Mayer (2006) nor the code used for generating and selecting the features used to represent epitope peptides as input to the classifiers is publicly available. The code for the AAP method (Chen et al., 2007) is also not publicly available, however, in contrast to the other methods, it is relatively straightforward to implement. Fortunately, although the code used to train the neural network classifier used in ABCPred is not publicly available, Saha and Raghava (2006b) have made available the data set used for developing and evaluating the ABCPred server, as well as a blind test set (Saha and Raghava, 2006a). Thus, although we are unable to include direct comparisons with results of Söllner and Mayer (2006), in this paper we report direct comparisons of the ABCPred method (Saha and Raghava, 2006b), our implementation of the AAP method of Chen et al. (2007), and our proposed BCPred method, using the ABCPred data sets made publicly available by Saha and Raghava (2006a).

Methods

Data sets

Homology-reduced data sets

Bcipep database (Saha et al., 2005) contains 1230 unique linear B-cell epitopes. We retrieved a set of 947 unique epitopes with each epitope satisfying one of the following two conditions: i) the epitope length is at least 20 amino acids; or ii) the epitope is less than 20 amino acids in length and the accession number of the source antigen is provided.

A set of 20-mer peptides was derived from the 947 unique epitopes by: i) truncating epitopes longer than 20 residues by removing amino acids from both ends to yield a 20-mer from the middle, and ii) extending epitopes shorter than 20 residues by adding amino acids on both ends, based on the corresponding complete antigen sequences retrieved from SwissProt (Bairoch and Apweiler, 2000). Because the resulting data set of 947 20-mer peptides was no longer *non-redundant*, we removed duplicated and highly homologous peptides by filtering the data set based on an 80% sequence identity cutoff using the CD-HIT program (Li et al., 2002) to obtain a *homology-reduced* data set of 701 peptides (positive instances of B-cell epitopes). 701 non-epitope peptides were generated by randomly extracting 20-mer peptides from sequences in SwissProt database (Bairoch and Apweiler, 2000) while ensuring that none of the negative instances so obtained also occur in the positive instances.

Because there is no evidence that 20 amino acids is the optimal length for B-cell epitopes, we decided to experiment with different epitope lengths. Variants of the 20-mer data set were generated by repeating the above procedure for peptide lengths of 18, 16, 14, or 12 residues. For the sake of brevity, we will refer to these data sets by BCP*nn*, where *nn* is a two digit number representing the length of the peptides in the data set (e.g., BCP16 refers to the *homology-reduced* data set where each peptide is composed of 16 residues).

It should be noted that deriving a data set of shorter peptides from the 20-mer data set by trimming amino acids from both termini of each peptide is not guaranteed to produce a data set with <80% sequence identity because such trimming could increase the similarity between two peptides in the data set. Therefore, to ensure that the resulting data sets are *homology-reduced*, we re-applied the 80% sequence identity cutoff filter in generating each data set of epitopes less than 20 residues in length. The resulting *homology-reduced* data sets (BCP20, BCP18, BCP16, BCP14, and BCP12) are available at `http://ailab.cs.iastate.edu/bcpreds/`.

ABCPred data set

Saha and Raghava (2006a) have made available the data sets used to train and evaluate ABCPred. Because the best reported performance of ABCPred was obtained using a 16-mer peptide data set (ABCP16), we chose this data set for directly comparing ABCPred with BCPred and AAP (Chen et al., 2007) using 5-fold cross validation.

Blind test set

Saha and Raghava (2006a) have made available a blind test set comprising 187 epitopes, none of which were used in training the ABCPred method, and a set of 200 16-mer non-epitope peptides extracted from the non-allergen data set of Björklund et al. (2005). B-cell epitopes less than 16 amino acids in length were extended to 16-mer peptides by adding an equal number of residues to both ends based on the protein sequence of the source antigen. In the remaining text, we will use the abbreviation (SBT16) to refer to Saha 16-mer blind test set.

Support vector machines and kernel methods

Support vector machines (SVMs) (Vapnik, 2000) are a class of supervised machine learning methods used for classification and regression. Given a set of labeled training data (x_i, y_i), where $x_i \in R^d$ and $y_i \in \{+1, -1\}$, training an SVM classifier involves finding a hyperplane that maximizes the geometric margin between positive and negative training data samples. The hyperplane is described as $f(x) = \langle w, x \rangle + b$, where w is a normal vector and b is a bias term. A test instance, x, is assigned a positive label if $f(x) > 0$, and a negative label otherwise. When the training data are not linearly separable, a kernel function is used to map nonlinearly separable data from the input space into a feature space. Given any two data samples x_i and x_j in an input space $X \in R^d$, the kernel function K returns $K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ where Φ is a nonlinear map from the input space X to the corresponding feature space. The kernel function K has the property that $K(x_i, x_j)$ can be computed without explicitly mapping x_i and x_j into the feature space, but instead, using their dot product $\langle x_i, x_j \rangle$ in the input space. Therefore, the kernel trick allows us to train a linear classifier, e.g., SVM, in a high-dimensional feature space where the data are assumed to be linearly separable without explicitly mapping each training example from the input space into the feature space. This approach relies implicitly on the selection of a feature space in which the training data are likely to be linearly separable (or nearly so) and explicitly on the selection of the kernel function to achieve such separability. Unfortunately, there is no single kernel that is guaranteed to perform well on every data set. Consequently, the SVM approach requires some care in selecting a suitable kernel and tuning the kernel parameters (if any).

String kernels

String kernels (Leslie et al., 2002, 2004; Lodhi et al., 2002; Saigo et al., 2004; Haussler, 1999) are a class of kernel methods that have been successfully used in many sequence classification tasks (Leslie et al., 2002, 2004; Saigo et al., 2004; Zaki et al., 2005; Rangwala et al., 2006; Wu et al., 2006). In these applications, a protein sequence is viewed as a string defined on a finite alphabet of 20 amino acids. In this work, we explore four string kernels: spectrum (Leslie et al., 2002), mismatch (Leslie et al., 2004), local alignment (Saigo et al., 2004), and subsequence (Lodhi et al., 2002), in predicting linear B-cell epitopes. The subsequence kernel (Lodhi et al., 2002) has proven useful in text classification (Lodhi et al., 2002) and natural language processing (Clark et al., 2006). However, to the best of our knowledge, this kernel has not been previously explored in the context of macromolecular sequence classification problems. A brief description of the four kernels follows.

Spectrum kernel

Let A denote a finite alphabet, e.g., 20 amino acids. x and y denote two strings defined on the alphabet A. For $k \geq 1$, the k-spectrum is defined as (Leslie et al., 2002):

$$\Phi_k = (\phi_\alpha(x))_{\alpha \in A^k} \tag{1}$$

where ϕ_α is the number of occurrences of the k-length substring α in the sequence x. The k-spectrum kernel of the two sequences x and y is obtained by taking the dot product of the corresponding k spectra:

$$K_k^{spct}(x,y) = \langle \Phi_k(x), \Phi_k(y) \rangle \tag{2}$$

Intuitively, this kernel captures a simple notion of string similarity: two strings are deemed similar (i.e., have a high k-spectrum kernel value) if they share many of the same k-length substrings.

Mismatch kernel

The mismatch kernel (Leslie et al., 2004) is a variant of the spectrum kernel in which inexact matching is allowed. Specifically, the (k, m)-mismatch kernel allows up to $m \leq k$ mismatches to occur when comparing two k-length substrings. Let α be a k-length substring, the (k, m)-mismatch feature map is defined on α as:

$$\Phi_{(k,m)}(\alpha) = (\phi_\beta(\alpha))_{\beta \in A^k} \tag{3}$$

where $\phi_\beta(\alpha) = 1$ if $\beta \in N_{(k,m)(\alpha)}$, where β is the set of k-mer substrings that differs from α by at most m mismatches. Then, the feature map of an input sequence x is the sum of the feature vectors for k-mer substrings in x:

$$\Phi_{(k,m)}(x) = \sum_{k-mers\,\alpha\,in\,x} \Phi_{(k,m)}(\alpha) \tag{4}$$

The (k, m)-mismatch kernel is defined as the dot product of the corresponding feature maps in the feature space:

$$K^{msmtch}_{(k,m)}(x, y) = \langle \Phi_{(k,m)}(x), \Phi_{(k,m)}(y) \rangle \tag{5}$$

It should be noted that the $(k, 0)$-mismatch kernel results in a feature space that is identical to that of the k-spectrum kernel. An efficient data structure for computing the spectrum and mismatch kernels in $O(|x| + |y|)$ and $O(k^{m+1}|A|^m(|x| + |y|))$, respectively, is provided in (Leslie et al., 2004).

Local alignment kernel

Local alignment (LA) kernel (Saigo et al., 2004) is a string kernel adapted for biological sequences. The LA kernel measures the similarity between two sequences by summing up scores obtained from gapped local alignments of the sequences. This kernel has several parameters:

the gap opening and extension penalty parameters, d and e, the amino acid mutation matrix s, and the factor β, which controls the influence of suboptimal alignments on the kernel value. Detailed formulation of the LA kernel and a dynamic programming implementation of the kernel with running time complexity in $O(|x||y|)$ are provided in (Saigo et al., 2004).

Subsequence kernel

The subsequence kernel (Lodhi et al., 2002) generalizes the k-spectrum kernel by considering a feature space generated by the set of all (contiguous and non-contiguous) k-mer subsequences. For example, if we consider the two strings "act'' and "$acctct''$, the value returned by the spectrum kernel with $k = 3$ is 0. On the other hand, the $(3, 1)$-mismatch kernel will return 3 because the 3-mer substrings "acc'', "cct'', and "tct'' have at most one mismatch when compared with "act''. The subsequence kernel considers the set ("$ac - t''$, "$a - ct''$, "$ac - - - t''$, "$a - c - -t''$, "$a - - - ct''$) of non-contiguous substrings and returns a similarity score that is weighted by the length of each non-contiguous substring. Specifically, it uses a decay factor, $\lambda \leq 1$, to penalize non-contiguous substring matches. Therefore, the subsequence kernel with $k = 3$ will return $2\lambda^4 + 3\lambda^6$ when applied to "act'' and "$acctct''$ strings. More precisely, the feature map Φ_k of a string x is given by:

$$\Phi_{(k,\lambda)}(x) = (\sum_{i:u=x[i]} \lambda^{l(i)})_{u \in A^k} \tag{6}$$

where $u = x[i]$ denotes a substring in x where $1 \leq i_1 < \ldots < i_{|u|} \leq |x|$ such that $u_j = s_{ij}$, for $j = 1, \ldots, |u|$ and $l(i) = i_{|u|} - i_1 + 1$ is the length of the subsequence in x. The subsequence kernel for two strings x and y is determined as the dot product of the corresponding feature maps:

$$\begin{aligned} K(x,y)^{sub}_{(k,\lambda)} &= \langle \Phi_{(k,\lambda)}(x), \Phi_{(k,\lambda)}(y) \rangle \\ &= \textstyle\sum_{u \in A^k} \sum_{i:u=x[i]} \lambda^{l(i)} \sum_{j:u=y[j]} \lambda^{l(j)} \\ &= \textstyle\sum_{u \in A^k} \sum_{i:u=x[i]} \sum_{j:u=y[j]} \lambda^{l(j)+l(j)} \end{aligned} \tag{7}$$

This kernel can be computed using a recursive algorithm based on dynamic programming in $O(k|x||y|)$ time and space. The running time and memory requirements can be further reduced using techniques described in (Seewald and Kleedorfer, 2005).

Amino acid pairs propensity scale

Amino acid pairs (AAPs) are obtained by decomposing a protein/peptide sequence into its 2-mer subsequences. Chen et al. (2007) observed that some particular AAPs tend to occur more frequently in B-cell epitopes than in non-epitope peptides. Based on this observation, they developed an AAP propensity scale defined by:

$$\theta(\alpha) = log(\frac{f_\alpha^+}{f_\alpha^-}) \tag{8}$$

where f_α^+ and f_α^- are the occurrence frequencies of AAP α in the epitope and non-epitope peptide sequences, respectively. These frequencies have been derived from Bcipep (Saha et al., 2005) and Swissprot (Bairoch and Apweiler, 2000) databases, respectively. To avoid the dominance of an individual AAP propensity value, the scale in Eq. 8 has been normalized to a $[-1, +1]$ interval through the following conversion:

$$\theta(\alpha) = 2(\frac{\theta(\alpha) - min}{max - min}) - 1 \tag{9}$$

where max and min are the maximum and minimum values of the propensity scale before the normalization.

Chen et al. (2007) explored SVMs using two kernels: a dot product kernel applied to the average of the AAP scale values for all the AAPs in a peptide and an RBF kernel defined in a 400-dimensional feature space as follows:

$$\Phi_{AAP}(x) = (\phi_\alpha(x) \cdot \theta(\alpha))_{\alpha \in A^2} \tag{10}$$

where $\phi_\alpha(x)$ is the number of occurrences of the 2-mer α in the peptide x. The optimal performance was obtained using the RBF kernel and a window of 20 amino acids (Chen et al., 2007).

Five-fold cross validation

In our experiments, we used stratified 5-fold cross validation tests in which the data set is randomly partitioned into five equal subsets such that the relative proportion of epitopes to non-epitopes in each subset is 1:1. Four of the five subsets are used for training the classifier and the fifth subset is used for testing the classifier. This procedure is repeated five times, each time choosing different subsets of the data for training and testing. The estimated performance of the classifier corresponds to an average of the results from the five cross validation runs.

Implementation and SVM parameter optimization

We used Weka (Witten and Frank, 2005) machine learning workbench for implementing the spectrum, mismatch, and LA kernels (RBF and the subsequence kernel are already implemented in Weka). We evaluated the k-spectrum kernel, K_k^{spct}, for $k = 1, 2$, and 3. The (k,m)-mismatch kernel was evaluated at (k,m) equals $(3, 1), (4, 1), (5, 1)$, and $(5, 2)$. The subsequence kernel, $K_{(k,\lambda)}^{sub}$, was evaluated at $k = 2, 3$, and 4 and the default value for λ, 0.5. The LA kernel was evaluated using the BLOSUM62 substitution matrix, gap opening and extension parameters equal to 10 and 1, respectively, and $\beta = 0.5$. For the SVM classifier, we used the Weka implementation of the SMO (Platt, 1998) algorithm. For the string kernels, the default value of the C parameter, $C = 1$, was used for the SMO classifier. For methods that uses the RBF kernel, we found that tuning the SMO cost parameter C and the RBF kernel parameter γ is necessary to obtain satisfactory performance. We tuned these parameters using a 2-dimensional grid search over the range $C = 2^{-5}, 2^{-3}, \ldots, 2^3, \gamma = 2^{-15}, 2^{-13}, \ldots, 2^3$. *It should be noted that the parameter optimization was performed using only the training data.*

Performance evaluation

The prediction Accuracy (ACC), Sensitivity (S_n), Specificity (S_p), and Correlation coefficient (CC) are often used to evaluate prediction algorithms (Baldi et al., 2000). The CC

measure has a value in the range from -1 to +1, and the closer the value to +1, the better the predictor. ACC, S_n, S_p, and CC are defined as follows:

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{11}$$

$$S_n = \frac{TP}{TP + FN} \ and \ S_p = \frac{TN}{TN + FP} \tag{12}$$

$$CC = \frac{TP \times TN - FP \times FN}{\sqrt{(TN + FN)(TN + FP)(TP + FN)(TP + FP)}} \tag{13}$$

where TP, FP, TN, FN are the numbers of true positives, false positives, true negatives, and false negatives respectively.

Although these metrics are widely used to assess the performance of machine learning methods, they all suffer from the important limitation of being threshold-dependent. Thershold-dependent metrics describe the classifier performance at a specific threshold value. It is often possible to increase the number of true positives (equivalently, sensitivity) of the classifier at the expense of an increase in false positives (equivalently, false alarm rate). The Receiver Operating Characteristic (ROC) curve describes the performance of the classifier over all possible thresholds. The ROC curve is obtained by plotting the true positive rate as a function of the false positive rate or, equivalently, sensitivity versus (1-specificity) as the discrimination threshold of the binary classifier is varied. Each point on the ROC curve describes the classifier at a certain threshold value, i.e., at a particular choice of tradeoff between true positive rate and false positive rate. The area under ROC curve (AUC) is a useful summary statistic for comparing two ROC curves. AUC is defined as the probability that a randomly chosen positive example will be ranked higher than a randomly chosen negative example. An ideal classifier will have an AUC = 1, while a classifier assigning labels at random will have an AUC = 0.5, any classifier performing better than random will have an AUC value that lies between these two extremes.

Results

SVM using the subsequence kernel outperforms other kernel methods and the AAP method

In the first set of experiments, we used our *homology-reduced* data sets to evaluate SVMs trained using the spectrum kernel at $k = 1, 2$, and 3, the (k,m)-mismatch kernel at $(k, m) = (3, 1), (4, 1), (5, 1)$, and $(5, 2)$, the LA kernel, and the subsequence kernel at $k = 2, 3$, and 4. We compared the performance of the four string kernels to that of the RBF kernel trained using a binary representation of the data in which each amino acid is represented by a 20-bit binary string. In addition, we evaluated our implementation of the AAP method (Chen et al., 2007) on our data sets. For all methods, the performance was evaluated using 5-fold cross validation. Because it is not feasible to include the complete set of results in this paper, we report only the results on the 20-mer peptides data set, BCP20, and provide the results on data sets BCP18, BCP16, BCP14, and BCP12 in the Supplementary Materials[1].

Table 1 compares the performance of different kernel-based SVM classifiers on BCP20 data set. The subsequence kernel has the best overall performance, in terms of AUC. The (5,1)-mismatch kernel performs slightly better than the k-spectrum kernel, and the performance of k-spectrum kernel with $k = 1$ and $k = 3$ is much better than its performance with $k = 2$. The performance of both the k-spectrum and (k,m)-mismatch kernels appears to be very sensitive to the choice of k and m parameters, because for some choices of k and m, the classifier performance deteriorates to that expected for random assignment of labels to test instances. In contrast, the performance of the subsequence kernel appears to be much less sensitive to the choice of parameter k.

Our implementation of the AAP method (Chen et al., 2007) has the second best overall performance and demonstrates the highest specificity. The LA kernel is very competitive in performance with AAP. Interestingly, the AAP significantly outperforms the RBF kernel trained using data in its binary representation. The AAP method is essentially an RBF kernel trained on the same data but using a different representation, in which each peptide is

[1]supplementary materials are provided in APPENDIX A

represented by a vector of 400 numeric values computed based on the AAP propensity scale. The significant difference observed in performance of these two RBF-based methods highlights the importance of the data representation in kernel methods. All of these observations hold not only for the BCP20 data set but also for the *homology-reduced* data sets of peptides with different lengths (see Supplementary Materials). Most of the methods have their best performance on BCP20 data set and show slight decreases in performance on data sets with decreasing peptide length.

Figure 1 shows the ROC curves for all methods evaluated in this experiment. The ROC curve for the subsequence kernel, $K^{sub}_{(4,0.5)}$, dominates the other ROC curves over a broad range of choices for the tradeoff between true positive and false positive rates. For any user-selected threshold corresponding to specificity in the range 100% to 20%, $K^{sub}_{(4,0.5)}$ has the best corresponding sensitivity. We conclude that BCPred, SVM-based classifier trained using the subsequence kernel $K^{sub}_{(4,0.5)}$, outperforms all other methods tested in predicting linear B-cell epitopes.

Table 1 Performance of different methods on our BCP20 *homology-reduced* data set using 5-fold cross validation. Best results are highlighted in bold. BCPred method denotes $K^{sub}_{(4,0.5)}$.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
K^{spct}_1	62.62	60.63	64.62	0.253	0.681
K^{spct}_2	58.56	59.49	57.63	0.171	0.614
K^{spct}_3	64.12	63.2	65.05	0.283	0.660
$K^{msmtch}_{(3,1)}$	48.86	50.5	47.22	-0.023	0.468
$K^{msmtch}_{(4,1)}$	55.35	54.64	56.06	0.107	0.593
$K^{msmtch}_{(5,1)}$	64.91	62.05	67.76	0.299	0.683
$K^{msmtch}_{(5,2)}$	55.85	55.35	56.35	0.117	0.584
LA	64.76	61.63	67.9	0.296	0.696
$K^{sub}_{(2,0.5)}$	62.62	62.34	62.91	0.253	0.664
$K^{sub}_{(3,0.5)}$	65.83	67.48	64.19	0.317	0.722
BCPred	**67.90**	**72.61**	63.2	**0.360**	**0.758**
RBF	57.28	57.49	57.06	0.146	0.617
AAP	64.05	52.92	**75.18**	0.288	0.700

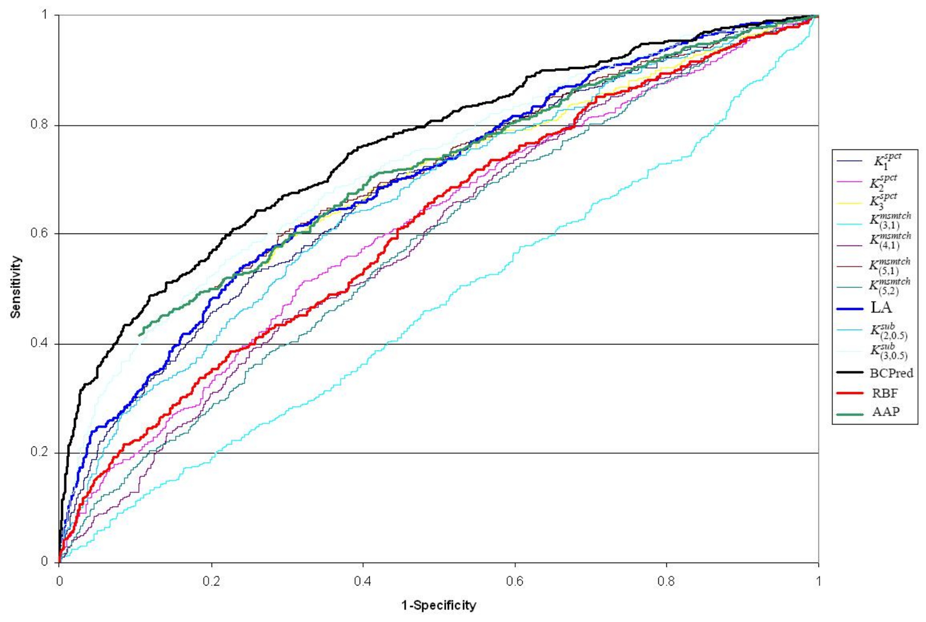


Figure 1 ROC curves for different methods on BCP20 *homology-reduced* data set. BCPred method denotes $K^{sub}_{(4,0.5)}$. BCPred ROC curve dominates all other ROC curves for any user-selected threshold corresponding to specificity in the range 100% to 20%.

Statistical analysis

We summarize statistical analysis of the results and conclusions presented in the preceding subsection. Specifically, we attempt to answer, from a statistical prospective, the following questions: Is the performance of BCPred significantly different from those of other methods? Or more generally, how do the different B-cell epitope prediction methods compare with each other?

To answer these questions, we utilized multiple hypothesis comparisons (Friedman, 1940; Fisher, 1973) for comparing a set of classifiers on multiple data sets. We chose to use the AUC as the performance metric in these tests. Table 2 shows the AUC values of 13 classifiers on the five *homology-reduced* data sets.

One approach for performing multiple hypothesis comparisons over the results in Table 2, is to perform paired t-tests between each pair of classifiers at p-value equals 0.05. However, when the number of classifiers being compared is large compared to the number of datasets, paired

Table 2 AUC values for different methods evaluated on *homology-reduced* data sets. For each data set, the rank of each classifier is shown in parentheses.

Method	BCP20	BCP18	BCP16	BCP14	BCP12	Avg
K_1^{spct}	0.681(6)	0.588(11)	0.652(7)	0.582(11)	0.591(10)	0.619(9)
K_2^{spct}	0.614(10)	0.636(8)	0.612(9)	0.597(9)	0.606(8)	0.613(8.8)
K_3^{spct}	0.660(8)	0.675(6)	0.645(8)	0.675(3)	0.636(6)	0.658(6.2)
$K_{(3,1)}^{msmtch}$	0.468(13)	0.465(13)	0.460(13)	0.506(13)	0.450(13)	0.470(13)
$K_{(4,1)}^{msmtch}$	0.593(11)	0.599(10)	0.569(11)	0.596(10)	0.548(11)	0.581(10.6)
$K_{(5,1)}^{msmtch}$	0.683(5)	0.691(4.5)	0.667(6)	0.649(6)	0.594(9)	0.657(6.1)
$K_{(5,2)}^{msmtch}$	0.584(12)	0.568(12)	0.563(12)	0.574(12)	0.535(12)	0.565(12)
LA	0.696(4)	0.691(4.5)	0.686(4)	0.671(4)	0.662(4)	0.681(4.1)
$K_{(2,0.5)}^{sub}$	0.664(7)	0.668(7)	0.681(5)	0.647(7)	0.643(5)	0.661(6.2)
$K_{(3,0.5)}^{sub}$	0.722(2)	0.726(2)	0.718(2)	0.697(2)	0.687(2)	0.710(2)
BCPred	0.758(1)	0.751(1)	0.730(1)	0.733(1)	0.709(1)	0.736(1)
RBF	0.617(9)	0.601(9)	0.594(10)	0.603(8)	0.620(7)	0.607(8.6)
AAP	0.700(3)	0.699(3)	0.689(3)	0.665(5)	0.663(3)	0.683(3.4)

t-tests are susceptible to type I error, i.e., falsely concluding that the two methods significantly differ from each other in terms of performance when in fact they do not. To reduce the chance of type I errors, we used Bonferroni adjustments (Neter et al., 1985) in performing multiple comparisons. Specifically, two classifiers are considered different at 0.05 significance level, if the null hypothesis (that they are not different) is rejected by a paired t-test at $0.05/12 = 0.0042$ confidence level (12 denotes the number of comparisons). Table 3 summarizes the results of Bonferroni-corrected tests comparing the performance of the classifiers. Significantly different pairs of classifiers are indicated with a $\times$. The results in Table 3 show that the reported performance of BCPred is significantly different from the performance of other classifiers. On the other hand, the differences between the performance of K_3^{spct}, $K_{(5,1)}^{msmtch}$, LA, and $K_{(3,0.5)}^{sub}$ classifiers and the performance of AAP are not statistically significant.

A second approach for performing multiple hypothesis comparisons over the results in Table 2 is to use non-parametric tests. Demšar (2006) has suggested that non-parametric tests should be preferred over parametric tests for comparing machine learning algorithms because the non-parametric tests, unlike parametric tests, do not assume normal distribution of the

Table 3 Results of Bonferroni adjustments using p-value = 0.0042. "×" indicates that the corresponding pair of methods is significantly different.

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12
K_1^{spct} (M1)												
K_2^{spct} (M2)	0											
K_3^{spct} (M3)	0	0										
$K_{(3,1)}^{msmtch}$ (M4)	0	×	×									
$K_{(4,1)}^{msmtch}$ (M5)	0	×	×	×								
$K_{(5,1)}^{msmtch}$ (M6)	0	0	0	×	×							
$K_{(5,2)}^{msmtch}$ (M7)	0	×	×	×	×	×						
LA (M8)	×	×	0	×	×	0	×					
$K_{(2,0.5)}^{sub}$ (M9)	0	×	0	×	×	0	×	×				
$K_{(3,0.5)}^{sub}$ (M10)	×	×	×	×	×	×	×	×	×			
BCPred (M11)	×	×	×	×	×	×	×	×	×	×		
RBF (M12)	0	0	0	×	0	0	×	×	0	×	×	
AAP (M13)	×	×	0	×	×	0	×	0	×	0	×	×

samples (e.g., the data sets). Demšar suggested a three-step procedure for performing multiple hypothesis comparisons using non-parametric tests. First, the classifiers being compared are ranked on the basis of their observed performance on each data set (see Table 2). Second, Friedman test is applied to determine whether the measured average ranks are significantly different from the mean rank under the null hypothesis. Third, if the null hypothesis can be rejected at 0.05 significance level, the Nemenyi test is used to determine whether significant differences exist between any given pair of classifiers. Unfortunately, this procedure requires the number of data sets to be greater than 10 and the number of methods to be greater than 5 (Demšar, 2006). Because we have 13 classifiers to compare and only 5 data sets, we cannot use this procedure. However, as noted by Demšar (2006), the average ranks by themselves provide a reasonably fair comparison of classifiers. Hence, we use average ranks to compare BCPred with the other methods. As shown in Table 2, BCPred and $K_{(3,0.5)}^{sub}$ have average ranks 1 and 2, respectively, followed by AAP and LA kernel with average ranks 3.4 and 4.1.

In summary, the results reported in Table 2 along with the statistical analysis of the results lend support to the conclusion summarized in the preceding subsection that the performance of BCPred is superior to that of the other 12 methods.

Effect of epitope length on BCPred performance

Our choice of an epitope length of 20 in the experiments summarized above was motivated by the previous work (Saha and Raghava, 2006b; Chen et al., 2007). Figure 2 shows the distribution of unique epitope lengths in Bcipep database (Saha et al., 2005). The Bcipep database contains 1230 unique B-cell epitopes with 99.4% of the epitopes have lengths ranging from 3 to 38 amino acids. It turns out that 86.7% of the unique B-cell epitopes are at most 20 amino acids in length. However, it is natural to ask as to how the performance of BCPred varies with the choice of epitope length. We now proceed to examine the effect of epitope length on the performance of BCPred.

In order to study the effect of epitope length we compared the performance of BCPred and other methods trained and tested on data sets with epitope lengths of 20, 18, 16, 14, and 12. Our results show that BCPred and five other methods reach their best performance (in terms of AUC) on data set BCP20 (corresponding to epitope length of 20) (see Table 2). This observation raises an obvious question: Can we improve the predictive performance of BCPred if we increase the epitope length to beyond 20? To explore this question, we generated five additional *homology-reduced* data sets, BCP22, BCP24, BCB26, BCP28, and BCP30 (corresponding to epitope lengths of 22, 24, 26, 28, and 30 respectively) and compared the performance of BCPred on the resulting data sets using 5-fold cross validation. The performance of BCPred on the five data sets is summarized in Table 4. It is interesting to note that the measured AUC on BCP22 is 0.788 compared to 0.758 on BCP20. A slightly better AUC, 0.804, was observed on BCP28.

Why does BCPred have a better performance on BCP28 compared with its performance on BCP20? There are at least three possible explanations: i) BCP28 includes longer segments of epitope sequences of length greater than 20 amino acids in the data set than BCP20; ii) Some of the hard-to-predict epitopes in BCP20 are eliminated from BCP28 because these epitopes are located very close to the ends of the antigen sequence and so extending these epitopes to 28 amino acids in length by adding an equal number of amino acids from both ends is not possible; iii) amino acid neighbors of the epitopes carry some useful signal that helps the classifier to better discriminate epitopes from non-epitopes.

To test these hypotheses, we constructed a modified version of BCP20 data set, MBCP20. MBCP20 was derived from BCP28 by trimming 4 amino acids from both ends of each peptide in BCP28. Therefore, BCP28 and MBCP20 can be viewed as two different representations of the same set of epitope/non-epitope data. However, the sequence similarity between any pair of epitopes in BCP28 is guaranteed to be less than 80% but this is not necessarily the case for epitopes in MBCP20. The performance of BCPred on MBCP20 data set is shown in Table 4. The results show that the performance of BCPred on MBCP20, the trimmed version of BCP28, is worse than that on BCP28. This observation provides some evidence against the second of the three possible explanations for observed improvements in performance with epitope length chosen to construct the data sets used to train and test BCPred. It also lends some credence to the suggestion that the amino acid neighbors of the B-cell epitopes may help the classifier to better discriminate between epitopes and non-epitope sequences. As noted earlier, another possibility is that increasing the length results in covering a larger fraction of epitope sequence in the data set in the case of epitope sequences that are longer than 20 amino acid in length (about 13% of the epitopes).

Table 4 Performance of BCPred on *homology-reduced* data sets containing longer epitopes (22-30 residues) and modified BCP20, MBCP20, data set.

Data set	ACC(%)	Sn(%)	Sp(%)	CC	AUC
BCP20	67.90	72.61	63.2	0.360	0.758
BCP22	70.91	73.20	68.63	0.419	0.788
BCP24	67.70	**79.81**	55.59	0.365	0.783
BCP26	70.66	74.18	67.14	0.414	0.796
BCP28	**72.06**	72.37	71.74	**0.441**	**0.804**
BCP30	70.62	68.95	**72.29**	0.413	0.788
MBCP20	68.45	67.97	68.92	0.369	0.758

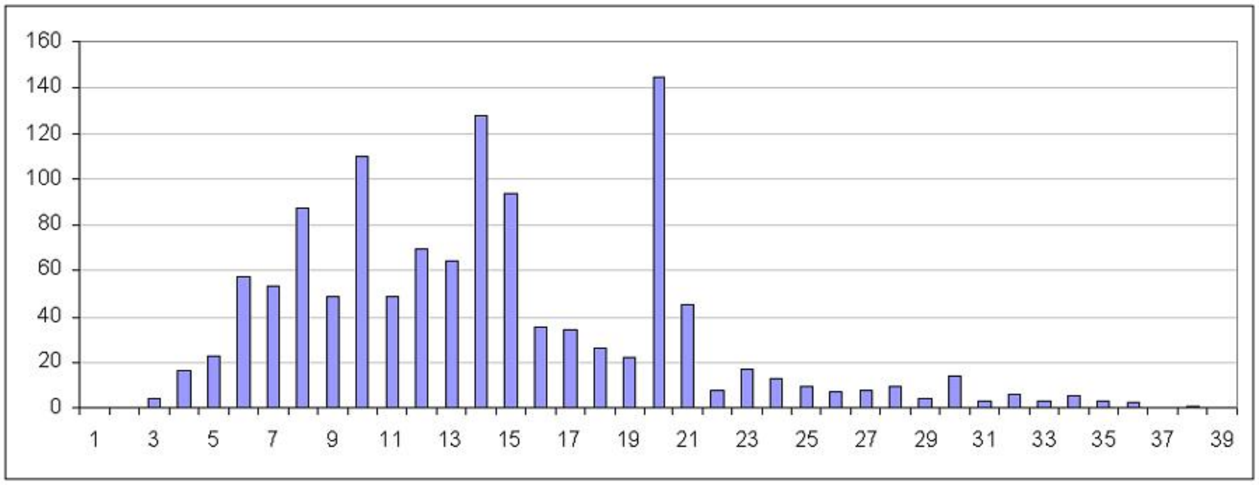


Figure 2 Length distribution of unique linear B-cell epitopes in Bcipep database. 86.7% of the epitopes are at most 20 amino acids in length.

Comparing BCPred with existing linear B-cell epitope prediction methods

Although a number of machine learning based methods for predicting linear B-cell epitopes have been proposed (Saha and Raghava, 2006b; Söllner and Mayer, 2006; Chen et al., 2007), little is known about how these methods directly compare with one another due to the lack of published comparisons using standard benchmark data sets. Unfortunately, because the code and precise parameters used to train several of these methods are not available, we were unable to make direct comparisons of these methods using the *homology-reduced* data sets we used in our first set of experiments (summarized in Table 1 and Table 2). However, we were able to compare BCPred with our implementation of APP and ABCPred, using the publicly available benchmark data sets (Saha and Raghava, 2006a) that were used to evaluate ABCPred. Because the best reported performance of ABCPred was obtained using a data set of 16-mer peptides, comprising 700 epitopes and 700 non-epitopes peptides, we used the same data set, ABCP16, to compare ABCPred with BCPred and AAP. In addition, a blind test set, SBT16, consisting of 187 epitopes and 200 16-mer non-epitopes, also made available by Saha and Raghava (2006a), was used to compare the three methods.

Table 5 compares the performance of BCPred, AAP, and ABCPred on ABCP16 data set (Saha and Raghava, 2006a), using 5-fold cross validation. In terms of overall accuracy, both BCPred and AAP outperformed ABCPred on this data set, with BCPred showing the best performance (74.57%). Interestingly, the performance of BCPred and AAP on ABCP16 data set was better than their performance on the *homology-reduced* data set used in the first set of experiments described above. The performance of the three classifiers trained on ABCP16 data set, but tested on the blind test set SBT16 is summarized in Table 6. In this case, the performance of ABCPred was slightly better than that of BCPred and APP.

Table 5 Performance of BCPred, AAP, and ABCPred evaluated on ABCP16 data set using 5-fold cross validation. "-" denotes unavailable information.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
BCPred	**74.57**	**70.14**	79.00	0.493	**0.801**
AAP	73.14	50.17	**95.57**	**0.518**	0.782
ABCPred	65.93	67.14	64.71	0.319	-

Table 6 Performance comparison of BCPred, AAP, and ABCPred. The three classifiers were trained using ABCP16 data set and evaluated using SBT16 blind test set.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
BCPred	65.89	66.31	**65.50**	**0.318**	**0.699**
AAP	64.60	64.17	65.00	0.292	0.689
ABCPred	**66.41**	**71.66**	61.50	-	-

What explains the discrepancy between the performance estimated on ABCP16 data set and the performance on SBT16 blind test set?

Based on the empirical results summarized above, it is natural to ask: How can we explain the differences in relative performance of BCPred and AAP on our *homology-reduced* data sets versus the performance of these methods on ABCP16 data set (Saha and Raghava, 2006b)? How can we explain the observation that BCPred and AAP outperform ABCPred in 5-fold cross validation experiments using ABCP16 data set but not on the blind test set, SBT16 data set?

Could the observed differences in relative performance be explained by differences in the two data sets, BCP16 and ABCP16? To explore this possibility, we considered the procedures used to create the data sets. Recall that Saha et al. started with a data set of 20-mer peptides (after extending the length of shorter B-cell epitopes based on the corresponding antigen sequences) (Saha and Raghava, 2006b). As noted above, there is a possibility that the resulting data set of 20-mer peptides includes several highly similar peptides (e.g., peptides that differ from each other in only one or two amino acids). More importantly, the 16-mer data set, ABCP16, was derived from the 20-mer data set, ABCP20, by trimming 2 amino acids from the ends of each 20-mer peptide; as a result, two 20-mers that were not duplicates of each other might yield 16-mers that are highly similar after the ends are trimmed off. In summary, the ABCP20 data set reported in Saha el al. (Saha and Raghava, 2006b) was constructed from unique epitopes without applying any homology reduction filters. Moreover, the procedure used by Saha and Raghava (2006b) to derive ABCP16 from ABCP20 can be expected to increase the pair-wise similarity between sequences in ABCP16 relative to the pairwise sequence similarity within ABCP20.

Indeed, when we scanned the positive peptides in ABCP16 data set (Saha and Raghava, 2006a) for duplicate peptides, we found 37 cases in which a 16-mer peptide has at least one exact duplicate in the 16-mer data set and several of these have multiple copies in the 16-mer data set (see Table 7). Consequently, 5-fold cross validation using ABCP16 data set is likely to yield overly optimistic performance estimates, especially for methods that rely on sequence features such as those identified by the subsequence kernel and AAP.

To determine exactly how redundant are the positive peptides in ABCP16 data set, we filtered them using an 80% sequence identity cutoff. We found that applying the 80% sequence identity cutoff resulted in the number of positive peptides in the ABCP16 data being reduced from 696 to 532. Thus, 23.5% of the positive peptides in ABCP16 data set have more than 80% sequence identity. This observation leads us to conclude that the observed differences in the performance of BCPred and AAP on the *homology-reduced* data set (BCP16) relative to that on the ABCP16 data set, as well as the results of comparisons of AAP, ABCPred, and BCPred on the blind test set (SBT16), are explained by the presence of a relatively large number of highly similar peptides in ABCP16 data set.

The preceding analysis highlights an important issue in evaluating linear B-cell prediction tools, which, to our knowledge, has not been addressed in previous studies. Previously published linear B-cell epitope prediction methods (Larsen et al., 2006; Saha and Raghava, 2006b; Söllner and Mayer, 2006; Chen et al., 2007) have been evaluated using data sets of unique epitopes without considering any sequence similarities that may exist among epitopes. In reality, *unique* epitopes may share a high degree of similarity (e.g., a shorter epitope may be included within a longer one, or two epitopes may differ in only one or two amino acids). In this work, we demonstrated that cross validation performance estimated on such data sets can be overly-optimistic. Moreover, such data sets can lead to false conclusions when used to compare different prediction methods. For instance, our comparison of ABCPred, AAP and BCPred using 5-fold cross validation on ABCP16 data set suggested that AAP and BCPred significantly outperform ABCPred. Such a conclusion may not be valid because evaluation of the three methods on a blind test set, SBT16, suggests that the three methods are comparable to each other.

BCPREDS web server

We have implemented BCPREDS, an online web server for B-cell epitope prediction, using classifiers trained on the *homology-reduced* data sets of B-cell epitopes developed in this work. The server can be accessed at `http://ailab.cs.iastate.edu/bcpreds/`.

Because it is often valuable to compare predictions of multiple methods, and consensus predictions are more reliable than individual predictions, the BCPREDS server allows users to choose the method for predicting B-cell epitopes, either BCPred or AAP (and in the future, additional methods). Users provide an antigen sequence and optionally can specify desired epitope length and specificity threshold. Results are returned in several user-friendly formats. In what follows, we illustrate the use of the BCPREDS server in a representative application

Table 7 List of 37 duplicated 16-mer peptides and number of occurrence of each (N) in ABCP16 data set.

Peptide	N	Peptide	N
RGPGRAFVTIGKIGNM	2	RKRIHIGPGRAFYTTK	3
GPQGLAGQRGIVGLPG	1	KSIRIQRGPGRAFVTI	9
QEVGKAMYAPPISGQI	2	SIRIQRGPGRAFVTIG	3
SEGATPQDLNTMLNTV	1	RQGPKEPFRDYVDRFY	1
LGIWGCSGKLICTTAV	1	AKATYEAALKQYEADL	1
GDRADGQPAGDRADGQ	1	NYNKRKRIHIGPGRAF	2
DGVGAASRDLEKHGAI	1	NNNTRKSITKGPGRVI	1
RLIEDNEYTARQGAKF	1	LQARILAVERYLKDQQ	1
EQELLELDKWASLWNW	2	KMQTLWDEIMDINKRK	1
NVTENFDMWKNDMVEQ	2	TRKSIRIQRGPGRAFV	2
NVTENFNMWKNDMVEQ	1	DPNPQEVVLVNVTENF	1
GIWGCSGKLICTTAVP	1	YLKDQQLLGIWGCSGK	1
QLLGIWGCSGKLICTT	1	TRKSITKGPGRVIYAT	1
QVTPGRGPGRAPCSAG	1	SFNISTSIRGKVQKEY	1
IRIQRGPGRAFVTIGK	1	RPVVSTQLLLNGSLAE	1
RIQRGPGRAFVTIGKI	3	RKDPVVYPWMKKVHVN	1
KRIHIGPGRAFYTTKN	2	RNRWEWRPDFESEKVK	1
AGTVGENVPDDLYIKG	1	FLQIYKQGGFLGLSNI	1
KRKRIHIGPGRAFYTT	4		

of B-cell epitope prediction.

Identifying B-cell epitopes in the receptor-binding domain of SARS-CoV spike protein

Since its outbreak in 2002, the development of an effective and safe vaccine against Severe Acute Respiratory Syndrome coronavirus (SARS-CoV) has become an urgent need for preventing future worldwide outbreak of SARS, a life threatening disease (Drosten et al., 2003; Fouchier et al., 2003; Ksiazek et al., 2003; Peiris et al., 2003). Infection by SARS-CoV is initiated by binding of its spike (S) protein to its functional receptor, angiotensin-converting enzyme (ACE2), which is expressed on the surface of host cells (Dimitrov, 2003; Li et al., 2003). The S protein comprises 1255 amino acids and consists of two functional domains: S1 (residues 1-667) and S2 (residues 668-1255) (Wu et al., 2004). The S1 domain is responsible for binding to receptors on target cells (Li et al., 2003) and the S2 domain contributes to the subsequence fusion between viral envelope and cellular membrane (Beniac et al., 2006). In addition, the S2 domain contains two highly conserved heptad repeat (HR) regions, HR1 and HR2 correspond to amino acid residues 915-949 and 1150-1184, respectively (Sainz et al., 2005). Several studies reported that the receptor-binding domain (RBD), residues 318-510, is an attractive target for developing SARS-CoV vaccine because blocking the binding of S1 domain to cellular receptors can prevent envelope fusion and virus entry mediated by the S2 domain (Sui et al., 2004; Prabakaran et al., 2006). Based on these findings, we surveyed the literature to collect previously identified epitopes within the RBD fragment of the SARS-CoV S protein. The collected epitopes are summarized in Table 8. None of these epitopes appears in our training data sets. Because epitope SP3 is included within the epitope (434-467) (GNYNYKYRYLKHGKLRPFERDISNVPFSPDGKPC) reported by Lien et al. (2007), we omitted the longer epitope.

We submitted 193 residues comprising the RBD region of SARS-CoV S protein (residues 318-510 according to accession AAT74874) to the BCPREDS, ABCPred, and Bepipred servers. For BCPREDS, we used the default specificity threshold (75%) and set the epitope length to 16 residues. For the other two servers, we used the default settings. Figure 3 shows the BCPred (top) and AAP (bottom) predictions returned by BCPREDS. Four of the B-cell epitopes predicted by BCPred overlap with epitopes that have been identified in the antigenic regions of RBD of SARS-CoV S protein through experiments. Three of the five epitopes predicted by AAP have substantial overlap with the SP1, SP2, and SP5 epitopes, and the fourth partially overlaps epitopes SP2 and SP3; the fifth does not overlap with any experimentally reported epitopes. In contrast, the ABCPred server, using default parameters, returned 22 predictions covering almost the entire query sequence. Bepipred returned 9 predicted variable-length epitopes, but only three of them are longer than 4 residues in length. Two out of these four epitopes overlap with experimentally reported epitopes. The complete ABCPred and Bepipred predictions are provided in the Supplementary Materials. In evaluating these results, it is worth noting that a high false positive rate is more problematic than an occasional false negative prediction in the B-cell epitope prediction task (Söllner and Mayer, 2006), because a major goal of B-cell epitope prediction tools is to reduce the time and expense of wet lab experiments.

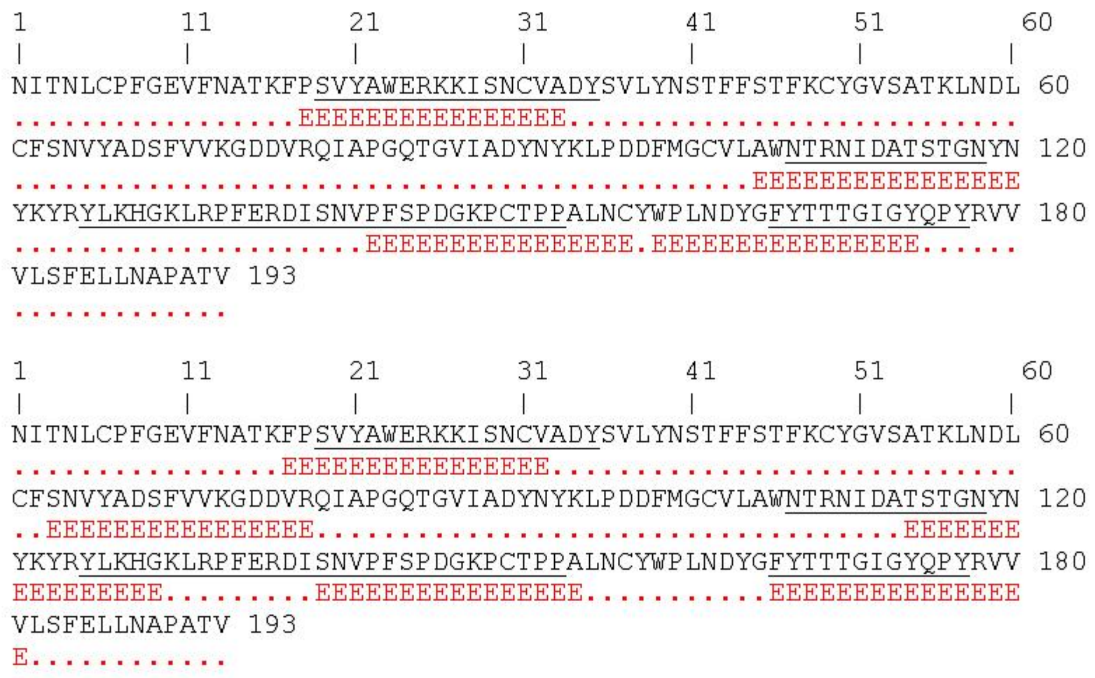
The B-cell epitopes predicted over the entire SARS-CoV S protein using BCPred is given in Figure A.3. Interestingly, the predictions of BCPred over the RBD region are identical regardless of whether the predictions are made over only the RBD sequence fragment or over the entire S protein sequence of SARS-CoV.

Table 8 B-cell epitopes previously identified in the RBD of SARS-CoV S protein.

Epitope	Amino acid sequence	PubMed ID
SP1 (336-352)	SVYAWERKKISNCVADY	16725238
SP2 (424-435)	NTRNIDATSTGN	17055022
SP3 (442-458)	YLKHGKLRPFERDISNV	16725238
SP4 (459-470)	PFSPDGKPCTPP	17055022
SP5 (483-494)	FYTTTGIGYQPY	16337697

Discussion

In this paper, we explored a family of SVM-based machine learning methods for predicting linear B-cell epitopes from primary amino acid sequence. We explored four string kernel methods and compared them to the widely used RBF kernel. Our results demonstrate the usefulness of the four string kernels in predicting linear B-cell epitopes, with the subsequence kernel showing a superior performance over other kernels. In addition, we observed that the subsequence kernel is less sensitive to the choice of the parameter k than the k-spectrum and (k,m)-mismatch kernels. Our experiments using 5-fold cross validation on a *homology-reduced* data set of 701 linear B-cell epitopes and 701 non-epitopes demonstrated that the subsequence kernel significantly outperforms other kernel methods in addition to APP method (Chen et al., 2007). To the best of our knowledge, the subsequence kernel (Lodhi et al., 2002), although previously used in text classification and natural language processing applications, have not been widely exploited in the context of macromolecular sequence classification tasks. The superior performance of the subsequence kernel on B-cell epitope prediction task suggests that it might find use in other related macromolecular sequence classification tasks, e.g., MHC bind-



```
1        11       21       31       41       51       60
|        |        |        |        |        |        |
NITNLCPFGEVFNATKFPSVYAWERKKISNCVADYSVLYNSTFFSTFKCYGVSATKLNDL 60
.................EEEEEEEEEEEEEEEE...........................
CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRNIDATSTGNYN 120
............................................EEEEEEEEEEEEEEEE
YKYRYLKHGKLRPFERDISNVPFSPDGKPCTPPALNCYWPLNDYGFYTTTGIGYQPYRVV 180
.....................EEEEEEEEEEEEEEEE.EEEEEEEEEEEEEEEE......
VLSFELLNAPATV 193
.............

1        11       21       31       41       51       60
|        |        |        |        |        |        |
NITNLCPFGEVFNATKFPSVYAWERKKISNCVADYSVLYNSTFFSTFKCYGVSATKLNDL 60
................EEEEEEEEEEEEEEEE............................
CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRNIDATSTGNYN 120
..EEEEEEEEEEEEEEEE...................................EEEEEEE
YKYRYLKHGKLRPFERDISNVPFSPDGKPCTPPALNCYWPLNDYGFYTTTGIGYQPYRVV 180
EEEEEEEEE.........EEEEEEEEEEEEEEEE...........EEEEEEEEEEEEEEE
VLSFELLNAPATV 193
E............
```

Figure 3 BCPREDS server predictions of epitopes within the RBD of SARS-CoV S protein, made using BCPred (top) and AAP (bottom). Experimentally identified epitopes are underlined. "E" indicates that the corresponding amino acid residue lies in a predicted linear B-cell epitopes.

ing peptide prediction (Salomon and Flower, 2006; Cui et al., 2006b) and protein subcellular localization prediction (Yu et al., 2006a; Bulashevska and Eils, 2006).

One of the challenges for developing reliable linear B-cell epitope predictors is how to deal with the large variability in the length of the epitopes which ranges from 3 to 30 amino acids in length. Many standard machine learning methods require training and testing the classifier using sequences of fixed length. For example, the AAP (Chen et al., 2007) method was evaluated on a data set where the length of the input sequences was fixed to 20 amino acids. Saha and Raghava (2006b) experimented with data sets consisting of peptide sequences of length 20 and shorter, and reported optimal performance of ABCPred classifier on a data set consisting of 16-mer peptides. In BepiPred (Larsen et al., 2006) and propensity scale based methods (Pellequer et al., 1991; Parker and Guo, 1986; Karplus and Schulz, 1985; Emini et al., 1985; Pellequer et al., 1993; Saha and Raghava, 2004), the training examples are windows of five or seven amino acids labeled according to whether the amino acid at the center of the window is included in a linear B-cell epitope or not. Here, we evaluated BCPred on several data sets consisting of fixed length peptides with lengths ranging from 12-30 amino acids with incremental step equals 2. Our results suggest that amino acid neighbors of the B-cell epitope carry some useful information that can help the classifier better discriminate between epitopes and non-epitopes. This is especially interesting in light of the observation that adding a single amino acid to a linear B-cell epitope may affect binding to the antibody.

A similar situation arises in predicting major histocompatibility complex class II (MHC-II) binding peptides. The length of MHC-II binding peptides typically varies from 11 to 30 amino acids in length. Most of the currently available MHC-II binding peptide prediction methods focus on identifying a putative 9-mer binding core region. Therefore, classifiers are trained using 9-mer peptides instead of variable length ones. Recently, two methods (Cui et al., 2006b; Salomon and Flower, 2006) for predicting variable length MHC-II peptides have been proposed. Both methods use the entire sequences of MHC-II binding peptides (as opposed to only the 9-mer cores) for training MHCII binding peptide predictors. The first method (Cui et al., 2006b) maps a variable length peptide into a fixed length feature vector obtained from sequence-derived structural and physicochemical properties of the peptide. The second method (Salomon and Flower, 2006) uses the local alignment (LA) kernel that we used in this study. It would be interesting to apply these methods to the problem of learning to identify variable length linear B-cell epitopes. Our ongoing work aims at exploring the application of string kernels for learning from flexible length linear B-cell epitopes.

In light of the significant room for improvement in performance of B-cell epitope prediction methods reported in the literature, it is important to understand the strengths and limitations of different methods through direct comparisons on standard benchmark data sets. Hence, we compared the BCPred method using the subsequence kernel-based SVM developed in this paper with two published methods: AAP (Chen et al., 2007) and ABCPred (Saha and Raghava, 2006b). In our experiments using the Saha 16-mer peptide data set (containing approximately 700 B-cell epitopes and 700 non-epitope peptides) on which ABCPred had the best reported performance of 66% (Saha and Raghava, 2006b), both BCPred and AAP outperformed ABCPred, based on 5-fold cross validation. However, when the classifiers were tested on a separate *blind* test set instead, no significant difference was observed in their performance. Careful examination of the ABCPred 16-mer data set revealed that the data set has a high degree of sequence redundancy among the epitope peptides, leading to overly optimistic estimates of performance in some cases.

Our demonstration that the only publicly available data set of linear B-cell epitopes (Saha and Raghava, 2006a) is, in fact, highly redundant (with almost 25% of individual 16-mer epitopes having at least one other epitope with >80% sequence identity) is significant. We showed that the redundancy in such a data set can be reflected in overly-optimistic performance estimates, especially for certain types of machine learning classifiers. Consequently, using such a data set can also lead to false conclusions when directly comparing different prediction methods. Therefore, it is very important to evaluate and compare different linear B-cell epitope prediction methods on data sets that are truly *non-redundant* or *homology-reduced* with respect to their constituent epitope sequences, i.e., in which the level of pair-wise sequence identity shared between individual epitopes is known. Towards this goal, we have made our *homology-reduced* data set of linear B-cell epitopes (with <80% sequence identity) publicly available as a benchmarking data set for comparing existing and future linear B-cell epitope prediction methods.

Based on the results of this study, we developed BCPREDS, an online web server for predicting linear B-cell epitopes using either the BCPred method, which implements the subsequence kernel introduced in this paper, or the AAP method of Chen et al. (2007). A case study in which BCPREDS was used to predict linear B-cell epitopes in the RBD of the SARS-CoV S protein demonstrates the potential value of this server in guiding clinical investigations.

Work in progress is aimed at further development and empirical comparisons of different methods for B-cell epitope prediction, in particular, addressing the more challenging problem of predicting discontinuous or conformational B-cell epitopes.

Acknowledgement

We thank anonymous reviewers for their comments and suggestions, Dr. Janez Demšar for discussing the applicability of non-parametric tests, Dr. Douglas Bonett for suggesting the Bonferroni adjustment test. This research was supported in part by a doctoral fellowship from the Egyptian Government to Yasser El-Manzalawy and a grant from the National Institutes of Health (GM066387) to Vasant Honavar and Drena Dobbs.

CHAPTER 3. PREDICTING FLEXIBLE LENGTH LINEAR B-CELL EPITOPES

A paper published in 7th International Conference on Computational Systems Bioinformatics (CSB'08)

Yasser EL-Manzalawy, Drena Dobbs, Vasant Honavar

Abstract

Identifying B-cell epitopes play an important role in vaccine design, immunodiagnostic tests, and antibody production. Therefore, computational tools for reliably predicting B-cell epitopes are highly desirable. We explore two machine learning approaches for predicting flexible length linear B-cell epitopes. The first approach utilizes four sequence kernels for determining a similarity score between any arbitrary pair of variable length sequences. The second approach utilizes four different methods of mapping a variable length sequence into a fixed length feature vector. Based on our empirical comparisons, we propose FBCPred, a novel method for predicting flexible length linear B-cell epitopes using the subsequence kernel. Our results demonstrate that FBCPred significantly outperforms all other classifiers evaluated in this study. An implementation of FBCPred and the datasets used in this study are publicly available through our linear B-cell epitope prediction server, BCPREDS, at: http://ailab.cs.iastate.edu/bcpreds/.

Introduction

B-cell epitopes are antigenic determinants that are recognized and bound by receptors (membrane-bound antibodies) on the surface of B lymphocytes (Pier et al., 2004). The identification and characterization of B-cell epitopes play an important role in vaccine design, immunodiagnostic tests, and antibody production. As identifying B-cell epitopes experimentally is time-consuming and expensive, computational methods for reliably and efficiently predicting B-cell epitopes are highly desirable (Greenbaum et al., 2007).

There are two types of B-cell epitopes: (i) linear (continuous) epitopes which are short peptides corresponding to a contiguous amino acid sequence fragment of a protein (Barlow et al., 1986; Langeveld et al., 2001); (ii) conformational (discontinuous) epitopes which are composed of amino acids that are not contiguous in primary sequence but are brought into close proximity within the folded protein structure. Although it is believed that a large majority of B-cell epitopes are discontinuous (Walter, 1986), experimental epitope identification has focused primarily on linear B-cell epitopes (Flower, 2007). Even in the case of linear B-cell epitopes, however, antibody-antigen interactions are often conformation-dependent. The conformation-dependent aspect of antibody binding complicates the problem of B-cell epitope prediction, making it less tractable than T-cell epitope prediction. Hence, the development of reliable computational methods for predicting linear B-cell epitopes is an important challenge in bioinformatics and computational biology (Greenbaum et al., 2007).

Previous studies have reported correlations between certain physicochemical properties of amino acids and the locations of linear B-cell epitopes within protein sequences (Pellequer et al., 1991; Parker and Guo, 1986; Karplus and Schulz, 1985; Emini et al., 1985; Pellequer et al., 1993). Based on that observation, several amino acid propensity scale based methods have been proposed. For example, methods in (Parker and Guo, 1986; Karplus and Schulz, 1985; Emini et al., 1985; Pellequer et al., 1993) utilized hydrophilicity, flexibility, turns, and solvent accessibility propensity scales, respectively. PREDITOP (Pellequer and Westhof, 1993), PEOPLE (Alix, 1999), BEPITOPE (Odorico and Pellequer, 2003), and BcePred (Saha and Raghava, 2004) utilized groups of physicochemical properties instead of a single property to improve the accuracy of the predicted linear B-cell epitopes. Unfortunately, Blythe and Flower (Blythe and Flower, 2005) showed that propensity based methods can not be used reliably for predicting B-cell epitopes. Using a dataset of 50 proteins and an exhaustive assessment of 484 amino acid propensity scales, Blythe and Flower (Blythe and Flower, 2005) showed that the best combinations of amino acid propensities performed only marginally better than random. They concluded that the reported performance of such methods in the literature is likely to have been overly optimistic, in part due to the small size of the data sets on which the methods had been evaluated.

Recently, the increasing availability of experimentally identified linear B-cell epitopes in addition to Blythe and Flower results (Blythe and Flower, 2005) motivated several researchers to explore the application of machine learning approaches for developing linear B-cell epitope prediction methods. BepiPred (Larsen et al., 2006) combines two amino acid propensity scales and a Hidden Markov Model (HMM) trained on linear epitopes to yield a slight improvement in prediction accuracy relative to techniques that rely on analysis of amino acid physicochemical properties. ABCPred (Saha and Raghava, 2006b) uses artificial neural networks for predicting linear B-cell epitopes. Both feed-forward and recurrent neural networks were evaluated on a *non-redundant* data set of 700 B-cell epitopes and 700 non-epitope peptides, using 5-fold cross validation tests. Input sequence windows ranging from 10 to 20 amino acids, were tested and the best performance, 66% accuracy, was obtained using a recurrent neural network trained on peptides 16 amino acids in length. In the method of Söllner and Mayer (Söllner and Mayer, 2006), each epitope is represented using a set of 1487 features extracted from a variety of propensity scales, neighborhood matrices, and respective probability and likelihood values. Of two machine learning methods tested, decision trees and a nearest-neighbor method combined with feature selection, the latter was reported to attain an accuracy of 72% on a data set of 1211 B-cell epitopes and 1211 non-epitopes, using a 5-fold cross validation test (Söllner and Mayer, 2006). Chen et al. (Chen et al., 2007) observed that certain amino acid pairs (AAPs) tend to occur more frequently in B-cell epitopes than in non-epitope peptides. Using an AAP propensity scale based on this observation, in combination with a support vector machine (SVM) classifier, they reported prediction accuracy of 71% on a data set of 872 B-cell epitopes and 872 non-B-cell epitopes, estimated using 5-fold cross validation. In addition, (Chen et al., 2007) demonstrated an improvement in the prediction accuracy, 72.5%, when the APP propensity scale is combined with turns accessibility, antigenicity, hydrophilicity, and flexibility propensity scales.

Existing linear B-cell epitope prediction tools fall into two broad categories. Tools in the first category, residue-based predictors, take as input a protein sequence and assign binary labels to each individual residue in the input sequence. Each group of neighboring residues with predicted positive labels define a variable length predicted linear B-cell epitope. Residue-based prediction methods scan the input sequence using a sliding window and assign a score to the amino acid at the center of the window based on the mean score of a certain propensity scale (e.g., flexibility or hydrophilicity). The target residue is predicted positive if its score is greater than a predetermined threshold. Unfortunately, it has been shown that the performance of these methods is marginally better than random (Blythe and Flower, 2005). PepiPred (Larsen et al., 2006) used the information extracted using the sliding window to train a HMM and combined it with two propensity scale based methods. BcePred (Saha and Raghava, 2004) combined several propensity scales and showed that the performance of the combined scales is better than the performance of any single scale.

The second category of linear B-cell prediction tools consist of the epitope-based predictors. An example of such predictors is the ABCPred server (Saha and Raghava, 2006b). For this server, the input is a protein sequence and an epitope length (should be in $\{20, 18, .., 10\}$). The server then applies a sliding window of the user specified length and passes the extracted peptides to a neural network classifier trained using epitope dataset in which all the epitope sequences have been set to the specified epitope length via trimming and extending longer and shorter epitopes, respectively. A limitation of this approach is that the user is forced to select one of the available six possible epitope lengths and can not specify a different epitope length.

Because linear B-cell epitopes can vary in length over a broad range (see Figure 1), it is natural to train classifiers using the experimentally reported epitope sequences without trimming or extending them. Such an approach will allow us to provide a linear B-cell epitope prediction tool that allows the user to experiment with virtually any arbitrary epitope length. In this work, we explore two machine learning approaches for predicting flexible length linear B-cell

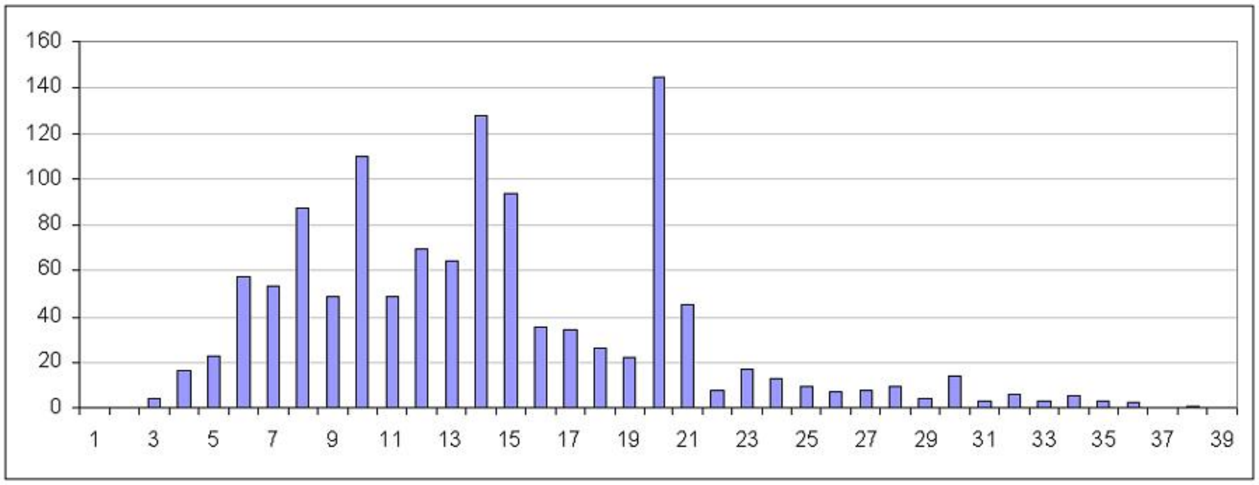


Figure 1 Length distribution of unique linear B-cell epitopes in Bcipep database.

epitopes. The first approach utilizes several sequence kernels for determining a similarity score between any arbitrary pair of variable length sequences. The second approach utilizes many different methods of mapping a variable length sequence into a fixed length feature vector. Based on our empirical comparisons, we propose FBCPred, a novel method for predicting flexible length linear B-cell epitopes using the subsequence kernel. Our results demonstrate that FBCPred significantly outperforms all other classifiers evaluated in this study. An implementation of FBCPred and the datasets used in this study are publicly available through our linear B-cell epitope prediction server, BCPREDS, at: `http://ailab.cs.iastate.edu/bcpreds/`.

Materials and methods

Data

We retrieved 1223 unique linear B-cell epitopes of lengths more than 3 amino acids from Bcipep database (Saha et al., 2005). To avoid over-optimistic performance of classifiers evaluated on the set of unique epitopes, we applied a homology reduction procedure proposed by Raghava (Raghava, 2004) for reducing sequence similarity among flexible length major histocompatibility complex class II (MHC-II) peptides. Briefly, given two peptides p_1 and

p_2 of lengths l_1 and l_2 such that $l_1 \leq l_2$, we compare p_1 with each l_1-length subpeptide in p_2. If the percent identity (PID) between p_1 and any subpeptide in p_2 is greater than 80%, then the two peptides are deemed to be similar. For example, to compute the PID between (ACDEFGHIKLMNPQRST) and (DEFGGIKLMN), we compare (DEFGGIKLMN) with (ACDEFGHIKL), (CDEFGHIKLM), ..., (IKLMNPQRST). The PID between (DEFGGIKLMN) and (DEFGHIKLMN) is 90% since nine out of 10 residues are identical.

Applying the above homology reduction procedure to the set of 1223 unique variable length linear B-cell epitopes yields a *homology-reduced* set of 934 epitopes. Two datasets of flexible length linear B-cell epitopes have been constructed. An *original* dataset constructed from the set of 1223 unique epitopes as the positive examples and 1223 non-epitopes randomly extracted from SwissProt (Bairoch and Apweiler, 2000) and a *homology-reduced* dataset constructed from *homology-reduced* set of 934 epitopes as positive examples and an equal number of negative examples extracted randomly form SwissProt sequences. In both datasets two selection criteria have been applied to the randomly extracted non-epitopes: (i) the length distribution in the negative data is identical to the length distribution in the positive data; (ii) none of the non-epitopes appears in the set of epitopes.

Support vector machines and kernel methods

Support vector machines (SVMs) (Vapnik, 2000) are a class of supervised machine learning methods used for classification and regression. Given a set of labeled training data (x_i, y_i), where $x_i \in R^d$ and $y_i \in \{+1, -1\}$, training an SVM classifier involves finding a hyperplane that maximizes the geometric margin between positive and negative training data samples. The hyperplane is described as $f(x) = \langle w, x \rangle + b$, where w is a normal vector and b is a bias term. A test instance, x, is assigned a positive label if $f(x) > 0$, and a negative label otherwise. When the training data are not linearly separable, a kernel function is used to map nonlinearly separable data from the input space into a feature space. Given any two data samples x_i and x_j in an input space $X \in R^d$, the kernel function K returns $K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ where Φ is a nonlinear map from the input space X to the corresponding feature space. The kernel function

K has the property that $K(x_i, x_j)$ can be computed without explicitly mapping x_i and x_j into the feature space, but instead, using their dot product $\langle x_i, x_j \rangle$ in the input space. Therefore, the kernel trick allows us to train a linear classifier, e.g., SVM, in a high-dimensional feature space where the data are assumed to be linearly separable without explicitly mapping each training example from the input space into the feature space. This approach relies implicitly on the selection of a feature space in which the training data are likely to be linearly separable (or nearly so) and explicitly on the selection of the kernel function to achieve such separability. Unfortunately, there is no single kernel that is guaranteed to perform well on every data set. Consequently, the SVM approach requires some care in selecting a suitable kernel and tuning the kernel parameters (if any).

Sequence kernel based methods

String kernels (Leslie et al., 2002, 2004; Lodhi et al., 2002; Saigo et al., 2004; Haussler, 1999) are a class of kernel methods that have been successfully used in many sequence classification tasks (Leslie et al., 2002, 2004; Saigo et al., 2004; Zaki et al., 2005; Rangwala et al., 2006; Wu et al., 2006). In these applications, a protein sequence is viewed as a string defined on a finite alphabet of 20 amino acids. In this work, we explore four string kernels: spectrum (Leslie et al., 2002), mismatch (Leslie et al., 2004), local alignment (Saigo et al., 2004), and subsequence (Lodhi et al., 2002), in predicting linear B-cell epitopes. A brief description of the four kernels follows.

Spectrum kernel

Let A denote a finite alphabet, e.g., the standard 20 amino acids. x and y denote two strings defined on the alphabet A. For $k \geq 1$, the k-spectrum is defined as (Leslie et al., 2002):

$$\Phi_k = (\phi_\alpha(x))_{\alpha \in A^k} \tag{1}$$

where ϕ_α is the number of occurrences of the k-length substring α in the sequence x. The k-spectrum kernel of the two sequences x and y is obtained by taking the dot product of the corresponding k spectra:

$$K_k^{spct}(x, y) = \langle \Phi_k(x), \Phi_k(y) \rangle \tag{2}$$

The k-spectrum kernel captures a simple notion of string similarity: two strings are deemed similar (i.e., have a high k-spectrum kernel value) if they share many of the same k-length substrings.

Mismatch kernel

The mismatch kernel (Leslie et al., 2004) is a variant of the spectrum kernel in which inexact matching is allowed. Specifically, the (k, m)-mismatch kernel allows up to $m \leq k$ mismatches to occur when comparing two k-length substrings. Let α be a k-length substring, the (k, m)-mismatch feature map is defined on α as:

$$\Phi_{(k,m)}(\alpha) = (\phi_\beta(\alpha))_{\beta \in A^k} \tag{3}$$

where $\phi_\beta(\alpha) = 1$ if $\beta \in N_{(k,m)(\alpha)}$, where β is the set of k-mer substrings that differs from α by at most m mismatches. Then, the feature map of an input sequence x is the sum of the feature vectors for k-mer substrings in x:

$$\Phi_{(k,m)}(x) = \sum_{k-mers\, \alpha\, in\, x} \Phi_{(k,m)}(\alpha) \tag{4}$$

The (k, m)-mismatch kernel is defined as the dot product of the corresponding feature maps in the feature space:

$$K_{(k,m)}^{msmtch}(x, y) = \langle \Phi_{(k,m)}(x), \Phi_{(k,m)}(y) \rangle \tag{5}$$

It should be noted that the $(k, 0)$-mismatch kernel results in a feature space that is identical to that of the k-spectrum kernel. An efficient data structure for computing the spectrum and mismatch kernels in $O(|x| + |y|)$ and $O(k^{m+1}|A|^m(|x| + |y|))$, respectively, is provided in (Leslie et al., 2004).

Local alignment kernel

Local alignment (LA) kernel (Saigo et al., 2004) is a string kernel adapted for biological sequences. The LA kernel measures the similarity between two sequences by summing up scores obtained from gapped local alignments of the sequences. This kernel has several parameters: the gap opening and extension penalty parameters, d and e, the amino acid mutation matrix s, and the factor β, which controls the influence of suboptimal alignments on the kernel value. Detailed formulation of the LA kernel and a dynamic programming implementation of the kernel with running time complexity in $O(|x||y|)$ are provided in (Saigo et al., 2004).

Subsequence kernel

The subsequence kernel (SSK) (Lodhi et al., 2002) generalizes the k-spectrum kernel by considering a feature space generated by the set of all (contiguous and non-contiguous) k-mer subsequences. For example, if we consider the two strings "act'' and "$acctct''$, the value returned by the spectrum kernel with $k = 3$ is 0. On the other hand, the $(3, 1)$-mismatch kernel will return 3 because the 3-mer substrings "acc'', "cct'', and "tct'' have at most one mismatch when compared with "act''. The subsequence kernel considers the set ("$ac-t''$, "$a-ct''$, "$ac--t''$, "$a-c--t''$, "$a---ct''$) of non-contiguous substrings and returns a similarity score that is weighted by the length of each non-contiguous substring. Specifically, it uses a decay factor, $\lambda \leq 1$, to penalize non-contiguous substring matches. Therefore, the subsequence kernel with $k = 3$ will return $2\lambda^4 + 3\lambda^6$ when applied to "act'' and "$acctct''$ strings. More precisely, the feature map $\Phi_{(k,\lambda)}$ of a string x is given by:

$$\Phi_{(k,\lambda)}(x) = (\sum_{i:u=x[i]} \lambda^{l(i)})_{u \in A^k} \tag{6}$$

where $u = x[i]$ denotes a substring in x where $1 \leq i_1 < \ldots < i_{|u|} \leq |x|$ such that $u_j = s_{i_j}$, for $j = 1, \ldots, |u|$ and $l(i) = i_{|u|} - i_1 + 1$ is the length of the subsequence in x. The subsequence kernel for two strings x and y is determined as the dot product of the corresponding feature maps:

$$\begin{aligned} K(x,y)^{sub}_{(k,\lambda)} &= \langle \Phi_{(k,\lambda)}(x), \Phi_{(k,\lambda)}(y) \rangle \\ &= \sum_{u \in A^k} \sum_{i:u=x[i]} \lambda^{l(i)} \sum_{j:u=y[j]} \lambda^{l(j)} \\ &= \sum_{u \in A^k} \sum_{i:u=x[i]} \sum_{j:u=y[j]} \lambda^{l(j)+l(j)} \end{aligned} \tag{7}$$

This kernel can be computed using a recursive algorithm based on dynamic programming in $O(k|x||y|)$ time and space. The running time and memory requirements can be further reduced using techniques described in (Seewald and Kleedorfer, 2005).

Sequence-to-features based methods

This approach has been previously used for protein function and structure classification tasks (Hua and Sun, 2001; Dobson and Doig, 2003; Eisenhaber et al., 1996; Luo et al., 2002) and the classification of flexible length MHC-II peptides. The main idea is to map each variable length amino acid sequence into a feature vector of fixed length. Once the variable length sequences are mapped to fixed length feature vectors, we can apply any of the standard machine learning algorithms to this problem. Here, we considered SVM classifiers trained on the mapped data using the widely used RBF kernel.

We explored four different methods for mapping a variable length amino acid sequence into a fixed length feature vector: (i) amino acid composition; (ii) dipeptide composition; (iii) amino acid pairs propensity scale; (iv) composition-transition-distribution. A brief summary of each method is given below.

Amino acid and dipeptide composition

Amino acid composition (AAC) represents a variable length amino acid sequence using a feature vector of 20 dimensions. Let x be a sequence of $|x|$ amino acids. Let A denote the set of the standard 20 amino acids. The amino acid composition feature mapping is defined as:

$$\Phi_{AAC}(x) = (\phi_\beta(x))_{\beta \in A} \tag{8}$$

where $\phi_\beta(x) = \frac{number\ of\ occurrences\ of\ amino\ acid\ \beta\ in\ x}{|x|}$.

A limitation of the amino acid composition feature representation of amino acid sequences is that we lose the sequence order information. Dipeptide composition (DC) encapsulates information about the fraction of amino acids as well as their local order. In dipeptide composition each variable length amino acid sequence is represented by a feature vector of 400 dimensions defined as:

$$\Phi_{DC}(x) = (\phi_\alpha(x))_{\alpha \in A^2} \tag{9}$$

where $\phi_\alpha(x) = \frac{number\ of\ occurrences\ of\ dipeptide\ \alpha\ in\ x}{total\ number\ of\ all\ possible\ dipeptides\ in\ x}$.

Amino acid pairs propensity scale

Amino acid pairs (AAPs) are obtained by decomposing a protein/peptide sequence into its 2-mer subsequences. (Chen et al., 2007) observed that some specific AAPs tend to occur more frequently in B-cell epitopes than in non-epitope peptides. Based on this observation, they developed an AAP propensity scale defined by:

$$\theta(\alpha) = log(\frac{f_\alpha^+}{f_\alpha^-}) \tag{10}$$

where f_α^+ and f_α^- are the occurrence frequencies of AAP α in the epitope and non-epitope peptide sequences, respectively. These frequencies have been derived from Bcipep (Saha et al., 2005) and Swissprot (Bairoch and Apweiler, 2000) databases, respectively. To avoid the dominance of an individual AAP propensity value, the scale in Eq. (10) has been normalized to a $[-1, +1]$ interval through the following conversion:

$$\theta(\alpha) = 2(\frac{\theta(\alpha) - min}{max - min}) - 1 \tag{11}$$

where max and min are the maximum and minimum values of the propensity scale before the normalization.

The AAP feature mapping, Φ_{AAP}, maps each amino acid sequence, x, into a 400-dimentional feature space defined as:

$$\Phi_{AAP}(x) = (\phi_\alpha(x) \cdot \theta(\alpha))_{\alpha \in A^2} \tag{12}$$

where $\phi_\alpha(x)$ is the number of occurrences of the 2-mer α in the peptide x.

Composition-Transition-Distribution

The basic idea behind the Composition-Transition-Distribution (CTD) method (Cai et al., 2003; Cui et al., 2006a) is to map each variable length peptide into a fixed length feature vector such that standard machine learning algorithms are applicable. From each peptide sequence, 21 features are extracted as follows:

- First, each peptide sequence p is mapped into a string s_p defined over an alphabet of three symbols, $\{1, 2, 3\}$. The mapping is performed by grouping amino acids into three groups using a physicochemical property of amino acids (see Table 1). For example the peptide (AIRHIPRRIR) is mapped into (2312321131) using the hydrophobicity division of amino acids into three groups (see Table 1).

- Second, for each peptide string s_p, three descriptors are derived as follows:

  - Composition (C): three features representing the percent frequency of the symbols, $\{1, 2, 3\}$, in the mapped peptide sequence.
  - Transition (T): three features representing the percent frequency of i followed by j or j followed by i, for $i, j \in \{1, 2, 3\}$.
  - Distribution (D): five features per symbol representing the fractions of the entire sequence where the first, 25, 50, 75, and 100% of the candidate symbol are contained in s_p. This yields an additional 15 features for each peptide.

Table 1 shows division of the 20 amino acids, proposed by Chinnasamy et al. (Chinnasamy et al., 2004) , into three groups based on hydrophobicity, polarizability, polarity, and Van der Waal's volume properties. Using these four properties, we derived 84 CTD features from each peptide sequence. In our experiments, we trained SVM classifiers using RBF kernel and peptide sequences represented using their amino acid sequence composition (20 features) and CTD descriptors (84 features).

Table 1 Categorization of amino acids into three groups for a number of physicochemical properties.

Proporty	Group 1	Group 2	Group 3
Hydrophobicity	RKEDQN	GASTPHY	CVLIMFW
Polarizability	GASCTPD	NVEQIL	MHKFRYW
Polarity	LIFWCMVY	PATGS	HQRKNED
Van der Waala volume	GASDT	CPNVEQIL	KMHFRYW

Performance evaluation

We report the performance of each classifier using the average of 10 runs of 5-fold cross validation tests. Each classifier performance is assessed by both threshold-dependent and threshold-independent metrics. For threshold-dependent metrics, we used accuracy (ACC), sensitivity (S_n), specificity (S_p), and correlation coefficient (CC). The CC measure has a value in the range from -1 to +1 and the closer the value to +1, the better the predictor. The S_n and S_p summarize the accuracies of the positive and negative predictions respectively. ACC, S_n, S_p, and CC are defined in Eq. (13-15) where TP, FP, TN, FN are the numbers of true positives, false positives, true negatives, and false negatives respectively.

For threshold-independent metrics, we report the Receiver Operating Characteristic (ROC)

$$ACC = \frac{TP+TN}{TP+FP+TN+FN} \tag{13}$$

$$S_n = \frac{TP}{TP+FN} \text{ and } S_p = \frac{TN}{TN+FP} \tag{14}$$

$$CC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TN+FN)(TN+FP)(TP+FN)(TP+FP)}} \tag{15}$$

curve. The ROC curve is obtained by plotting the true positive rate as a function of the false positive rate or, equivalently, sensitivity versus (1-specificity) as the discrimination threshold of the binary classifier is varied. Each point on the ROC curve describes the classifier at a certain threshold value and hence a particular choice of tradeoff between true positive rate and false negative rate. We also report the area under ROC curve (AUC) as a useful summary statistic for comparing two ROC curves. AUC is defined as the probability that a randomly chosen positive example will be ranked higher than a randomly chosen negative example. An ideal classifier will have an AUC = 1, while a classifier performs no better than random will have an AUC = 0.5, any classifier performing better than random will have an AUC value that lies between these two extremes.

Implementation and SVM parameter optimization

We used Weka machine learning workbench (Witten and Frank, 2005) for implementing the spectrum, mismatch, and LA kernels (RBF and SSK kernels are already implemented in Weka). We evaluated the k-spectrum kernel, K_k^{spct}, for $k = 1, 2$, and 3. The (k, m)-mismatch kernel was evaluated at (k,m) equals $(3, 1) and (4, 1)$. The subsequence kernel, $K_{(k,\lambda)}^{sub}$, was evaluated at $k = 2, 3$, and 4 and the default value for λ, 0.5. The LA kernel was evaluated using the BLOSUM62 substitution matrix, gap opening and extension parameters equal to 10 and 1, respectively, and $\beta = 0.5$. For the SVM classifier, we used the Weka implementation of the SMO (Platt, 1998) algorithm. For the string kernels, the default value of the C parameter, C = 1, was used for the SMO classifier. For methods that uses the RBF kernel, we found that tuning the SMO cost parameter C and the RBF kernel parameter γ is necessary to obtain satisfactory performance. We tuned these parameters using a 2-dimensional grid search over the range $C = 2^{-5}, 2^{-3}, \ldots, 2^3, \gamma = 2^{-15}, 2^{-13}, \ldots, 2^3$.

Results and discussion

Table 2 compares the performance of different SVM based classifiers on the *original* dataset of unique flexible length linear B-cell epitopes. The SVM classifier trained using SSK with

$k = 4$ and $\lambda = 0.5$, $k^{sub}_{(4,0.5)}$, significantly (using statistical paired t-test (Nadeau and Bengio, 2003) with p-value = 0.05) outperforms all other classifiers in terms of the AUC. The two classifiers based on the mismatch kernel have the worst AUC. The classifier trained using k^{spct}_{3} is competitive to those trained using the LA kernel and $k^{sub}_{(2,0.5)}$. The last four classifiers belong to the sequence-to-feature approach. Each of these classifiers has been trained using an SVM classifier and the RBF kernel but on different data representation. The results suggest that representation of the peptides using their dipeptide composition performs better than other feature representations on the *original* dataset. Figure 2 shows the ROC curves for different methods on *original* dataset of unique flexible length linear B-cell epitopes. The ROC curve of $K^{sub}_{(4,0.5)}$ based classifier almost dominates all other ROC curves (i.e., for any choice of specificity value, the $K^{sub}_{(4,0.5)}$ based classifier almost has the best sensitivity) .

Table 3 reports the performance of the different SVM based classifiers on the *homology-reduced* dataset of flexible length linear B-cell epitopes. We note that the performance of each classifier is considerably worse than its performance on the *original* dataset of unique epitopes. This discrepancy can be explained by the existence of epitopes with significant pairwise sequence similarity in the *original* dataset. Interestingly, the SVM classifier based on the $k^{sub}_{(4,0.5)}$ kernel still significantly outperforms all other classifiers at 0.05 level of significance. Figure 3 shows the ROC curves for different methods on *homology-reduced* dataset of flexible length linear B-cell epitopes. Again, the ROC curve of $K^{sub}_{(4,0.5)}$ based classifier almost dominates all other ROC curves.

Comparing results on Table 2 and Table 3 reveals two important issues that to the best of our knowledge have not been addressed before in the literature on B-cell epitope prediction. First, our results demonstrate that performance estimates reported on the basis of the *original* dataset of unique linear B-cell epitopes is overly optimistic compared to the performance estimates obtained using the *homology-reduced* dataset. Hence, we suspect that the actual performance of linear B-cell epitope prediction methods on *homology-reduced* datasets is somewhat lower than the reported performance on the original dataset of unique peptides. Second, our results suggest that conclusions regarding how different prediction methods compare to each other drawn on the basis of datasets of unique epitopes may be misleading. For example, from the reported results in Table 2, one may conclude that k_3^{spct} outperforms k_1^{spct} and k_2^{spct} while results on the *homology-reduced* dataset (see Table 3) demonstrate that the three classifiers are competitive with each other. Another example of misleading conclusions drawn from results in Table 2 is that dipeptide composition features is a better representation than amino acid composition representation of the data. This conclusion is contradicted by results in Table 3 which show that the classifier constructed using the amino acid composition representation of the data slightly outperforms the classifier constructed using the dipeptide composition of the same data.

The results in Table 2 and Table 3 show that the classifier that used the amino acid composition features outperforms the classifier that used CTD features. This is interesting because the set of amino acid composition features is a subset of the CTD features. Recall that CTD is composed of 20 amino acid composition features plus 84 physicochemical features, we conclude that the added physicochemical features did not yield additional information that was relevant for the classification task. In addition, we observed that the classifier that used the dipeptide composition outperforms the classifier that used the AAP features. This is interesting because AAP features as defined in Eq. (12) can be viewed as dipeptide composition features weighted by the amino acid propensity of each dipeptide.

Web server

An implementation of FBCPred is available as a part of our B-cell epitope prediction server (BCPREDS) (EL-Manzalawy et al., 2008d) which is freely accessible at `http://ailab.cs.iastate.edu/bcpreds/`. Because it is often valuable to compare predictions of multiple methods, and consensus predictions are more reliable than individual predictions, the BCPREDS server aims at providing predictions using several B-cell epitope prediction methods. The current implementation of BCPREDS allows the user to select among three prediction methods: (i) Our implementation of AAP method (Chen et al., 2007); (ii) BCPred (EL-Manzalawy et al., 2008d), a method for predicting linear B-cell epitope using the subsequence kernel; (iii)

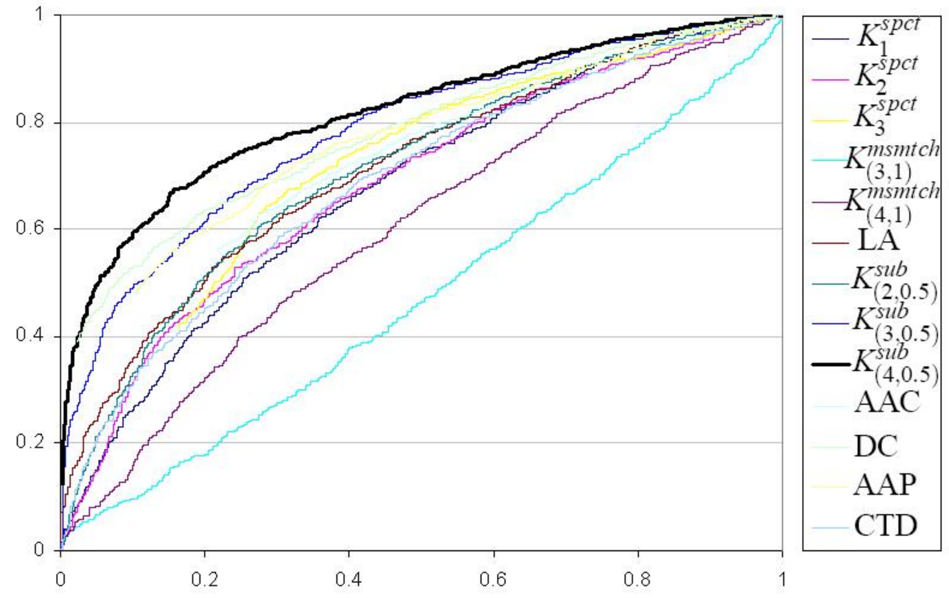


Figure 2 ROC curves for different methods on *original* dataset of unique flexible length linear B-cell epitopes. The ROC curve of $K_{(4,0.5)}^{sub}$ based classifier almost dominates all other ROC curves.

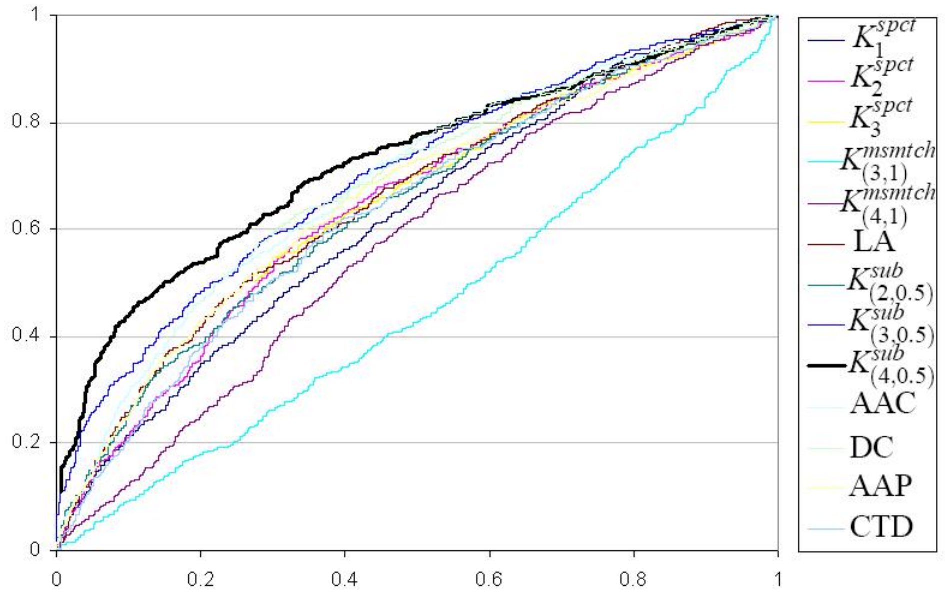


Figure 3 ROC curves for different methods on *homology-reduced* dataset of flexible length linear B-cell epitopes. The ROC curve of $K_{(4,0.5)}^{sub}$ based classifier almost dominates all other ROC curves.

Table 2 Performance of different SVM based classifiers on original dataset of unique flexible length linear B-cell epitopes. Results are the average of 10 runs of 5-fold cross validation.

Method	ACC	Sn	Sp	CC	AUC
K_1^{spct}	62.86	61.76	63.95	0.257	0.680
K_2^{spct}	63.29	63.84	62.74	0.266	0.683
K_3^{spct}	65.36	79.28	51.44	0.320	0.720
$K_{(3,1)}^{msmtch}$	47.88	48.42	47.33	-0.042	0.480
$K_{(4,1)}^{msmtch}$	58.93	57.79	60.07	0.179	0.618
LA	65.41	63.36	67.46	0.308	0.716
$K_{(2,0.5)}^{sub}$	65.58	65.08	66.09	0.312	0.710
$K_{(3,0.5)}^{sub}$	70.56	71.05	70.07	0.411	0.778
$K_{(4,0.5)}^{sub}$	**73.37**	**74.08**	72.67	**0.468**	**0.812**
AAC	65.61	68.41	62.81	0.313	0.722
DC	70.55	68.28	**72.83**	0.411	0.750
AAP	65.65	66.20	65.11	0.313	0.717
CTD	63.21	63.15	63.28	0.264	0.686

FBCPred, the method introduced in this study for predicting flexible length B-cell epitopes. The major difference between FBCPred and the other two methods is that FBCPred can predict linear B-cell epitopes of virtually any arbitrary length while for the other two methods the length has to be one of possible six values, $\{12, 14, \ldots, 22\}$.

Another goal of BCPREDS server is to serve as a repository of benchmark B-cell epitope datasets. The datasets used for training and evaluating BCPred and the two datasets used in this study can be freely downloaded from the web server.

Summary and discussion

We explored two machine learning approaches for predicting flexible length linear B-cell epitopes. The first approach utilizes sequence kernels for determining a similarity score between any arbitrary pair of variable length sequences. The second approach utilizes several methods of mapping a variable length sequence into a fixed length feature vector. Our results demonstrated a superior performance of the subsequence kernel based SVM classifier compared to other SVM classifiers examined in our study. Therefore, we proposed FBCPred, a novel

Table 3 Performance of different SVM based classifiers on homology-reduced dataset of flexible length linear B-cell epitopes. Results are the average of 10 runs of 5-fold cross validation.

Method	ACC	Sn	Sp	CC	AUC
K_1^{spct}	58.22	56.70	59.74	0.165	0.621
K_2^{spct}	60.26	61.04	59.49	0.205	0.642
K_3^{spct}	60.86	62.45	59.27	0.217	0.635
$K_{(3,1)}^{msmtch}$	46.42	46.34	46.50	-0.072	0.451
$K_{(4,1)}^{msmtch}$	54.35	54.75	53.95	0.087	0.561
LA	61.38	60.41	62.35	0.228	0.658
$K_{(2,0.5)}^{sub}$	60.09	60.52	59.66	0.202	0.647
$K_{(3,0.5)}^{sub}$	63.85	65.05	62.65	0.277	0.701
$K_{(4,0.5)}^{sub}$	**65.49**	68.36	62.61	**0.310**	**0.738**
AAC	63.31	**70.90**	55.73	0.269	0.683
DC	63.78	63.05	**64.51**	0.276	0.667
AAP	61.42	62.85	60.00	0.229	0.658
CTD	60.32	59.66	60.98	0.206	0.639

method for predicting flexible length linear B-cell epitopes using the subsequence kernel. An implementation of FBCPred and the datasets used in this study are publicly available through our linear B-cell prediction server, BCPREDS, at: `http://ailab.cs.iastate.edu/bcpreds/`.

Previous methods for predicting linear B-cell epitopes (e.g., (Saha and Raghava, 2004; Larsen et al., 2006; Söllner and Mayer, 2006; Saha and Raghava, 2006b; Chen et al., 2007)) have been evaluated on datasets of unique epitopes without applying any homology reduction procedure as a pre-processing step on the data. We showed that performance estimates reported on the basis of such datasets is considerably over-optimistic compared to performance estimates obtained using the *homology-reduced* datasets. Moreover, we showed that using such *non homology-reduced* datasets for comparing different prediction methods may lead to false conclusions regarding how these methods compare to each other.

Related work

Residue-based prediction methods (Pellequer et al., 1991; Parker and Guo, 1986; Karplus and Schulz, 1985; Emini et al., 1985; Pellequer et al., 1993; Saha and Raghava, 2004; Larsen et al., 2006) assign labels to each residue in the query sequence and therefore are capable of predicting linear B-cell epitopes of variable length. However, most of these methods have been shown to be of low to moderate performance (Blythe and Flower, 2005).

AAP method (Chen et al., 2007) maps each peptide sequence into a set of fixed length numeric features and therefore it can be trained using datasets of flexible length sequences. However, the performance of this method had been reported using a dataset of 20-mer peptides.

Söllner and Mayer (Söllner and Mayer, 2006) introduced a method for mapping flexible length epitope sequences into feature vectors of 1478 attributes. This method has been evaluated on a dataset of flexible length linear B-cell epitopes. However, no homology reduction procedure was applied to remove highly similar sequences from the data. In addition, the implementation of this method is not publicly available.

Recently, two methods (Salomon and Flower, 2006; Cui et al., 2006a) have been successfully applied to the problem of predicting flexible length MHC-II binding peptides. The first method (Salomon and Flower, 2006) utilized the LA kernel (Saigo et al., 2004) for developing efficient SVM based classifiers. The second method (Cui et al., 2006a) mapped each flexible length peptide into the set of CTD features employed in our study in addition to some extra features extracted using two secondary structure and solvent accessibility prediction classifiers. In our study we could not use these extra features due to the unavailability of these two programs.

Acknowledgments

This work was supported in part by a doctoral fellowship from the Egyptian Government to Yasser EL-Manzalawy and a grant from the National Institutes of Health (GM066387) to Vasant Honavar and Drena Dobbs.

CHAPTER 4. QUALITATIVE VERSUS QUANTITATIVE APPROACHES FOR PREDICTING MHC-I PEPTIDES

A paper to be submitted to the Journal of Immunome Research

Yasser EL-Manzalawy and Vasant Honavar

Abstract

Several methods for predicting peptide binding to major histocompatibility complex class I (MHC-I) molecules have been proposed in the literature. These methods can be categorized into two major types: i) qualitative methods, which predict whether a test peptide is an MHC-I binder or non-binder; ii) quantitative methods, which predict the value of the binding affinity of a test peptide. For each type, we compare a representative set of scoring matrix based and direct machine learning based prediction methods over a similarity-reduced data set covering 22 MHC-I human leukocyte antigen (HLA) alleles.

The 13 methods considered in this study include three new matrix based methods for qualitative MHC-I binding peptide prediction: i) modified position specific scoring matrix (MPSSM), a method for computing a PSSM from both binding and non-binding training peptides; ii) area under receiver operating characteristic curve (AUC) optimized matrix method (AOMM), a method for finding a scoring matrix that maximizes the AUC over the training data; iii) SMMBin, a method for qualitative via quantitative (QVQ) prediction using the Stabilized Matrix Method (SMM).

The results of our experiments show that quantitative MHC-I predictors based on direct machine learning approaches outperform qualitative MHC-I predictors based on both machine learning and scoring matrix methods. However, the observed differences between the best-performing quantitative MHC-I predictor and AOMM, SMMBin and support vector machine (SVM) methods are not statistically significant. We conclude that advanced scoring matrix based machine learning methods (e.g., AOMM, SMM) are highly competitive with a broad class of machine learning methods for predicting MHC-I peptides. Our similarity-reduced data set and an implementation of an online web server, MHCIPREDS, supporting predictions using the best performing qualitative and quantitative MHC-I prediction methods considered in this comparison study are freely available at http://ailab.cs.iastate.edu/mhcipreds.

Introduction

Major histocompatibility complex (MHC) molecules play a crucial role in the immune system dynamics by binding short peptides and presenting them for recognition by T-cell receptors. Peptides presented by MHC class I (MHC-I) molecules are derived from proteasomal degradation of intracellular proteins and their lengths range from 8 to 11 amino acids. Peptides presented by MHC class II (MHC-II) molecules are derived from endogenous proteins or intracellular pathogens and the binding peptides range from 11 to 30 residues in length. The recognition of Peptide-MHC complexes aids killer T-cell in identifying and destroying abnormal or foreign cells. Peptides that can complete this pathway are called T-cell epitopes. The identification of peptides binding to MHC molecules is a major step in identifying T-cell epitopes. However, due to the fact that MHC genes are highly polymorphic and there exists a substantial number of possible antigens, computational methods for identifying MHC binding peptides are urgently needed to reduce the time and the cost of the laboratory work required for mapping T-cell epitopes.

A variety of methods for predicting MHC-I binding peptides from amino acid sequence information have been proposed. Examples of these MHC-I peptide prediction methods include methods based on: scoring matrices (Parker et al., 1994; Rammensee et al., 1999; Reche et al., 2004; Bui et al., 2005; Peters and Sette, 2005); hidden Markov models (HMM) (Mamitsuka, 1998); additive method (Hattotuwagama et al., 2004); artificial neural networks (ANN) (Buus et al., 2003); support vector machine (SVM) (Donnes and Kohlbacher, 2006); support vector regression (SVR) (Liu et al., 2006). These methods can be categorized into two major types: i) qualitative methods (e.g., (Reche et al., 2004; Donnes and Kohlbacher, 2006)), which predict whether a test peptide is an MHC-I binder or non-binder; ii) quantitative methods (e.g., (Hattotuwagama et al., 2004; Liu et al., 2006; Peters and Sette, 2005)), which predicts the value of the binding affinity of a test peptide.

Recently, two studies (Trost et al., 2007; Lin et al., 2008) compared several publicly available MHC-I peptide prediction servers. Trost et al. (2007) used test data sets extracted from the community resource MHC-I benchmark (Peters et al., 2006) to compare 16 MHC-I peptide prediction servers and showed that a consensus method combining the predictions of the 16 servers is performing better than any individual server. Lin et al. (2008) compared 30 MHC-I peptide prediction servers using a test data set of two antigens and showed that the best prediction servers are implemented using ANN and SMM based predictors. An important limitation of such comparison studies is that they neglect the possibility that some servers (especially those that are recently developed) may have a considerable amount of overlap between the training and test data. Furthermore, because different servers have been developed using different training data sets, the results of such comparison studies should be interpreted as comparisons between different servers and not as comparisons between the underlying prediction methods utilized to develop these servers. For example, we may conclude that the prediction servers developed using ANN or SVM algorithms are performing better than the servers developed using matrix based methods. However, the conclusion that ANN and SVM based prediction methods outperform matrix based methods can not be judged from such comparison studies. Consequently, studies comparing different MHC-I peptide prediction servers may be important only from the user perspective because these studies can guide the user in selecting a server to use among several existing servers. Alternatively, studies for directly comparing an extensive number of MHC-I peptide prediction methods are needed to improve our understanding of the peptide-MHC binding problem and to facilitate more advances in that field of study.

Against this background, we compare a representative set of scoring matrix based and machine learning based prediction methods using a similarity-reduced data set covering 22 MHC-I human leukocyte antigen (HLA) alleles. The 13 methods considered in this study include three new matrix based methods for qualitative MHC-I binding peptide prediction: i) modified position specific scoring matrix (MPSSM), a method for computing a PSSM from both binding and non-binding training peptides; ii) area under ROC curve (AUC) optimized matrix method (AOMM), a method for finding a scoring matrix that maximizes the AUC over the training data; iii) SMMBin, a method for qualitative via quantitative (QVQ) prediction using the Stabilized Matrix Method (SMM).

The results of our experiments show that quantitative MHC-I predictors based on direct machine learning approaches outperform qualitative MHC-I predictors based on both machine learning and scoring matrix methods. However, the observed differences between the best-performing quantitative MHC-I predictor and AOMM, SMMBin and SVM methods are not statistically significant. We conclude that advanced scoring matrix based machine learning methods (e.g., AOMM, SMM) are highly competitive with a broad class of machine learning methods for predicting MHC-I peptides. Our similarity-reduced data set and an implementation of an online web server, MHCIPREDS, supporting predictions using the best performing qualitative and quantitative MHC-I prediction methods considered in this comparison study are freely available at http://ailab.cs.iastate.edu/mhcipreds.

Methods

Qualitative and quantitative data

We constructed a benchmarking data set from the Immune Epitope Database and Analysis Resource (IEDB) (Peters et al., 2005), which is a rich resource of MHC binding data curated from the literature or submitted by immunologists. For each reported peptide, IEDB provides qualitative (i.e., Negative or Positive) and quantitative (i.e., IC50) measurements whenever available. We used both qualitative and quantitative measurements for constructing 22 HLA binary labeled data sets as follows:

- Peptides with no reported quantitative measurements are discarded.
- Peptides with “Positive” qualitative measurement and quantitative measurement less than 500 nM are classified as binders.
- Peptides with “Positive” qualitative measurement and quantitative measurement greater than or equal 500 nM are classified as non-binders.
- Peptides with “Negative” qualitative measurement and quantitative measurement greater than or equal 500 nM are classified as non-binders.
- Peptides with “Negative” qualitative measurement and quantitative measurement less than 500 nM are discarded.

The length of peptides in each allele data set ranges from 8 to 11 amino acids. Because the prediction methods considered in this study require all peptides to be of the same length, we grouped peptides in each allele data set by their lengths and created a length-specific allele data set only if the number of both length-specific binders and length-specific non-binders is more than 100 peptides.

To avoid overly optimistic performance of prediction methods, we applied a similarity reduction filter to the set of binders and non-binders separately in each length-specific allele data set. Two peptides were considered similar if they share a sequence similarity $\geq$ 80%. Table 1 summarizes the resultant qualitative MHC-I data set.

Finally, a quantitative MHC-I data set was derived from the qualitative data set described above by replacing the binary label of each peptide with the associated binding affinity (e.g., IC50) reported in IEDB. Then, the IC50 scores in nM units are log-transformed using the relation $1 - log_{50k}(IC50\, nM)$.

Modified position specific scoring matrix (MPSSM)

Let a sequence motif $S = \{s_1, s_2, \ldots, s_n\}$ denote a set of fixed length substrings such that each substring, $s_i \in \Sigma^k$, is drawn from a finite alphabet Σ of the twenty amino acids.

PSSM is a commonly used probabilistic representation of sequence motifs. Given a motif S of $k-$length subsequences in Σ^k, an $k \times |\Sigma|$ matrix M is constructed using Eq. 1:

$$M_{ij} = log\frac{p_{ij}}{q_{ij}} \tag{1}$$

where p_{ij} is the probability of finding amino acid j at position i in S subsequences and q_{ij} is the probability of finding amino acid j at position i in a background model. Assuming that positions within the motif are independent of each other, the probability that an input subsequence s belongs to the motif is defined as $\sum_{i=1}^{k} M_{is[i]}$, where $s[i]$ is the i^{th} amino acid in s.

For small data sets, some amino acids may not appear at all in some positions of the motif. In such cases, the estimated p_{ij} probability will be zero. To avoid zero probabilities, pseudo-counts based on substitution probabilities have been proposed by Henikoff and Henikoff (1996).

An advantage of the PSSM method is that only positive data is required to build a model assuming a background probability distribution (e.g., unified background probability distribution of amino acids). For the classification task at hand, building a PSSM using only the set of binding peptides has the limitation of not utilizing all the available information in the training data in order to improve the predictive performance. Therefore, we propose a slight modification of the PSSM method described above where both positive and negative training data are utilized to build the PSSM matrix. First, the foreground probabilities p_{ij} are estimated from the set of binders, and pseudo-counts based on BLOSUM62 are computed as described in (Henikoff and Henikoff, 1996) to avoid zero probabilities. Second, the same procedure is applied to the set of non-binders to estimate the background probabilities q_{ij}. Finally, Eq. 1 is used to compute the resultant PSSM matrix.

AUC optimized matrix method (AOMM)

The idea of maximizing the AUC has been previously explored in the context of several machine learning algorithms including AUC optimized artificial neural networks (Yan et al., 2003), RankOpt (Herschtal and Raskutti, 2004), and AUC maximizing SVM (Brefeld and Scheffer, 2005). Inspired by this machine learning work in developing AUC optimized classifiers, we introduce a novel method for determining a position specific scoring matrix (PSSM) that maximizes the AUC over the training data.

Let D denote a set of k-length training MHC-I peptides. $D = D^+ \cup D^-$ where D^+ and D^- denote the set of binding and non-binding examples, respectively. Let $X = X^+ \cup X^-$ denote a binary representation of peptides in D where each amino acid is represented by a 20-bit binary string of 19 zeros and one in the position corresponding to the alphabetical order of that amino acid in the set of the standard 20 amino acids. For example, amino acids A and C will be represented as 10000000000000000000 and 01000000000000000000, respectively. Let M be an $k \times 20$ scoring matrix model of D (i.e., M could be a position specific scoring matrix (PSSM)). Given a test peptide p, the score (rank) assigned to p by matrix M can be written as $M(p) = w \cdot x$, where w is a weight vector derived by concatenating all rows in M and x is the binary representation of p. This notation implies that finding a scoring matrix M that maximizes the AUC over D is equivalent to finding a weight vector w that maximizes the AUC over X.

The AUC of a scoring matrix M over a set of training k-length peptides $AUC(M, D)$ or equivalently the AUC of a weight vector w over X, $AUC(w, X)$, where X is the binary representation of D has been shown equal to the normalized Wilcoxon-Mann-Whitney statistic (WMW) (Mann and Whitney, 1947) in the form:

$$AUC(M, D) = AUC(w, X) = \frac{1}{|X^+||X^-|} \sum_{i=1}^{|X^+|} \sum_{j=1}^{|X^-|} 1_{w \cdot x_i^+ > w \cdot x_j^-} \tag{2}$$

The indicator function 1_α returns 1 iff α is true, otherwise it returns zero. Because the indicator function is non-differentiable, Yan et al. (2003) suggested maximizing an approximation of WMW by replacing the indicator function with a sigmoid function $g(x) = \frac{1}{1+e^{-\beta x}}$. The current implementation of AOMM applies a gradient algorithm to maximize the objective function proposed by Yan et al. (2003) plus a regularization term in order to avoid overfitting the training data (see Eq. 3).

$$R(w, D) = \frac{1}{|X^+||X^-|} \sum_{i=1}^{|X^+|} \sum_{j=1}^{|X^-|} g(w \cdot (x_i^+ - x_j^-)) - \frac{1}{2}\lambda \sum_{l=1}^{k} w_l^2 \tag{3}$$

Qualitative via quantitative (QVQ) approach

Classification via regression (CVR) (Frank et al., 1998) is a machine learning approach where a regression algorithm (regressor) is used for a binary classification task. Simply, the $\{0, 1\}$ binary labels are treated as real value labels and the regressor is trained in such settings. For assigning a binary label to a test instance, the regressor returns a real value and a positive label is returned if and only if the real value returned by the regressor is greater than a threshold value (e.g., 0.5). Otherwise, a negative label is returned.

In this study, we apply the same idea to solve a qualitative MHC-I prediction task using a quantitative matrix method. Our proposed method, SMMBin, employs the quantitative matrix method, SMM, for qualitative MHC-I predictions. The suggested approach is not limited to SMM but it is applicable to any quantitative matrix based or machine learning based algorithm.

Implementation and parameters settings

We implemented PSSM, MPSSM, and AOMM in Java using Weka machine learning workbench (Witten and Frank, 2005). All machine learning classification and regression methods considered in this study are already available in Weka. We also used the Visual C++ implementation of the SMM method freely available at `http://70.167.3.42/smm/`.

All the methods have been evaluated using their default parameter settings provided by the developers of these methods except for the SMO (Platt, 1998) implementation of the SVM classifier where we turned on the parameter "M" in order to allow the SVM classifier to return the estimated probabilities. For the AOMM, the default value for β is 2 and the default value for λ is 0.01.

Performance evaluation

Receiver Operating Characteristic (ROC) analysis is probably the most widely used technique for assessing the performance of machine learning algorithms and the majority of bioinformatics classification methods. The ROC curve is obtained by plotting the true positive rate as a function of the false positive rate or, equivalently, sensitivity versus (1-specificity) as the discrimination threshold of the binary classifier is varied. Each point on the ROC curve describes the classifier at a certain threshold value and hence a particular choice of tradeoff between true positive rate and false negative rate. The area under ROC curve (AUC) is a useful summary statistic for comparing two ROC curves. AUC is defined as the probability that a randomly chosen positive example will be ranked higher than a randomly chosen negative example. An ideal classifier will have an AUC = 1, while a classifier performs no better than random will have an AUC = 0.5, any classifier performing better than random will have an AUC value that lies between these two extremes. In our experiments, we use the AUC to assess the performance of both qualitative and quantitative MHC-I predictors.

The ROC analysis measures how good the predictors in ranking test data. A good predictor is expected to assign higher ranks to positive data than negative data. Such a predictor can easily discriminate between positive and negative instances. However, for quantitative MHC-I predictors, we need also to assess how good is the correlation between predicted and actual binding affinities. For this purpose, we used Pearson correlation coefficient (PCC) (Urdan, 2005) to assess the performance of quantitative methods.

Statistical analysis of the results

When comparing multiple predictors on multiple data sets, it is useful to know (from a statistical perspective) whether the differences in the reported performance of different predictors are significant or not. To address this question, we utilized multiple hypothesis comparisons (Friedman, 1940; Fisher, 1973) for comparing a set of classifiers on multiple data sets. We followed a three-step non-parametric approach suggested by Demšar (2006). Briefly, the predictors being compared are ranked on the basis of their observed performance (AUC or PCC) on each data set (see Table 2 as an example). Then, Friedman test is applied to determine whether the measured average ranks are significantly different from the mean rank under the null hypothesis. Finally, if the null hypothesis can be rejected at 0.05 significance level, the Nemenyi test is used to determine whether significant differences exist between any given pair of classifiers.

Results and discussion

We report the results of comparing a representative set of scoring matrix based and machine learning based methods for qualitative and quantitative MHC-I predictions. For qualitative MHC-I predictions, we compared PSSM (Henikoff and Henikoff, 1996), MPSSM, AOMM, and SMMBin matrix based methods with Naive Bayes (NB) (Mitchell, 1997), Logistic Regression (LR) (Le Cessie and Van Houwelingrn, 1992), AdaBoost (Boost) with 50 Decision Stump base classifiers (Freund and Schapire, 1996), Alternating Decision Tree (ADTree) (Freund and Mason, 1999), and Support Vector Machine (SVM) (Vapnik, 2000) with linear kernel. For quantitative MHC-I predictions, we considered the Stabilized Matrix Method (SMM) (Peters and Sette, 2005) and three regression algorithms; Linear Regression (LR) (Witten and Frank, 2005), model trees (M5P) (Quinlan, 1992; Wang and Witten, 1997), and Gaussian Process (GP) (MacKay, 1998) with linear kernel. The predictive performance of these classifiers was estimated using the standard 10-fold cross-validation procedure.

Comparison of qualitative MHC-I predictors

The statistical post-hoc test, Table 3, shows that the three proposed matrix based methods, MPSSM, AOMM, and SMMBin, are significantly performing better than the PSSM method. This result suggest that incorporating non-binding information in building scoring matrices leads to significant improvements in predicting MHC-I binding peptides. This result is consistent with the results reported in the two recent comparison studies (Trost et al., 2007; Lin et al., 2008) where the two servers implementing ARB (Bui et al., 2005) and SMM (Peters et al., 2006) demonstrated a better performance than Rankpep (Reche et al., 2004) which was implemented using the PSSM method.

The AOMM is performing slightly better than MPSSM (average rank is 5.66 versus 7.75 for the MPSSM) but the difference in performance is not statistically significant. Interestingly, the method SMMBin, implementing our proposed qualitative via quantitative (QVQ) approach, has the best average rank among all MHC-I qualitative prediction methods and its performance is significantly petter than the remaining qualitative methods except AOMM and SVM.

SVM combined with a linear kernel has the best average rank among machine learning based qualitative MHC-I peptide prediction methods (Table 2). However, a significant difference in performance is reported only when comparing with ADTree method (see Table 3). The simple but yet efficient NB classier demonstrates a competitive performance with SVM.

Comparison of quantitative MHC-I predictors

Table 2 shows that the GP methods combined with the linear kernel has the best average rank, 3.1, followed by M5P and LnrReg with the ranks 4.12 and 4.29, respectively. However, Table 3 shows the existence of no significant differences among the three machine learning regression methods, LnrReg, M5P, and GP. However, the three methods significantly outperform the quantitative matrix methods, SMM.

Table 4 compares the four MHC-I quantitative prediction methods in terms of Pearson correlation coefficients between the actual and predicted binding affinities. The three machine learning regression methods are competitive in performance to each other with the average ranks 1.78, 2.16, and 2.31 for GP, M5P, and LnrReg, respectively. SMM has the lowest average rank, 3.63. Furthermore, the application of the Friedman test indicated the rejection of the null hypothesis at 0.05 level of significance. A Nemenyi test comparing each pair of predictors suggested that the three regression methods are significantly better than SMM and no significance difference in performance exists among the three regression methods.

It should be mentioned that the SMM performance reported in this study is different from that reported in the study reported by Peters et al. (2006) due to two major differences between the two studies according to the data sets and the experimental settings. First, the data sets used in our experiments are similarity-reduced while no similarity reduction was applied to the data sets in (Peters et al., 2006). Hence, the reported performance of SMM in (Peters et al., 2006) is expected to be overly optimistic. Second, Peters et al. (2006) evaluated SMM using 5-fold cross-validation and tuned some parameters of the SMM program while we evaluated SMM using 10-fold cross-validation and using the default SMM parameters settings. However, regardless these differences, both studies demonstrate that machine learning regression methods outperform SMM method on predicting MHC-I binding affinities.

Matrix based versus machine learning based methods

Table 2 indicates that the best performing method is the GP (average rank = 3.1) which is a machine learning based quantitative MHC-I prediction method. This result suggests that including the binding affinities in training MHC-I prediction methods may help improving the predictive performance. Furthermore, quantitative MHC-I predictors have the advantage of predicting the binding affinity of the query peptide not just whether the query peptide is a binder or not.

The two best performing qualitative MHC-I predictors are SMMBin and AOMM with average ranks 3.57 and 5.66, respectively. Interestingly, these two methods are matrix based methods and their performance although lower than GP but no statistically significant difference is observed. Hence, our results suggest that advanced scoring matrix methods trained using both binding and non-binding information (e.g., AOMM and SMMBin) are highly competitive with a representative set of machine learning methods for predicting MHC-I peptides.

Unlike several machine learning methods, scoring matrix methods requires less memory and the running time required for scanning a submitted sequence using a scoring matrix is generally much less than the time required for performing the scan using many machine learning based predictors. Therefore, matrix based methods may be preferred for developing MHC-I prediction servers (specially, when the server is supporting predictions for a large number of MHC-I alleles). The faster testing time of matrix based methods compared with many other machine learning based predictors suggests that matrix based predictors may be preferred for large scale studies (e.g., for identifying potential MHC-I alleles on a genomic scale).

MHCIPREDS web server

Several studies (Yu et al., 2002; Peters et al., 2006; Trost et al., 2007; Lin et al., 2008), including this study, demonstrate that there is no single MHC-I prediction method that outperforms other methods consistently on every MHC-I allele data set. To deal with this limitation of existing prediction tools, users are recommended to submit their query sequence to more than one prediction tool. The underlying hypothesis is that consensus predictions are more reliable than individual predictions. We believe that having an MHC-I prediction server that allows users to get predictions from multiple MHC-I predictors, covering the major approaches for predicting MHC-I peptides, in one web server that does not rely on submitting the query sequence to other external MHC-I prediction servers would be a very useful resource for experimentalists to plan their MHC-I peptide mapping experiments. To facilitate this goal, we implemented MHCIPREDS, an online web server for MHC-I peptide predictions. The server supports predictions using the best performing qualitative and quantitative MHC-I prediction methods considered in this comparison study. Specifically, the server supports qualitative MHC-I predictions using MPSSM, AOMM, SMMBin, and SVM predictors. For quantitative MHC-I predictions, the server provides predictions using SMM and GP predictors. The current implementation of the server provides predictions for the 22 MHC-I HLA alleles conceived in our comparisons. However, we plan to enrich the server with more MHC-I alleles as soon as enough training data become available. The data sets used in our experiments and the web server can be freely accessed via http://ailab.cs.iastate.edu/mhcipreds.

Conclusion

We empirically compared four major approaches for predicting MHC-I peptides: i) matrix based qualitative MHC-I binding peptide predictions; ii) machine learning based qualitative MHC-I binding peptide predictions; iii) matrix based quantitative MHC-I peptides binding affinity predictions; v) machine learning based quantitative MHC-I peptides binding affinity predictions.

For each prediction approach, a representative set of prediction methods has been evaluated on a similarity-reduced MHC-I benchmark data set covering 22 MHC-I HLA alleles using 10-fold cross-validation tests. A statistical multiple hypothesis comparison test has been conducted to assess our conclusions regarding how each method compares to the others.

In addition, we proposed three matrix based qualitative scoring matrix methods and demonstrated their usefulness in predicting MHC-I binding peptides. Specifically, we proposed: i) MPSSM, a method for finding a PSSM matrix model using both binding and non-binding training information; ii) AOMM, a method for finding a PSSM matrix model that maximizes the AUC over the training data; iii) SMMBin, a method for finding a PSSM matrix model using a quantitative matrix method, SMM (Peters et al., 2006).

Our results demonstrated that out of the 13 prediction methods evaluated in this study, the Gaussian Process (GP) (MacKay, 1998) machine learning based regression method combined with the linear kernel has the best predictive performance. However, the performance of GP predictors is not significantly different from the observed performance of AOMM and SMMBin predictors. Hence, we conclude that advanced scoring matrix methods trained using both binding and non-binding information (e.g., AOMM and SMMBin) are highly competitive with a representative set of machine learning methods for predicting MHC-I peptides.

An online web server, MHCIPREDS, supporting predictions using the best performing methods in each MHC-I prediction approach was implemented and is freely accessible through the link `http://ailab.cs.iastate.edu/mhcipreds`.

Acknowledgments

This research was supported in part by a doctoral fellowship from the Egyptian Government to Yasser El-Manzalawy and a grant from the National Institutes of Health (GM066387) to Vasant Honavar.

Table 1 Summary of the MHC-I benchmark data set. Binding peptides were identified using an IC50 binding threshold of 500 nM.

Allele	Length	binders	non-binders
A0101	9	160	1911
A0201	10	614	792
A0201	9	1395	2398
A0202	10	472	586
A0202	9	691	672
A0203	10	482	574
A0203	9	697	666
A0206	10	361	693
A0206	9	601	757
A0301	10	523	602
A0301	9	628	2005
A1101	10	578	547
A1101	9	806	1763
A2402	9	197	566
A2902	10	108	113
A2902	9	167	368
A3002	9	145	270
A3101	10	386	716
A3101	9	506	1929
A3301	10	163	934
A3301	9	202	977
A6801	10	531	570
A6801	9	572	611
A6802	10	264	789
A6802	9	418	1226
B0702	10	103	146
B0702	9	228	1490
B1501	9	184	1156
B3501	9	219	260
B4002	9	70	69
B4501	9	68	66
B5101	9	98	405
B5301	9	108	167
B5401	9	91	189

Table 2 AUC values for different methods evaluated on the similarity-reduced MHC-I data set. For each data set, the rank of each predictor is shown in parentheses.

Allele	Length	PSSM	MPSSM	AOMM	SMMBin	NB	LR
A0101	9	0.94(7.0)	0.94(7.0)	0.95(3.0)	0.95(3.0)	0.93(10.5)	0.93(10.5)
A0201	10	0.84(12.0)	0.89(7.5)	0.89(7.5)	0.9(3.5)	0.89(7.5)	0.89(7.5)
A0201	9	0.92(10.5)	0.94(6.0)	0.93(9.0)	0.94(6.0)	0.94(6.0)	0.95(2.0)
A0202	10	0.76(13.0)	0.83(8.0)	0.83(8.0)	0.85(2.0)	0.83(8.0)	0.83(8.0)
A0202	9	0.82(12.0)	0.88(6.0)	0.87(9.0)	0.88(6.0)	0.88(6.0)	0.87(9.0)
A0203	10	0.72(13.0)	0.79(9.5)	0.8(7.0)	0.81(4.5)	0.79(9.5)	0.8(7.0)
A0203	9	0.79(12.0)	0.87(9.0)	0.87(9.0)	0.89(4.0)	0.87(9.0)	0.88(6.0)
A0206	10	0.76(12.0)	0.83(5.5)	0.82(7.5)	0.84(4.0)	0.83(5.5)	0.81(9.5)
A0206	9	0.84(11.0)	0.9(6.5)	0.9(6.5)	0.9(6.5)	0.9(6.5)	0.9(6.5)
A0301	10	0.73(13.0)	0.81(9.0)	0.82(6.5)	0.83(4.5)	0.81(9.0)	0.82(6.5)
A0301	9	0.88(10.0)	0.89(8.0)	0.89(8.0)	0.9(6.0)	0.89(8.0)	0.91(3.5)
A1101	10	0.74(13.0)	0.82(9.5)	0.83(7.0)	0.86(2.5)	0.82(9.5)	0.83(7.0)
A1101	9	0.91(10.5)	0.94(6.0)	0.93(9.0)	0.94(6.0)	0.94(6.0)	0.95(2.0)
A2402	9	0.81(7.0)	0.82(3.5)	0.83(1.5)	0.83(1.5)	0.82(3.5)	0.78(11.0)
A2902	10	0.64(8.5)	0.67(7.0)	0.71(2.5)	0.72(1.0)	0.68(5.0)	0.61(11.5)
A2902	9	0.76(11.0)	0.79(4.5)	0.83(1.0)	0.81(2.0)	0.79(4.5)	0.69(13.0)
A3002	9	0.75(10.0)	0.77(6.0)	0.78(3.5)	0.79(2.0)	0.77(6.0)	0.64(13.0)
A3101	10	0.67(13.0)	0.81(7.5)	0.82(4.5)	0.82(4.5)	0.81(7.5)	0.8(10.5)
A3101	9	0.84(13.0)	0.91(7.5)	0.91(7.5)	0.92(3.0)	0.91(7.5)	0.91(7.5)
A3301	10	0.71(13.0)	0.79(11.0)	0.82(6.0)	0.85(3.0)	0.8(8.5)	0.73(12.0)
A3301	9	0.73(13.0)	0.88(8.0)	0.89(4.5)	0.89(4.5)	0.88(8.0)	0.89(4.5)
A6801	10	0.68(13.0)	0.79(9.5)	0.81(6.5)	0.83(4.0)	0.79(9.5)	0.81(6.5)
A6801	9	0.66(13.0)	0.82(9.5)	0.84(6.5)	0.86(3.5)	0.82(9.5)	0.84(6.5)
A6802	10	0.75(11.5)	0.84(6.5)	0.84(6.5)	0.84(6.5)	0.84(6.5)	0.83(9.0)
A6802	9	0.81(11.0)	0.89(8.0)	0.89(8.0)	0.9(5.5)	0.89(8.0)	0.91(2.5)
B0702	10	0.77(10.0)	0.79(6.5)	0.84(1.0)	0.8(3.5)	0.8(3.5)	0.73(11.5)
B0702	9	0.94(6.0)	0.93(9.5)	0.95(2.5)	0.94(6.0)	0.93(9.5)	0.9(12.0)
B1501	9	0.87(12.0)	0.89(8.0)	0.9(5.5)	0.91(3.5)	0.89(8.0)	0.88(10.5)
B3501	9	0.6(13.0)	0.72(7.5)	0.73(5.0)	0.77(1.0)	0.72(7.5)	0.68(11.0)
B4002	9	0.67(7.5)	0.67(7.5)	0.68(5.0)	0.72(1.0)	0.65(10.0)	0.68(5.0)
B4501	9	0.66(11.0)	0.7(9.0)	0.74(5.0)	0.76(3.0)	0.71(8.0)	0.61(12.0)
B5101	9	0.78(11.0)	0.82(7.5)	0.86(2.0)	0.86(2.0)	0.82(7.5)	0.71(13.0)
B5301	9	0.78(12.0)	0.8(11.0)	0.85(4.0)	0.87(1.0)	0.81(9.5)	0.68(13.0)
B5401	9	0.76(12.0)	0.79(11.0)	0.84(7.0)	0.87(1.0)	0.8(10.0)	0.71(13.0)
Avg		0.77(11.19)	0.83(7.75)	0.84(5.66)	0.85(3.57)	0.83(7.6)	0.81(8.63)

Table 2 (Continued)

Allele	Length	Boost	ADTree	SVM	SMM	LnrReg	M5P	GP
A0101	9	0.94(7.0)	0.93(10.5)	0.95(3.0)	0.68(13.0)	0.95(3.0)	0.93(10.5)	0.95(3.0)
A0201	10	0.86(11.0)	0.82(13.0)	0.9(3.5)	0.88(10.0)	0.9(3.5)	0.9(3.5)	0.91(1.0)
A0201	9	0.86(12.0)	0.85(13.0)	0.95(2.0)	0.92(10.5)	0.94(6.0)	0.95(2.0)	0.94(6.0)
A0202	10	0.81(11.0)	0.77(12.0)	0.84(4.5)	0.83(8.0)	0.85(2.0)	0.84(4.5)	0.85(2.0)
A0202	9	0.83(11.0)	0.79(13.0)	0.87(9.0)	0.89(4.0)	0.91(1.5)	0.9(3.0)	0.91(1.5)
A0203	10	0.77(11.0)	0.73(12.0)	0.81(4.5)	0.8(7.0)	0.83(1.5)	0.82(3.0)	0.83(1.5)
A0203	9	0.83(11.0)	0.78(13.0)	0.88(6.0)	0.88(6.0)	0.91(1.5)	0.9(3.0)	0.91(1.5)
A0206	10	0.77(11.0)	0.73(13.0)	0.82(7.5)	0.81(9.5)	0.86(2.0)	0.86(2.0)	0.86(2.0)
A0206	9	0.8(12.0)	0.78(13.0)	0.9(6.5)	0.89(10.0)	0.92(1.5)	0.91(3.0)	0.92(1.5)
A0301	10	0.76(11.0)	0.74(12.0)	0.83(4.5)	0.81(9.0)	0.85(2.0)	0.85(2.0)	0.85(2.0)
A0301	9	0.85(11.0)	0.84(12.5)	0.91(3.5)	0.84(12.5)	0.91(3.5)	0.92(1.0)	0.91(3.5)
A1101	10	0.8(11.0)	0.75(12.0)	0.84(5.0)	0.83(7.0)	0.86(2.5)	0.85(4.0)	0.87(1.0)
A1101	9	0.89(12.5)	0.89(12.5)	0.95(2.0)	0.91(10.5)	0.94(6.0)	0.95(2.0)	0.94(6.0)
A2402	9	0.79(10.0)	0.76(12.0)	0.81(7.0)	0.69(13.0)	0.81(7.0)	0.81(7.0)	0.81(7.0)
A2902	10	0.61(11.5)	0.64(8.5)	0.68(5.0)	0.62(10.0)	0.38(13.0)	0.71(2.5)	0.68(5.0)
A2902	9	0.79(4.5)	0.77(9.0)	0.79(4.5)	0.7(12.0)	0.77(9.0)	0.77(9.0)	0.78(7.0)
A3002	9	0.76(8.5)	0.76(8.5)	0.72(11.5)	0.72(11.5)	0.78(3.5)	0.81(1.0)	0.77(6.0)
A3101	10	0.8(10.5)	0.76(12.0)	0.81(7.5)	0.81(7.5)	0.85(2.0)	0.84(3.0)	0.86(1.0)
A3101	9	0.9(10.0)	0.88(11.0)	0.92(3.0)	0.87(12.0)	0.92(3.0)	0.92(3.0)	0.92(3.0)
A3301	10	0.83(5.0)	0.8(8.5)	0.8(8.5)	0.8(8.5)	0.85(3.0)	0.85(3.0)	0.86(1.0)
A3301	9	0.86(10.0)	0.83(12.0)	0.88(8.0)	0.84(11.0)	0.91(1.5)	0.89(4.5)	0.91(1.5)
A6801	10	0.77(11.0)	0.74(12.0)	0.82(5.0)	0.8(8.0)	0.84(2.5)	0.84(2.5)	0.85(1.0)
A6801	9	0.8(11.0)	0.76(12.0)	0.84(6.5)	0.84(6.5)	0.87(1.5)	0.86(3.5)	0.87(1.5)
A6802	10	0.75(11.5)	0.74(13.0)	0.85(3.0)	0.78(10.0)	0.85(3.0)	0.85(3.0)	0.86(1.0)
A6802	9	0.72(12.0)	0.71(13.0)	0.9(5.5)	0.85(10.0)	0.91(2.5)	0.91(2.5)	0.91(2.5)
B0702	10	0.72(13.0)	0.73(11.5)	0.79(6.5)	0.78(9.0)	0.79(6.5)	0.79(6.5)	0.81(2.0)
B0702	9	0.95(2.5)	0.93(9.5)	0.93(9.5)	0.88(13.0)	0.95(2.5)	0.94(6.0)	0.95(2.5)
B1501	9	0.89(8.0)	0.88(10.5)	0.9(5.5)	0.86(13.0)	0.92(1.5)	0.91(3.5)	0.92(1.5)
B3501	9	0.74(4.0)	0.66(12.0)	0.72(7.5)	0.7(10.0)	0.75(3.0)	0.72(7.5)	0.76(2.0)
B4002	9	0.68(5.0)	0.69(2.5)	0.69(2.5)	0.59(12.5)	0.62(11.0)	0.59(12.5)	0.66(9.0)
B4501	9	0.83(1.0)	0.72(7.0)	0.69(10.0)	0.79(2.0)	0.39(13.0)	0.73(6.0)	0.75(4.0)
B5101	9	0.86(2.0)	0.85(4.0)	0.8(10.0)	0.76(12.0)	0.82(7.5)	0.84(5.0)	0.82(7.5)
B5301	9	0.82(7.5)	0.85(4.0)	0.81(9.5)	0.83(6.0)	0.82(7.5)	0.86(2.0)	0.85(4.0)
B5401	9	0.85(5.5)	0.83(8.0)	0.82(9.0)	0.86(3.0)	0.85(5.5)	0.86(3.0)	0.86(3.0)
Avg		0.81(9.01)	0.79(10.65)	0.84(6.07)	0.80(9.34)	0.83(4.29)	0.85(4.12)	0.86(3.1)

Table 3 Results of pair-wise comparisons of the performance of different predictors using Nemenyi post-hoc test. x indicates that the performance of the corresponding pair of methods is significant at 0.05.

	PSSM	MPSSM	AOMM	SMMBin	NB	LR	Boost	ADTree	SVM	SMM	LnrReg	M5P
PSSM												
MPSSM	x											
AOMM	x	0										
SMMBin	x	x	0									
NB	x	0	0	x								
LR	0	0	0	x	0							
Boost	0	0	x	x	0	0						
ADTree	0	0	x	x	0	0	0					
SVM	x	0	0	0	0	0	0	x				
SMM	0	0	x	x	0	0	0	0	x			
LnrReg	x	x	0	0	x	x	x	x	0	x		
M5P	x	x	0	0	x	x	x	x	0	x	0	
GP	x	x	0	0	x	x	x	x	0	x	0	0

Table 4 Pearson correlation coefficient (PCC) values for different methods evaluated on the similarity-reduced MHC-I data set. For each data set, the rank of each classifier is shown in parentheses.

Allele	Length	SMM	LnrReg	M5P	GP
A0101	9	0.22(4.0)	0.61(2.5)	0.74(1.0)	0.61(2.5)
A0201	10	0.71(4.0)	0.74(1.5)	0.73(3.0)	0.74(1.5)
A0201	9	0.74(4.0)	0.78(3.0)	0.82(1.0)	0.79(2.0)
A0202	10	0.61(4.0)	0.68(1.5)	0.66(3.0)	0.68(1.5)
A0202	9	0.74(4.0)	0.77(2.5)	0.77(2.5)	0.78(1.0)
A0203	10	0.64(4.0)	0.67(2.0)	0.66(3.0)	0.68(1.0)
A0203	9	0.72(4.0)	0.77(1.5)	0.76(3.0)	0.77(1.5)
A0206	10	0.6(4.0)	0.67(2.5)	0.67(2.5)	0.68(1.0)
A0206	9	0.73(4.0)	0.78(1.5)	0.77(3.0)	0.78(1.5)
A0301	10	0.65(4.0)	0.69(1.5)	0.67(3.0)	0.69(1.5)
A0301	9	0.63(4.0)	0.72(2.5)	0.77(1.0)	0.72(2.5)
A1101	10	0.68(4.0)	0.72(2.0)	0.7(3.0)	0.73(1.0)
A1101	9	0.75(4.0)	0.79(2.5)	0.84(1.0)	0.79(2.5)
A2402	9	0.38(4.0)	0.54(1.5)	0.53(3.0)	0.54(1.5)
A2902	10	0.32(3.0)	-0.19(0.0)	0.42(1.0)	0.4(2.0)
A2902	9	0.44(4.0)	0.49(2.5)	0.5(1.0)	0.49(2.5)
A3002	9	0.49(4.0)	0.55(2.0)	0.62(1.0)	0.54(3.0)
A3101	10	0.66(4.0)	0.7(2.0)	0.69(3.0)	0.72(1.0)
A3101	9	0.66(4.0)	0.75(2.5)	0.79(1.0)	0.75(2.5)
A3301	10	0.63(4.0)	0.68(2.0)	0.67(3.0)	0.69(1.0)
A3301	9	0.6(4.0)	0.68(2.0)	0.66(3.0)	0.69(1.0)
A6801	10	0.64(4.0)	0.69(2.0)	0.69(2.0)	0.69(2.0)
A6801	9	0.69(4.0)	0.73(2.5)	0.73(2.5)	0.74(1.0)
A6802	10	0.56(4.0)	0.64(2.0)	0.63(3.0)	0.65(1.0)
A6802	9	0.64(4.0)	0.72(3.0)	0.75(1.0)	0.73(2.0)
B0702	10	0.61(1.5)	0.58(3.0)	0.54(4.0)	0.61(1.5)
B0702	9	0.61(4.0)	0.74(2.5)	0.77(1.0)	0.74(2.5)
B1501	9	0.61(4.0)	0.69(2.5)	0.71(1.0)	0.69(2.5)
B3501	9	0.49(4.0)	0.53(2.0)	0.5(3.0)	0.55(1.0)
B4002	9	0.28(2.0)	0.26(3.0)	0.24(4.0)	0.38(1.0)
B4501	9	0.69(1.0)	0.17(4.0)	0.55(3.0)	0.59(2.0)
B5101	9	0.48(4.0)	0.57(2.5)	0.6(1.0)	0.57(2.5)
B5301	9	0.65(2.0)	0.6(4.0)	0.7(1.0)	0.63(3.0)
B5401	9	0.69(2.0)	0.66(4.0)	0.7(1.0)	0.68(3.0)
Avg		0.60(3.63)	0.62(2.31)	0.66(2.16)	0.66(1.78)

CHAPTER 5. ON EVALUATING MHC-II BINDING PEPTIDE PREDICTION METHODS

A paper published in the Journal of PLoS ONE

Yasser EL-Manzalawy, Drena Dobbs, Vasant Honavar

Abstract

Choice of one method over another for MHC-II binding peptide prediction is typically based on published reports of their estimated performance on standard benchmark datasets. We show that several standard benchmark datasets of unique peptides used in such studies contain a substantial number of peptides that share a high degree of sequence identity with one or more other peptide sequences in the same dataset. Thus, in a standard cross-validation setup, the test set and the training set are likely to contain sequences that share a high degree of sequence identity with each other, leading to overly optimistic estimates of performance. Hence, to more rigorously assess the relative performance of different prediction methods, we explore the use of similarity-reduced datasets. We introduce three similarity-reduced MHC-II benchmark datasets derived from MHCPEP, MHCBN, and IEDB databases. The results of our comparison of the performance of three MHC-II binding peptide prediction methods estimated using datasets of unique peptides with that obtained using their similarity-reduced counterparts show that the former can be rather optimistic relative to the performance of the same methods on similarity-reduced counterparts of the same datasets. Furthermore, our results demonstrate that conclusions regarding the superiority of one method over another drawn on the basis of performance estimates obtained using commonly used datasets of unique peptides are often

contradicted by the observed performance of the methods on the similarity-reduced versions of the same datasets. These results underscore the importance of using similarity-reduced datasets in rigorously comparing the performance of alternative MHC-II peptide prediction methods.

Introduction

T-cells epitopes are short linear peptides generated by cleavage of antigenic proteins. The identification of T-cell epitopes in protein sequences is important for understanding disease pathogenesis, identifying potential autoantigens, and designing vaccines and immune-based cancer therapies. A major step in identifying potential T-cell epitopes involves identifying the peptides that bind to a target major histocompatibility complex (MHC) molecule. Because of the high cost of experimental identification of such peptides, there is an urgent need for reliable computational methods for predicting MHC binding peptides (Korber et al., 2006).

There are two major classes of MHC molecules: MHC class I (MHC-I) molecules characterized by short binding peptides, usually consisting of nine residues; and MHC class II (MHC-II) molecules with binding peptides that range from 11 to 30 residues in length, although shorter and longer peptide lengths are not uncommon (Rammensee et al., 1995). The binding groove of MHC-II molecules is open at both ends, allowing peptides longer than 9-mers to bind. However, it has been reported that a 9-mer core region is essential for MHC-II binding (Rammensee et al., 1995; Madden, 1995). Because the precise location of the 9-mer core region of MHC-II binding peptides is unknown, predicting MHC-II binding peptides tends to be more challenging than predicting MHC-I binding peptides.

Despite the high degree of variability in the length of MHC-II binding peptides, most existing computational methods for predicting MHC-II binding peptides focus on identifying a 9-mer core peptide. Computational approaches available for predicting MHC-II binding peptides from amino acid sequences include: (i) Motif-based methods such as methods that use a position weight matrix (PWM) to model an ungapped multiple sequence alignment of MHC binding peptides (Singh and Raghava, 2001; Reche et al., 2004; Nielsen et al., 2004, 2007; Rajapakse et al., 2007), and a statistical approach based on Hidden Markov Models (HMMs) (Mamitsuka, 1998; Noguchi et al., 2002); (ii) Machine learning methods based on Artificial Neural Networks (ANN) (Nielsen et al., 2004, 2003; Buus et al., 2003; Burden and Winkler, 2006) and Support Vector Machines (SVMs) (Donnes and Kohlbacher, 2006; Bhasin and Raghava, 2004; Cui et al., 2006a; Salomon and Flower, 2006); (iii) Semi-supervised machine learning methods (Murugan and Dai, 2005; Hertz and Yanover, 2006).

The choice of one method over another for MHC-II binding peptide prediction requires reliable assessment of their performance relative to each other. Such assessments usually rely on estimates of their performance on standard benchmark datasets (typically obtained using cross-validation). Several studies (Reche et al., 2004; Bhasin and Raghava, 2004; Cui et al., 2006a; Salomon and Flower, 2006; Hertz and Yanover, 2006) have reported the performance of MHC-II binding peptide prediction methods using datasets of unique peptides. Such datasets can in fact contain peptide sequences that share a high degree of sequence similarity with other peptide sequences in the dataset. Hence, several authors (Nielsen et al., 2004, 2007; Noguchi et al., 2002; Raghava, 2004) have proposed methods for eliminating redundant sequences. However, because MHC-II peptides have lengths that vary over a broad range, similarity reduction of MHC-II peptides is not a straightforward task (Nielsen et al., 2007). Consequently, standard cross-validation based estimates of performance obtained using such datasets are likely to be overly optimistic because the test set is likely to contain sequences that share significant sequence similarity with one or more sequences in the training set.

In order to obtain more realistic estimates of performance of MHC-II binding peptide prediction methods, we explored several methods for constructing similarity-reduced MHC-II datasets. We constructed similarity-reduced MHC-II benchmark datasets, derived from MHCPEP (Brusic et al., 1998), MHCBN (Bhasin et al., 2003), and IEDB (Peters et al., 2005) databases, using several approaches to reduce the degree of pair-wise sequence similarity shared by sequences in the resulting datasets. The similarity reduction procedures were applied separately to binders and non-binders. Details of the similarity reduction methods are provided in the Materials and Methods Section. Specifically, we generated:

- Datasets of unique peptides MHCPEP-UPDS, MHCBN-UPDS, and IEDB-UPDS extracted from MHCPEP, MHCBN, and IEDB, respectively.

- Datasets of similarity-reduced peptides, MHCPEP-SRDS1, MHCBN-SRDS1, and IEDB-SRDS1 derived from the corresponding UPDS datasets using a similarity reduction procedure which ensures that no two peptides in the resulting dataset share a 9-mer subsequence.

- Datasets of similarity-reduced peptides, MHCPEP-SRDS2, MHCBN-SRDS2, and IEDB-SRDS2, extracted MHCPEP-SRDS1, MHCBN-SRDS1, and IEDB-SRDS1 respectively by filtering the binders and non-binders in SRDS1 such that the sequence identity between any pair of peptides is less than 80

- Datasets of similarity-reduced peptides, MHCPEP-SRDS3, MHCBN-SRDS3, and IEDB-SRDS3, derived from the corresponding UPDS datasets using the similarity reduction procedure introduced by Raghava and previously used to construct the MHCBench dataset (Raghava, 2004).

- Datasets of weighted unique peptides, MHCPEP-WUPDS, MHCBN-WUPDS, and IEDB-WUPDS, derived from the corresponding UPDS datasets (where the weight assigned to a peptide is inversely proportional to the number of peptides that are similar to it).

We then used the resulting similarity-reduced benchmark datasets to explore the effect of similarity reduction on the performance of different MHC-II binding peptide prediction methods and, more importantly, to rigorously compare the performance of the different prediction methods. Our experiments focused on two state-of-the-art methods for training MHC-II binding peptide predictors using variable-length MHC-II peptides and a third method that is designed to exploit the sequence similarity between a test peptide sequence and the peptide sequences in the training set (and is hence likely to perform well on non similarity-reduced datasets but poorly on the similarity-reduced datasets).

Specifically, we compared: (i) An approach (Cui et al., 2006a) that maps each variable-length peptide into a fixed-length feature vector (the so-called composition-transition distribution or CTD) consisting of sequence-derived structural features and physicochemical properties of the input peptide sequence; (ii) An approach (Salomon and Flower, 2006) that uses a local alignment (LA) kernel that defines the similarity between two variable-length peptides as the average of all possible local alignments between the two peptides; (iii) An approach that uses the k-spectrum kernel (Leslie et al., 2002) with $k = 5$.

Because neither the programs used to calculate secondary structure and solvent accessibility of peptides used for generating the CTD representation (Cui et al., 2006a) nor the precise choices of parameters used for training the LA kernel based classifier (Salomon and Flower, 2006) were available to us, we used in our experiments, our own implementations of the corresponding methods. Hence, the results of our experiments should not be viewed as providing direct assessment of performance of the exact implementations of the CTD and LA methods developed by the original authors and used in studies reported in (Cui et al., 2006a; Salomon and Flower, 2006). However, it is worth noting that, the broad conclusions of our study are largely independent of the specific machine learning methods or data transformations.

Our results demonstrate that, regardless of the similarity reduction method employed, a substantial drop in performance of classifiers is observed compared to their reported performance on benchmark datasets of unique peptide sequences. Our results also demonstrate that conclusions regarding the superiority of one prediction method over another can be misleading when they are based on evaluations using benchmark datasets with a high degree of sequence similarity (e.g., the benchmark dataset of unique peptide sequences). These results underscore the importance of using similarity-reduced datasets in evaluating and comparing alternative MHC-II peptide prediction methods.

Results

Limitations of the unique peptides MHC-II data

Tables 1-3 show that MHC-II datasets derived from MHCPEP, MHCBN, and IEDB databases have a large number of highly similar peptides: the number of peptides in the similarity-reduced versions in the three benchmark datasets is $\approx 50\%$of the original number. In each case, the estimated performance of the prediction methods evaluated on similarity-reduced datasets is substantially worse than that estimated using the datasets of unique peptides. This finding is especially significant in light of the fact that MHCPEP and MHCBN datasets have been used for comparing alternative MHC-II peptide prediction methods in most of the published studies (Reche et al., 2004; Nielsen et al., 2004; Bhasin and Raghava, 2004; Cui et al., 2006a; Salomon and Flower, 2006; Murugan and Dai, 2005; Hertz and Yanover, 2006; Yu et al., 2006b).

For the sake of brevity, we focus discussion here on the results of two representative examples of datasets extracted from the MHCPEP and MHCBN benchmarks and provide the complete set of results in the supplementary materials (Data S1).

As shown in Table 4, for the MHCPEP benchmark, we focus on the results on the data for HLA-DR4, which has the largest number of unique binders. On the MHCPEP-UPDS version of the HLA-DR4 dataset, the 5-spectrum kernel outperforms the other two prediction methods and CTD outperforms the LA kernel. We notice a substantial drop in the observed performance of the three prediction methods on the similarity-reduced and weighted datasets relative to that on their UPDS counterpart.

In the case of the MHCBN benchmark, we focus on the results on the HLA-DRB1*0301 data (Table 5) because it has been used in a number of recent studies of MHC-II binding peptide prediction methods (Cui et al., 2006a; Salomon and Flower, 2006; Yu et al., 2006b). Most MHCBN allele-specific datasets are unbalanced, i.e., the numbers of binding peptides in the datasets are larger (typically by a factor of 2 to 4) than the corresponding numbers of non-binding peptides (see Table 2). On such unbalanced datasets, classification accuracy can be misleading in terms of providing a reliable and useful assessment of the performance of the classifier. A classifier that simply returns the label of the majority class as the predicted label for each instance to be classified can achieve a rather high accuracy. However such a classifier is rather useless in reliably identifying members of the minority class. Hence, in the case of unbalanced datasets, the correlation coefficient (CC) or the area under the Receiver Operating Characteristic (ROC) curve (AUC) provide more useful measures than accuracy in assessing the performance of the classifiers (Baldi et al., 2000). As shown in Table 5, the observed performance of the three prediction methods on HLA-DRB1*0301 MHCBN-UPDS version of this dataset appears to be overly optimistic relative to that on its similarity-reduced and weighted counterparts. Interestingly, the 5-spectrum kernel is competitive with CTD and LA on the MHCBN-UPDS dataset, whereas its performance on MHCBN-SRDS1 and MHCBN-SRDS2 is much worse than that of the CTD and the LA classifiers.

Our results also demonstrate that conclusions of superior performance of one method relative to another that are based on estimates of performance obtained using UPDS versions of MHC-II benchmark datasets can be misleading. For example, from results shown in Tables 4 and 5, one might be tempted to conclude that predictors that use the 5-spectrum kernel are competitive with those that use CTD representation and the LA kernel. However, the 5-spectrum kernel is outperformed by CTD and LA on the similarity-reduced datasets. Similarly, conclusions drawn from experiments using the UPDS datasets (Tables 4 and 5) regarding the performance of the CTD and the LA kernel classifiers are contradicted by the their observed performance on the corresponding similarity-reduced datasets SRDS1 and SRDS2.

Limitations of the MHCBench benchmark data

Comparison of SRDS1, SRDS2, and SRDS3 versions of the datasets used in this study reveals an important limitation of the MHCBench dataset which is a widely used benchmark for comparing MHC-II binding peptide prediction methods.

Recall that the SRDS3 versions of our datasets are derived using the same procedure that was used in MHCBench to generate similarity-reduced datasets. It is clear from the data summarized in Tables 1-3 that the size of a SRDS3 version of a dataset is: often larger than the size of its SRDS2 counterpart, and sometimes larger than the size of its SRDS1 counterpart. Closer examination of the peptides in SRDS3 datasets reveals that SRDS3 datasets may contain several highly similar peptides (e.g., peptides with more than 80% sequence similarity). This is illustrated by the example shown in Figure 1: the two peptides in the SRDS3 version of the HLA-DRB1*0301 dataset share overall sequence similarity of 85.71%. However, the procedure

Table 1 Number of binding peptides in MHCPEP benchmark dataset. UPDS refers to datasets of non-redundant peptides. The last three columns refer to similarity-reduced datasets (see text for details).

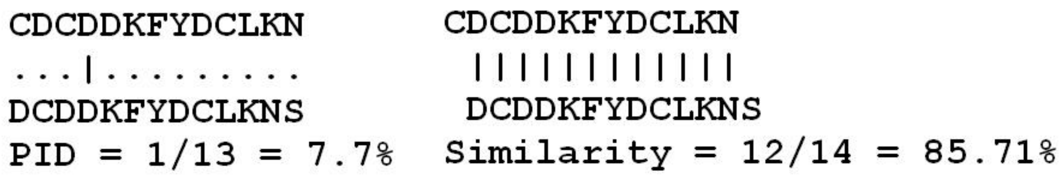
Allele	UPDS	SRDS1	SRDS2	SRDS3
HLA-DQ2	113	67	32	39
HLA-DQ4	97	84	79	82
HLA-DQ7	135	73	65	75
HLA-DR1	703	336	242	278
HLA-DR2	315	148	104	134
HLA-DR3	192	81	69	73
HLA-DR4	1085	439	298	353
HLA-DR5	189	92	61	75
HLA-DR7	341	137	87	101
HLA-DR8	125	47	46	54
HLA-DR9	94	41	34	37
HLA-DR11	473	160	100	103
HLA-DR13	121	68	34	36
HLA-DR15	121	48	36	49
HLA-DR17	158	82	40	45
HLA-DR51	115	45	39	55
I-Ab	136	62	51	61
I-Ad	415	168	101	135
I-Ag7	157	62	53	81
I-Ak	254	96	67	85
I-Ed	294	188	68	76
I-Ek	334	204	64	78

used to construct similarity-reduced MHCBench dataset will keep both of these peptides in the resulting dataset because the computed percent identity (PID) between the two peptides is only 7.7%, well below the threshold of 80% PID used to identify similar peptides in MHCBench (Raghava, 2004). Thus, the similarity reduction procedure used in MHCBench dataset (which relies on a strict gapless alignment) may not eliminate all highly similar peptides.

The preceding observation explains why the number of peptides in the SRDS3 versions of the datasets is usually greater than that in SRDS1 and SRDS2 datasets (see Tables 1-3). More importantly, because of the presence of a number of highly similar peptides in some SRDS3 datasets, the observed performance of the three prediction methods on the SRDS3 datasets may be overly optimistic relative to that estimated from their SRDS1 and SRDS2

Table 2 Number of binding/non-binding peptides in MHCBN benchmark dataset. UPDS refers to datasets of non-redundant peptides. The last three columns refer to similarity-reduced datasets (see text for details).

Allele	UPDS	SRDS1	SRDS2	SRDS3
HLA-DR1	636/180	328/111	223/106	259/130
HLA-DR2	416/168	197/124	153/123	232/149
HLA-DR5	218/173	111/131	80/129	100/154
HLA-DRB10101	531/127	325/88	279/76	390/112
HLA-DRB10301	261/230	137/150	127/145	175/215
HLA-DRB10401	805/201	471/136	404/119	543/174
HLA-DRB10701	292/107	179/68	152/66	213/92
HLA-DRB11101	352/137	213/87	182/87	239/131



```
CDCDDKFYDCLKN          CDCDDKFYDCLKN
...|.........           |||||||||||| 
DCDDKFYDCLKNS           DCDDKFYDCLKNS
PID = 1/13 = 7.7%      Similarity = 12/14 = 85.71%
```

Figure 1 Example of two peptides from MHCBN-SRDS3 HLA-DRB1*0301 dataset. Although the two peptides share 85.71% sequence similarity, the computed percent identity (PID) used to define the similarity between these two peptides in MHCBench benchmark is only 7.7%.

counterparts. Because the classifier using the 5-spectrum kernel in fact relies on the degree of (gapless) match between a sequence pattern present in one or more training sequences and a test sequence, it benefits from the presence of a high degree of similarity between a test sequence and one or more training sequences in ways that the other two classifiers do not. Consequently classifiers that use the 5-spectrum kernel can appear to be competitive with, and perhaps even outperform those that use the CTD representation or the LA kernel when their performance is compared using SRDS3 datasets (and for similar reasons, the MHCBench benchmark data).

Table 3 Number of binding/non-binding peptides in IEDB benchmark dataset. UPDS refers to datasets of non-redundant peptides. The last three columns refer to similarity-reduced datasets (see text for details).

Allele	UPDS	SRDS1	SRDS2	SRDS3
HLA-DRB1-0101	1105/432	645/268	623/261	938/365
HLA-DRB1-0301	135/556	78/292	69/276	81/396
HLA-DRB1-0401	317/412	197/262	176/255	215/340
HLA-DRB1-0404	113/132	69/100	62/98	74/109
HLA-DRB1-0405	113/119	74/85	70/84	81/89
HLA-DRB1-0701	228/302	147/203	137/202	173/274
HLA-DRB1-0802	65/120	46/101	46/100	49/108
HLA-DRB1-1101	197/411	122/218	111/212	139/328
HLA-DRB1-1302	152/103	105/81	97/81	110/92
HLA-DRB1-1501	269/283	165/176	142/174	185/260
HLA-DRB4-0101	92/215	64/120	63/119	85/200
HLA-DRB5-0101	215/377	123/201	113/194	147/309

Comparison of the CTD, LA, and the k-spectrum kernel methods

In machine learning and bioinformatics literature, claims of superiority of one method over another are often based on the outcome of suitable statistical tests. Hence, it is interesting to examine the differences in the conclusions obtained when statistical tests are used to compare the performance of prediction methods based on the empirical estimates of their performance on the UPDS, SRDS1, SRDS2, SRDS3, and WPDS versions of the datasets.

Several non-parametric statistical tests (Friedman, 1940; Fisher, 1973) have been recently recommended for comparing different classifiers on multiple datasets (accounting for the effects of multiple comparisons) (Demšar, 2006). In our analysis, we apply a three-step procedure proposed by Demšar (2006). First, the classifiers to be compared are ranked on the basis of their observed performance (e.g., AUC) on each dataset. Second, the Friedman test is applied to determine whether the measured average ranks are significantly different from the mean rank under the null hypothesis. Third, if the null hypothesis can be rejected at a significance level of 0.05, the Nemenyi test is used to determine whether significant differences exist between any given pair of classifiers.

Table 4 Performance of prediction methods on MHCPEP HLA-DR4 unique, similarity-reduced, and weighted datasets using 5-fold cross-validation test.

Dataset	Method	ACC	Sn	Sp	CC	AUC
UPDS	CTD	86.59	73.36	99.82	0.759	0.906
	LA	77.10	71.71	82.49	0.545	0.862
	5-spectrum	90.55	81.29	99.82	0.825	0.917
SRDS1	CTD	69.93	66.06	73.80	0.400	0.723
	LA	68.56	63.78	73.35	0.373	0.751
	5-spectrum	70.96	43.28	98.63	0.503	0.710
SRDS2	CTD	64.77	60.40	69.13	0.296	0.692
	LA	64.43	65.10	63.76	0.289	0.711
	5-spectrum	56.04	33.22	78.86	0.136	0.578
SRDS3	CTD	65.01	61.47	68.56	0.301	0.695
	LA	64.02	62.89	65.16	0.281	0.717
	5-spectrum	68.56	38.81	98.30	0.462	0.679
WUPDS	CTD	85.41	31.98	99.91	0.516	0.730
	LA	79.38	22.50	94.81	0.249	0.723
	5-spectrum	87.14	42.14	99.35	0.580	0.723

Statistical analysis of results on the MHCPEP datasets

Tables 6-10 compare the AUC of the three prediction methods on the five versions of the MHCPEP datasets. For each dataset, the rank of each classifier is shown in parentheses. The last row in each table summarizes the average AUC and rank for each classifier. Demšar (2006) has suggested that the average ranks by themselves provide a reasonably fair comparison of classifiers. Interestingly, the LA kernel has the worst rank among the three methods when the comparison is based on the observed performance on the UPDS datasets, whereas it has the best rank among the three methods when the comparison is based on the similarity-reduced or the weighted datasets. Tables 6-10 also show that the rank of the 5-spectrum kernel is competitive with that of CTD on UPDS and SRDS3. This observation is consistent with the presence of a number of highly similar sequences in SRDS3 datasets.

To determine whether the differences in average ranks are statistically significant, we applied the Friedman test (Demšar, 2006) to the rank data in Tables 6-10. At significance level of 0.05, the Friedman test did not indicate a statistically significant difference between the

Table 5 Performance of prediction methods on MHCBN HLA-DRB1*0301 unique, similarity-reduced, and weighted datasets using 5-fold cross-validation test.

Dataset	Method	ACC	Sn	Sp	CC	AUC
UPDS	CTD	72.51	73.95	70.87	0.448	0.787
	LA	71.89	73.56	70.00	0.436	0.795
	5-spectrum	70.26	82.76	56.09	0.405	0.770
SRDS1	CTD	63.41	64.23	62.67	0.269	0.661
	LA	58.54	59.85	57.33	0.172	0.617
	5-spectrum	42.16	63.50	22.67	-0.152	0.323
SRDS2	CTD	59.93	59.06	60.69	0.197	0.628
	LA	55.88	54.33	57.24	0.116	0.563
	5-spectrum	35.29	37.01	33.79	-0.292	0.273
SRDS3	CTD	64.62	60.57	67.91	0.285	0.675
	LA	67.18	61.14	72.09	0.334	0.736
	5-spectrum	63.08	49.71	73.95	0.244	0.678
WUPDS	CTD	65.27	61.04	68.88	0.300	0.678
	LA	66.66	64.47	68.53	0.330	0.710
	5-spectrum	59.97	58.70	61.04	0.197	0.648

methods on the UPDS and WUPDS datasets. However, in the case of the similarity-reduced datasets, the Friedman test indicated statistically significant differences between the methods being compared. Thus, we conclude that the three methods are competitive with each other on the UPDS and WUPDS datasets, and that there is at least one pair of classifiers with significant difference in performance on the three versions of similarity-reduced datasets. Furthermore, for each version of MHCPEP similarity-reduced datasets, the Nemenyi test was applied to determine whether significant differences exist between any given pair of classifiers. Figure 2 summarizes the results of the pair-wise comparisons performed using the Nemenyi test. We find that on the SRDS1 versions of the datasets, both the LA and the CTD methods significantly outperform the 5-spectrum kernel and that there are no statistically significant differences between the LA kernel and the CTD classifier. On SRDS2 datasets, we find that, the performance of each of the three methods is significantly different from that of the other two methods, with the LA and the CTD methods ranked first and second, respectively. On SRDS3 datasets, we observe that the performance of the LA kernel is significantly better than that of the CTD and the 5-spectrum classifiers, with no significant differences between the

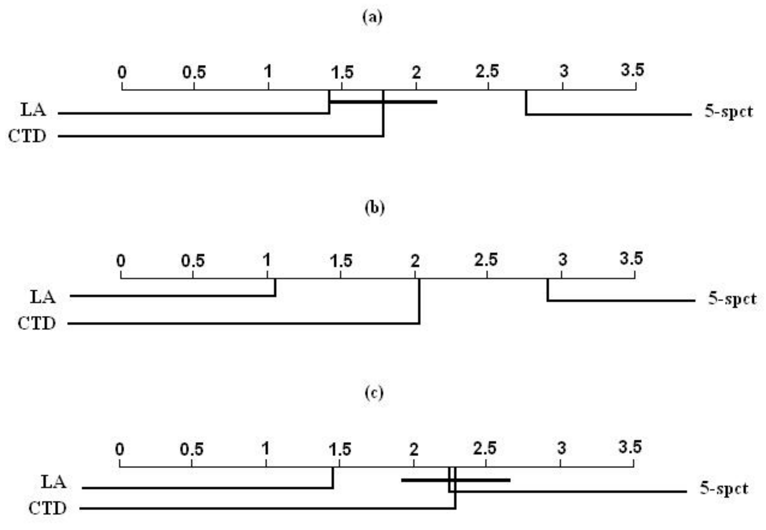


Figure 2 Pair-wise comparisons of classifiers with the Nemenyi test applied to results on a) MHCPEP-SRDS1, b) MHCPEP-SRDS2, and c) MHCPEP-SRDS3. Classifiers that are not significantly different (at p-value = 0.05) are connected.

CTD and the 5-spectrum classifiers.

Statistical analysis of results on the MHCBN and the IEDB datasets

We summarize the results of applying Demar's three-step procedure to the results obtained on the five versions of MHCBN and IEDB datasets, respectively. In the case of the MHCBN datasets, Tables 11-15 show the estimated AUC and rank of each classifier on each dataset. The results of the Freidman test (at a significance level of 0.05) applied to the results in each table did not indicate significant differences in performance among the CTD, the LA, and the 5-spectrum kernel classifiers on the UPDS dataset. However, the test indicated statistically significant differences among the methods in the case of the SRDS1, SRDS2, SRDS3, and the WUPDS datasets. Figure 3 summarizes the results of the pair-wise comparisons using the Nemenyi test. In the case of the SRDS1 and the SRDS2 datasets, we find that the performance of both the LA kernel and the CTD classifiers is significantly better than that of the 5-spectrum kernel classifier and that there are no significant differences between the LA kernel and the CTD classifiers. In the case of the SRDS3 datasets, we find that the performance of the LA

Table 6 AUC values for the three methods evaluated on MHCPEP-UPDS datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DQ2	0.908(2)	0.905(3)	0.939(1)
HLA-DQ4	0.628(3)	0.903(2)	0.934(1)
HLA-DQ7	0.856(2)	0.860(1)	0.853(3)
HLA-DR1	0.883(1)	0.872(2)	0.863(3)
HLA-DR2	0.884(1)	0.866(2)	0.829(3)
HLA-DR3	0.854(3)	0.869(1)	0.862(2)
HLA-DR4	0.917(1)	0.862(3)	0.906(2)
HLA-DR5	0.905(1)	0.864(3)	0.887(2)
HLA-DR7	0.916(1)	0.858(3)	0.904(2)
HLA-DR8	0.894(3)	0.896(2)	0.903(1)
HLA-DR9	0.836(3)	0.880(2)	0.913(1)
HLA-DR11	0.910(3)	0.938(2)	0.958(1)
HLA-DR13	0.875(3)	0.905(2)	0.920(1)
HLA-DR15	0.887(1)	0.829(3)	0.867(2)
HLA-DR17	0.907(2)	0.907(2)	0.934(1)
HLA-DR51	0.924(1)	0.891(2)	0.886(3)
I-Ab	0.865(2)	0.855(3)	0.875(1)
I-Ad	0.942(1)	0.898(3)	0.902(2)
I-Ag7	0.916(1)	0.896(2)	0.887(3)
I-Ak	0.909(1)	0.872(3)	0.881(2)
I-Ed	0.918(3)	0.921(2)	0.936(1)
I-Ek	0.934(3)	0.940(2)	0.951(1)
Average	0.885(1.91)	0.886(2.27)	0.900(1.77)

kernel classifier is significantly better than that of the CTD and the 5-spectrum classifiers, and that no significant differences exist between the CTD and the 5-spectrum classifiers. In the case of the WUPDS datasets, we find that the LA kernel classifier significantly outperforms the 5-spectrum kernel and that there are no significant differences between the LA and the CTD and between the CTD and the 5-spectrum classifiers.

Results of Demšar's statistical test applied to the IEDB datasets are shown in Tables S46-S50 (Data S1 in supporting information) and Figure 4. As in the case of MHCPEP and MHCBN, we see no significant differences in the performance of different classifiers on IEDB-UPDS datasets. However, in the case of the other datasets, we find at least one pair

Table 7 AUC values for the three methods evaluated on MHCPEP-SRDS1 datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DQ2	0.789(3)	0.852(2)	0.853(1)
HLA-DQ4	0.544(3)	0.854(2)	0.881(1)
HLA-DQ7	0.677(3)	0.799(1)	0.726(2)
HLA-DR1	0.662(3)	0.801(1)	0.744(2)
HLA-DR2	0.694(3)	0.795(1)	0.781(2)
HLA-DR3	0.603(2)	0.678(1)	0.572(3)
HLA-DR4	0.710(3)	0.751(1)	0.723(2)
HLA-DR5	0.691(3)	0.776(2)	0.784(1)
HLA-DR7	0.721(2)	0.702(3)	0.732(1)
HLA-DR8	0.552(3)	0.625(2)	0.694(1)
HLA-DR9	0.620(3)	0.746(1)	0.721(2)
HLA-DR11	0.703(3)	0.912(1)	0.890(2)
HLA-DR13	0.746(3)	0.827(2)	0.837(1)
HLA-DR15	0.711(2)	0.718(1)	0.667(3)
HLA-DR17	0.789(3)	0.806(2)	0.876(1)
HLA-DR51	0.651(2)	0.788(1)	0.603(3)
I-Ab	0.620(3)	0.705(1)	0.680(2)
I-Ad	0.787(3)	0.818(1)	0.804(2)
I-Ag7	0.718(2)	0.778(1)	0.702(3)
I-Ak	0.761(3)	0.800(1)	0.796(2)
I-Ed	0.826(3)	0.903(2)	0.932(1)
I-Ek	0.874(3)	0.913(2)	0.941(1)
Average	0.702(2.77)	0.789(1.45)	0.770(1.77)

of classifiers with significant differences in performance. As shown in Figure 4, both the LA and the CTD classifiers significantly outperform the 5-spectrum classifier on the SRDS1 and the SRDS2 versions of the IEDB datasets. However, no significant differences are observed between the CTD and the 5-spectrum methods on the SRDS3 and WUPDS versions of the IEDB datasets.

Performance on the blind test set

The results summarized above underscore the importance of similarity-reduced MHC-II datasets for obtaining a realistic estimation of the classifier performance and avoiding mis-

Table 8 AUC values for the three methods evaluated on MHCPEP-SRDS2 datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DQ2	0.566(3)	0.678(1)	0.573(2)
HLA-DQ4	0.590(3)	0.954(1)	0.817(2)
HLA-DQ7	0.616(3)	0.713(1)	0.709(2)
HLA-DR1	0.562(3)	0.715(1)	0.711(2)
HLA-DR2	0.548(3)	0.633(1)	0.614(2)
HLA-DR3	0.514(3)	0.602(1)	0.572(2)
HLA-DR4	0.578(3)	0.711(1)	0.692(2)
HLA-DR5	0.583(3)	0.622(2)	0.625(1)
HLA-DR7	0.562(3)	0.622(1)	0.599(2)
HLA-DR8	0.526(3)	0.717(1)	0.680(2)
HLA-DR9	0.488(3)	0.754(1)	0.690(2)
HLA-DR11	0.528(3)	0.810(1)	0.792(2)
HLA-DR13	0.518(3)	0.827(1)	0.587(2)
HLA-DR15	0.592(3)	0.698(1)	0.689(2)
HLA-DR17	0.568(2)	0.612(1)	0.550(3)
HLA-DR51	0.578(3)	0.664(1)	0.595(2)
I-Ab	0.570(3)	0.624(2)	0.638(1)
I-Ad	0.623(2)	0.700(1)	0.618(3)
I-Ag7	0.713(2)	0.756(1)	0.632(3)
I-Ak	0.586(3)	0.664(1)	0.661(2)
I-Ed	0.645(3)	0.760(1)	0.744(2)
I-Ek	0.606(3)	0.756(1)	0.703(2)
Average	0.575(2.86)	0.709(1.09)	0.659(2.05)

leading conclusions. However, one might argue that in practice, when developers of MHC-II binding peptide prediction methods make an implementation of their methods publicly available (e.g., as an online web server or as a web service), it might be better to utilize as much of the available data as possible to train the predictor. Hence, it is interesting to explore whether the UPDS datasets should be preferred over the similarity-reduced counterparts to avoid any potential loss of useful information due to the elimination of highly similar peptides in a setting where the goal is to optimize the predictive performance of the classifier on novel peptides. In what follows, we attempt to answer this question using five allele-specific blind test sets (Wang et al., 2008) to evaluate the performance of the three prediction methods trained on the unique,

Table 9 AUC values for the three methods evaluated on MHCPEP-SRDS3 datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DQ2	0.663(2)	0.655(3)	0.754(1)
HLA-DQ4	0.608(3)	0.900(1)	0.900(1)
HLA-DQ7	0.699(3)	0.757(1)	0.706(2)
HLA-DR1	0.676(3)	0.747(1)	0.720(2)
HLA-DR2	0.724(2)	0.736(1)	0.686(3)
HLA-DR3	0.623(2)	0.657(1)	0.532(3)
HLA-DR4	0.679(3)	0.717(1)	0.695(2)
HLA-DR5	0.719(2)	0.723(1)	0.617(3)
HLA-DR7	0.631(2)	0.765(1)	0.613(3)
HLA-DR8	0.608(3)	0.732(1)	0.714(2)
HLA-DR9	0.520(3)	0.779(2)	0.792(1)
HLA-DR11	0.544(3)	0.854(1)	0.850(2)
HLA-DR13	0.563(3)	0.623(2)	0.630(1)
HLA-DR15	0.805(1)	0.713(2)	0.663(3)
HLA-DR17	0.629(3)	0.769(1)	0.682(2)
HLA-DR51	0.800(1)	0.780(2)	0.672(3)
I-Ab	0.606(3)	0.611(2)	0.618(1)
I-Ad	0.821(1)	0.785(2)	0.676(3)
I-Ag7	0.823(1)	0.804(2)	0.757(3)
I-Ak	0.768(1)	0.766(2)	0.691(3)
I-Ed	0.828(2)	0.852(1)	0.787(3)
I-Ek	0.714(2)	0.789(1)	0.699(3)
Average	0.684(2.23)	0.751(1.45)	0.702(2.27)

similarity-reduced, and weighted versions of the MHCBN data for the corresponding alleles.

Table 16 shows that the 5-spectrum kernel classifier consistently performs poorly ($AUC \approx 0.5$) on the allele-specific blind test sets regardless of the version of the MHCBN dataset used for training the classifier. This finding is consistent with the cross-validation performance estimates obtained on the MHCBN SRDS1 and SRDS2 datasets (see Tables 12 and 13).

Table 17 shows the performance on the blind test sets of the CTD classifiers trained on different versions of MHCBN datasets. Interestingly, the CTD classifiers appear to be relatively insensitive to the choice of the specific version of the MHCBN dataset on which they were trained, with an average $AUC \approx 0.66$ in each case.

Table 10 AUC values for the three methods evaluated on MHCPEP-WUPDS datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DQ2	0.717(3)	0.738(2)	0.772(1)
HLA-DQ4	0.543(3)	0.882(2)	0.925(1)
HLA-DQ7	0.716(3)	0.786(2)	0.812(1)
HLA-DR1	0.696(3)	0.710(1)	0.699(2)
HLA-DR2	0.682(2)	0.688(1)	0.617(3)
HLA-DR3	0.612(3)	0.678(1)	0.614(2)
HLA-DR4	0.723(2.5)	0.723(2.5)	0.730(1)
HLA-DR5	0.765(1)	0.709(3)	0.733(2)
HLA-DR7	0.714(1)	0.599(3)	0.632(2)
HLA-DR8	0.796(3)	0.810(1)	0.802(2)
HLA-DR9	0.806(2)	0.819(1)	0.738(3)
HLA-DR11	0.612(3)	0.798(2)	0.830(1)
HLA-DR13	0.620(2)	0.714(1)	0.605(3)
HLA-DR15	0.760(1)	0.627(2)	0.587(3)
HLA-DR17	0.747(2)	0.760(1)	0.679(3)
HLA-DR51	0.838(1)	0.786(2)	0.718(3)
I-Ab	0.650(2)	0.669(1)	0.636(3)
I-Ad	0.815(1)	0.740(2)	0.707(3)
I-Ag7	0.820(1)	0.797(2)	0.700(3)
I-Ak	0.778(1)	0.684(2)	0.680(3)
I-Ed	0.742(3)	0.760(2)	0.805(1)
I-Ek	0.734(3)	0.824(1)	0.805(2)
Average	0.722(2.11)	0.741(1.7)	0.719(2.18)

Finally, Table 18 summarizes the performance on the blind test sets of the LA classifiers trained on the different versions of MHCBN datasets. Interestingly, the best performance (on four out of the five allele-specific blind test sets) is observed in the case of the LA classifiers trained on the SRDS2 versions of the corresponding allele-specific datasets.

In summary, our results show that MHC-II predictors trained on the similarity reduced versions of the dataset generally outperform those trained on the UPDS dataset. This suggests that similarity reduction contributes to improved generalization on blind dataset.

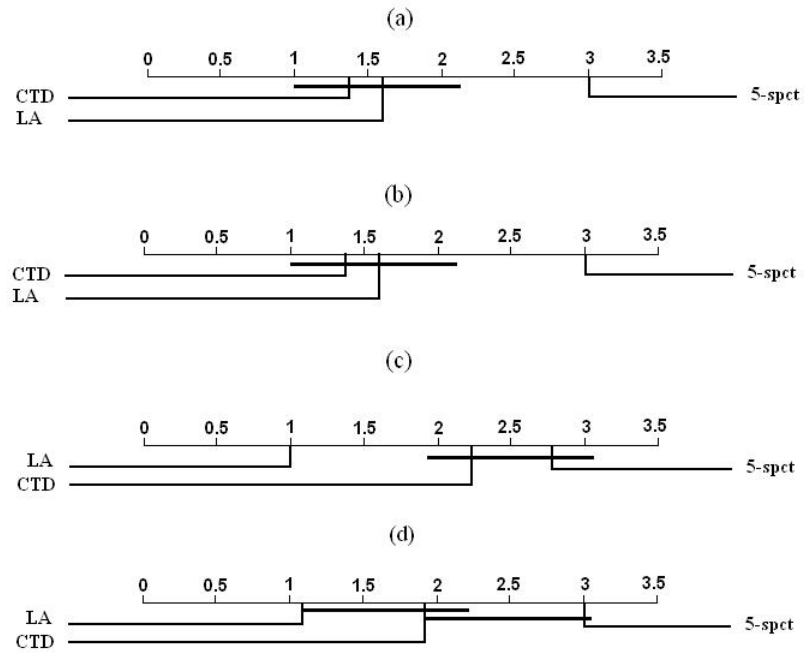


Figure 3 Pair-wise comparisons of classifiers with the Nemenyi test applied to results on a) MHCBN-SRDS1, b) MHCBN-SRDS2, c) MHCBN-SRDS3, and d) MHCBN-WUPDS. Classifiers that are not significantly different (at p-value = 0.05) are connected.

Discussion

Related work

Several previous studies have considered the importance of similarity reduction in datasets of MHC-II peptides. MHCBench (Raghava, 2004) is a benchmark of eight HLA-DRB1*0401 datasets representing a set of unique peptides (Set1), a dataset of natural peptides (Set2, derived from Set1 by removing peptides with Alanine residues), two non-redundant datasets (Set3a and Set3b derived from Set1 and Set2, respectively), two balanced datasets (Set4a and Set4b derived from Set1 and Set2 by randomly selecting equal numbers of binding and non-binding peptides), and two recent datasets of ligands (Set5a and Set5b, derived from Set1 and Set2 by considering only the most recently reported peptides). However, this benchmark considers only a single MHC-II allele, namely, HLA-DR4 (B1*0401). More importantly, as shown by our analysis of SRDS3 datasets, the similarity reduction procedure used in MHCBench is

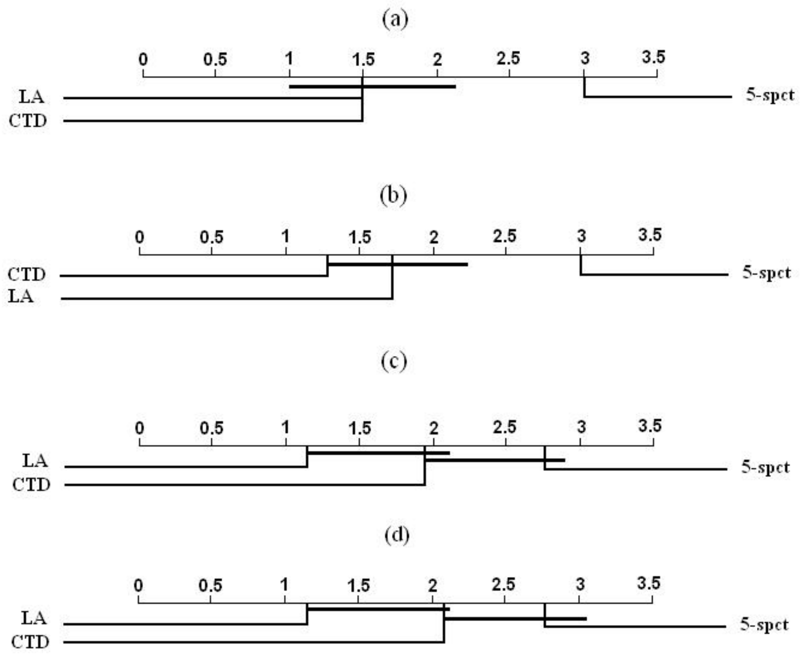


Figure 4 Pair-wise comparisons of classifiers with the Nemenyi test applied to results on a) IEDB-SRDS1, b) IEDB-SRDS2, c) IEDB-SRDS3, and d) IEDB-WUPDS. Classifiers that are not significantly different (at p-value = 0.05) are connected.

not stringent enough to ensure elimination of highly similar peptides.

Nielsen et al. (2004) and Murugan and Dai (2005) trained their classifiers using data extracted from MHCPEP and SYFPETHI databases and evaluated the classifiers using ten test sets, from which peptides similar to peptides in the training datasets had been removed. Recently, Nielsen et al. (2007) presented an MHC-II benchmarking dataset for regression tasks: each peptide is labeled with a real value indicating the binding affinity of the peptide. In this benchmark dataset, each set of allele-specific data had been partitioned into five subsets with minimal sequence overlap. However, neither of these studies explicitly examined the limitations of widely used benchmark datasets or the full implications of using MHC-II datasets of unique peptides in evaluating alternative methods.

Mallios (2003) compared three HLA-DRB1*0101 and HLA-DRB1*0401 prediction tools using an independent test set of two proteins. A consensus approach combining the predictions of the three methods was shown to be superior to the three methods. However, the significance

Table 11 AUC values for the three methods evaluated on MHCBN-UPDS datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DR1	0.747(3)	0.768(2)	0.789(1)
HLA-DR2	0.806(1)	0.771(3)	0.786(2)
HLA-DR5	0.743(3)	0.748(2)	0.752(1)
HLA-DRB10101	0.758(3)	0.799(2)	0.804(1)
HLA-DRB10301	0.770(3)	0.795(1)	0.787(2)
HLA-DRB10401	0.705(3)	0.780(1)	0.721(2)
HLA-DRB10701	0.778(2)	0.842(1)	0.732(3)
HLA-DRB11101	0.754(3)	0.874(1)	0.832(2)
Average	0.758(2.63)	0.797(1.63)	0.775(1.75)

Table 12 AUC values for the three methods evaluated on MHCBN-SRDS1 datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DR1	0.545(3)	0.784(1)	0.738(2)
HLA-DR2	0.456(3)	0.707(2)	0.750(1)
HLA-DR5	0.533(3)	0.657(2)	0.692(1)
HLA-DRB10101	0.456(3)	0.690(2)	0.748(1)
HLA-DRB10301	0.323(3)	0.617(2)	0.661(1)
HLA-DRB10401	0.381(3)	0.676(1)	0.655(2)
HLA-DRB10701	0.424(3)	0.665(2)	0.748(1)
HLA-DRB11101	0.493(3)	0.776(1)	0.759(2)
Average	0.451(3.00)	0.697(1.63)	0.719(1.38)

of this result is limited by the small dataset utilized in this study.

Two recent studies (Wang et al., 2008; Gowthaman and Agrewala, 2008) have pointed out some of the limitations of existing MHC-II prediction methods in identifying potential MHC-II binding peptides. Gowthaman and Agrewala (2008) used 179 peptides derived from eight antigens and covering seven MHC-II alleles to evaluate the performance of six commonly used MHC-II prediction methods and concluded that none of these methods can reliably identify potential MHC-II binding peptides. Wang et al. (2008) introduced a large benchmark dataset of previously unpublished peptides and used it to assess the performance of nine publicly available MHC-II binding peptide prediction methods. Both studies showed that the predictive

Table 13 AUC values for the three methods evaluated on MHCBN-SRDS2 datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DR1	0.448(3)	0.717(1)	0.698(2)
HLA-DR2	0.374(3)	0.665(2)	0.716(1)
HLA-DR5	0.369(3)	0.459(2)	0.588(1)
HLA-DRB10101	0.351(3)	0.705(1)	0.683(2)
HLA-DRB10301	0.273(3)	0.563(2)	0.628(1)
HLA-DRB10401	0.261(3)	0.658(1)	0.620(2)
HLA-DRB10701	0.414(3)	0.617(2)	0.696(1)
HLA-DRB11101	0.386(3)	0.705(2)	0.757(1)
Average	0.360(3.00)	0.636(1.63)	0.673(1.38)

Table 14 AUC values for the three methods evaluated on MHCBN-SRDS3 datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DR1	0.685(3)	0.768(1)	0.743(2)
HLA-DR2	0.709(3)	0.741(1)	0.719(2)
HLA-DR5	0.557(3)	0.616(1)	0.608(2)
HLA-DRB10101	0.691(3)	0.819(1)	0.725(2)
HLA-DRB10301	0.678(2)	0.736(1)	0.675(3)
HLA-DRB10401	0.624(3)	0.760(1)	0.710(2)
HLA-DRB10701	0.737(2)	0.794(1)	0.671(3)
HLA-DRB11101	0.755(3)	0.816(1)	0.775(2)
Average	0.680(2.75)	0.756(1.00)	0.703(2.25)

performance of existing MHC-II prediction tools on independent blind test sets is substantially worse than the performance of these tools reported by their developers. Our work complements these studies by providing a plausible explanation of this result.

We have shown that the previously reported similarity reduction methods may not eliminate highly similar peptides, i.e., peptides that share $> 80\%$ sequence identity still pass the similarity test. We have proposed a two-step similarity reduction procedure that is much more stringent than those currently in use for similarity reduction with MHC-II benchmark datasets. We have used the similarity reduction method used in MHCBench, as well as our proposed 2-stage method to derive similarity-reduced MHC-II benchmark datasets based on peptides retrieved

Table 15 AUC values for the three methods evaluated on MHCB-N-WUPDS datasets. For each dataset, the rank of each classifier is shown in parentheses.

Allele	5-spectrum	LA	CTD
HLA-DR1	0.655(3)	0.747(1)	0.732(2)
HLA-DR2	0.636(3)	0.717(2)	0.740(1)
HLA-DR5	0.518(3)	0.594(1)	0.543(2)
HLA-DRB10101	0.535(3)	0.672(1)	0.666(2)
HLA-DRB10301	0.648(3)	0.710(1)	0.678(2)
HLA-DRB10401	0.536(3)	0.757(1)	0.701(2)
HLA-DRB10701	0.667(3)	0.724(1)	0.702(2)
HLA-DRB11101	0.676(3)	0.820(1)	0.789(2)
Average	0.609(3)	0.718(1.13)	0.694(1.88)

Table 16 AUC values for 5-spectrum based classifiers trained using MHCBN- UPDS, SRDS1, SRDS2, SRDS3, and WUPDS datasets and evaluated on the blind test sets of Wang et al. (2008).

Allele	UPDS	SRDS1	SRDS2	SRDS3	WUPDS
HLA-DRB1-0101	0.505	0.503	0.504	0.506	0.505
HLA-DRB1-0301	0.518	0.515	0.515	0.516	0.518
HLA-DRB1-0401	0.504	0.500	0.500	0.501	0.487
HLA-DRB1-0701	0.500	0.500	0.500	0.500	0.496
HLA-DRB1-1101	0.500	0.500	0.500	0.500	0.496
Average	0.505	0.504	0.504	0.505	0.500

from MHCPEP and MHCBN databases. Comparison of the similarity-reduced versions of MHCPEP, MHCBN, and IEDB datasets with their original UPDS counterparts showed that nearly 50% of the peptides in the UPDS datasets are, in fact, highly similar.

Extensions to multi-class and multi-label prediction problems

Our description of the proposed similarity reduction procedure assumes a 2-class prediction problem. However, our proposed approach can easily be adapted to multi-class prediction (wherein an instance has associated with one of several mutually exclusive labels). One can simply apply the similarity reduction procedure separately to data from each class.

A more interesting setting is that of multi-label prediction (wherein each instance is associated with a subset of a set of candidate labels). Consider for example, the problem of

Table 17 AUC values for CTD classifiers trained using MHCBN- UPDS, SRDS1, SRDS2, SRDS3, and WUPDS datasets and evaluated on the blind test sets of Wang et al. (2008).

Allele	UPDS	SRDS1	SRDS2	SRDS3	WUPDS
HLA-DRB1-0101	0.689	0.707	0.684	0.714	0.629
HLA-DRB1-0301	0.595	0.589	0.597	0.596	0.585
HLA-DRB1-0401	0.605	0.584	0.611	0.633	0.601
HLA-DRB1-0701	0.675	0.711	0.699	0.684	0.694
HLA-DRB1-1101	0.732	0.701	0.719	0.713	0.735
Average	0.659	0.658	0.662	0.668	0.649

Table 18 AUC values for LA classifiers trained using MHCBN- UPDS, SRDS1, SRDS2, SRDS3, and WUPDS datasets and evaluated on the blind test sets of Wang et al. (2008).

Allele	UPDS	SRDS1	SRDS2	SRDS3	WUPDS
HLA-DRB1-0101	0.675	0.650	0.756	0.736	0.703
HLA-DRB1-0301	0.604	0.647	0.651	0.637	0.604
HLA-DRB1-0401	0.554	0.548	0.610	0.595	0.573
HLA-DRB1-0701	0.627	0.692	0.692	0.677	0.627
HLA-DRB1-1101	0.775	0.722	0.701	0.730	0.775
Average	0.647	0.652	0.682	0.675	0.656

predicting promiscuous MHC binding peptides (Zhang et al., 2007), where each peptide can bind to multiple HLA molecules. Current methods for multi-label prediction typically reduce the multi-label prediction task to a collection of binary prediction tasks (Tsoumakas and Katakis, 2007). Hence, the similarity reduction methods proposed in this paper can be directly applied to the binary labeled datasets resulting from such a reduction.

Implications for rigorous assessment of MHC-II binding peptide prediction methods

The results of our study show that the observed performance of some of the methods (e.g., the CTD and the LA kernels) on benchmark datasets of unique peptides can be rather optimistic relative to the performance of the same methods on similarity-reduced counterparts of the same datasets or on blind test sets. This suggests that the performance of existing MHC-II prediction methods, when applied to novel peptide sequences, may turn out to be less satisfactory than one might have been led to believe based on the reported performance of such methods on some of the widely used benchmark. Moreover, the conclusions based on observed performance on datasets of unique peptides regarding the superior performance of one method relative to another can be highly unreliable in more realistic settings e.g., predictions of novel peptides.

These results underscore the importance of rigorous comparative evaluation of a broad range of existing methods for MHC-II binding peptides prediction methods using similarity-reduced datasets. We expect that such studies are likely to show much greater room for improvement over the state-of-the-art MHC-II prediction tools than one might be led to believe based on reported performance on the widely-used benchmark datasets and motivate the research community to develop improved methods for this important task. We hope that such comparisons will be facilitated by the availability of the similarity-reduced versions of MHCPEP, MHCBN, and IEDB datasets used in our experiments. These datasets (Datasets S1-S3), Java source code implementation of the similarity reduction and weighting procedures (Code S1), and the supplementary materials (Data S1) have been made freely available (see Supporting Information).

Materials and Methods

The datasets used in this study are derived from MHCPEP (Brusic et al., 1998), MHCBN (Bhasin et al., 2003), and IEDB (Peters et al., 2005), which are manually curated repositories of MHC binding peptides reported in the literature.

We extracted 22 MHC-II allele datasets (each with at least 100 binders) from the MHCPEP database. Because MHCPEP contains only MHC-II binding peptides (”positive examples”), for each allele, we generated an equal number of non-binders (”negative examples”) by randomly extracting protein fragments from SwissProt (Bairoch and Apweiler, 2000) protein sequences such that: (i) The length distribution of negative examples is identical to that of the positive examples; (ii) None of the non-binding peptides appear in the set of binders.

Unlike MHCPEP, MHCBN is a database of binding and non-binding MHC peptides.

MHCBN version 4.0 has 35 MHC-II alleles with at least 100 binders. Out of these 35 alleles, only eight alleles have at least 100 non-binders. We extracted the MHCBN benchmark dataset used in this study from the alleles for which at least 100 binders and non-binders peptides are available in MHCBN.

The Immune Epitope Database and Analysis Resource (IEDB) (Peters et al., 2005) is a rich resource of MHC binding data curated from the literature or submitted by immunologists. For each reported peptide, IEDB provides qualitative (i.e., Negative or Positive) and quantitative (i.e., IC50) measurements whenever available. We used both qualitative and quantitative measurements for constructing 12 HLA binary labeled datasets as follows:

- Peptides with no reported quantitative measurements are discarded.
- Peptides with "Positive" qualitative measurement and quantitative measurement less than 500 nM are classified as binders.
- Peptides with "Positive" qualitative measurement and quantitative measurement greater than or equal 500 nM are classified as non-binders.
- Peptides with "Negative" qualitative measurement and quantitative measurement greater than or equal 500 nM are classified as non-binders.
- Peptides with "Negative" qualitative measurement and quantitative measurement less than 500 nM are discarded.

The reported MHC binding sites are typically identified using truncation, substitution, or mutations in a base peptide (O'Sullivan et al., 1991). Because different reported MHC-II binding peptides might actually correspond to experimental manipulation of the same MHC-II binding region using different experimental techniques or different choices of amino acids targeted for truncation, substitution, or mutation, it is not surprising that that MHC databases contain a significant number of highly similar peptides. Hence, we used several similarity reduction methods to extract several different versions of the dataset from each set of sequences.

It should be noted that the existence of highly similar peptides belonging to the same category may result in an over-optimistic estimation of the classifier performance. Therefore, we applied the similarity reduction procedures separately to the set of binders and non-binders in each dataset. The following sections describe the similarity reduction procedures and the resulting similarity-reduced datasets.

Similarity reduction procedures

An example of two different types of similar peptides that frequently occur in MHC peptides databases is shown in Figure 5. In type I, two peptides differ from each other in terms of only one or two amino acids (see Figure 5A). Such highly similar peptides are likely to have come from different mutation experiments targeting different sites of the same MHC-II binding peptide. For example, Garcia et al. (2007) report an HLA-DRB1*0401 binding peptide (WGENDTDVFVLNNTR) and 12 additional binding peptides derived from that peptide by replacing one of the amino acid in (WGENDTDVFVLNNTR) sequence with Glycine and experimentally determining the binding affinity of the new peptide. In type II, we find that a shorter peptide in one allele dataset corresponds to a sub-sequence of a longer one that is also in the allele dataset (see Figure 5B).

Standard approaches to identifying similar peptide sequences rely on the use of a sequence similarity threshold. Sequences that are within a certain predetermined similarity threshold relative to a target sequence are eliminated from the dataset. However, the use of such a simple approach to obtaining a similarity reduced dataset is complicated by the high degree of variability in the length of MHC-II peptides. Using a single fixed similarity cutoff value (e.g. 80%) might not be effective in eliminating type II similar peptides. On the other hand, an attempt to eliminate one of the two such similar sequences by using of a more stringent similarity threshold could result in elimination of most of the dataset.

To address this problem, we used a two-step similarity reduction procedure to eliminate similar peptides of types I and II:

- Step 1 eliminates similar peptides based on a criterion proposed by Nielsen et al. (2007).

Two peptides are considered similar if they share a 9-mer subsequence. This step will eliminate all similar peptides of type II but is not guaranteed to remove all similar peptides of Type I. For example, this method will not eliminate one of the two peptides in Figure 5A although they share 84.6% sequence similarity.

- Step 2 filters the dataset using an 80% similarity threshold to eliminate any sequence that has a similarity of 80% or greater with one or more sequences in the dataset.

In addition, we also used a procedure proposed by Raghava (2004) for similarity reduction of MHCBench benchmark datasets. Briefly, given two peptides p_1 and p_2 of lengths l_1 and l_2 such that $l_1 \leq l_2$, we compare p_1 with each l_1-length subpeptide in p_2. If the percent identity (PID) between p_1 and any subpeptide in p_2 is greater than 80%, then the two peptides are deemed to be similar. For example, to compute the PID between (ACDEFGHIKLMNPQRST) and (DEFGGIKLMN), we compare (DEFGGIKLMN) with (ACDEFGHIKL), (CDEFGHIKLM), . . ., (IKLMNPQRST). The PID between (DEFGGIKLMN) and (DEFGHIKLMN) is 90% since nine out of 10 residues are identical.

Finally, we explored a method for assigning weights to similar peptides as opposed to eliminating similar peptides from the dataset. Specifically, the peptides within the binders category that are similar to each other (i.e., share a 9-mer subsequence or have sequence similarity of 80% or greater) are clustered together. Each peptide that is assigned to a cluster is similar to at least one other peptide within the cluster, and no two similar peptides are assigned to different clusters. Each peptide in a cluster is assigned a weight $1/n$, where n is the number of peptides assigned to the cluster. The process is repeated with peptides in the non-binders category. The result is a dataset of weighted instances.

Thus, from each MHC-II benchmark dataset, we generated five versions summarized below:

- Three datasets of unique peptides, MHCPEP-UPDS, MHCBN-UPDS, and IEDB-UPDS extracted from MHCPEP, MHCBN, and IEDB, respectively after eliminating short peptides consisting of fewer than 9 residues, unnatural peptides, peptides with greater than 75% Alanine residues, and duplicated peptides.

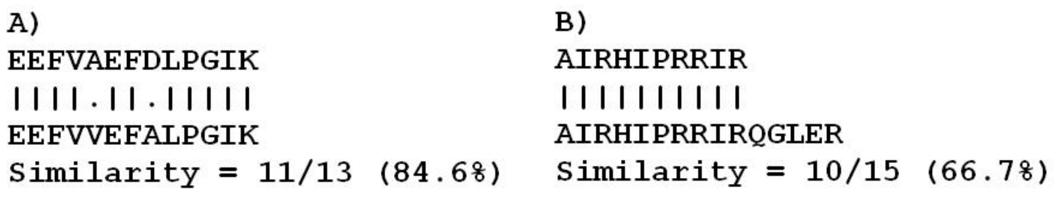
- Three datasets of similarity-reduced peptides, MHCPEP-SRDS1, MHCBN-SRDS1, and IEDB-SRDS1 derived from the corresponding UPDS datasets described above using only step 1 of the two-step similarity reduction procedure described above which ensures that no two peptides in the resulting datasets of binders or non binders share a 9-mer subsequence.

- Three datasets of similarity-reduced peptides, MHCPEP-SRDS2, MHCBN-SRDS2, and IEDB-SRDS2, extracted MHCPEP-SRDS1, MHCBN-SRDS1, and IEDB-SRDS1 respectively by filtering the binders and non-binders in SRDS1 such that the sequence identity between any pair of peptides in the binders category or in the non-binders category is less than 80

- Three datasets of similarity-reduced peptides, MHCPEP-SRDS3, MHCBN-SRDS3, and IEDB-SRDS3, derived from the corresponding UPDS datasets by applying the similarity reduction procedure introduced by Raghava which has been used to construct the MHCBench dataset (Raghava, 2004).

- Three weighted unique peptide datasets, MHCPEP-WUPDS, MHCBN-WUPDS, and IEDB-WUPDS, derived from the corresponding UPDS datasets by applying the peptide weighting method described above.

The procedure used to generate the five different versions of each allele-specific dataset using the different similarity reduction methods and the peptide weighting method described above is shown in Figure 6. Note that UPDS can contain similar peptides of both types I and II; SRDS1 can contain similar peptides of type I; SRDS2 is free from both type I and type II similar peptides; SRDS3 simulates similarity-reduced datasets using the method employed with MHCBench; WUPDS is a weighted version of the UPDS dataset where similar peptides are grouped into disjoint clusters and the weight of each peptide is set to one over the size of its cluster.



```
A)                                  B)
EEFVAEFDLPGIK                       AIRHIPRRIR
||||.||.|||||                       ||||||||||
EEFVVEFALPGIK                       AIRHIPRRIRQGLER
Similarity = 11/13  (84.6%)         Similarity = 10/15  (66.7%)
```

Figure 5 Two types of similar peptides that frequently appear in MHC databases.

Summary of the datasets

Datasets derived from MHCPEP

Table 1 summarizes the number of binders in each unique peptides dataset, MHCPEP-UPDS, and the corresponding three similarity-reduced datasets, MHCPEP-SRDS1, MHCPEP-SRDS2, and MHCPEP-SRDS3. Note that on average, the number of binders in the similarity-reduced datasets, MHCPEP-SRDS1, MHCPEP-SRDS2, and MHCPEP-SRDS3, is reduced to 48%, 33%, and 39%, respectively, of the number of binders in MHCPEP-UPDS datasets.

Datasets derived from MHCBN

Table 2 summarizes the number of binders and non-binders in MHCBN-UPDS, MHCBN-SRDS1, MHCBN-SRDS2 and MHCBN-SRDS3 datasets derived for each of the eight MHCBN alleles satisfying our selection criteria. Note that the average number of binders in similarity-reduced datasets, MHCBN-SRDS1, MHCBN-SRDS2, and MHCBN-SRDS3, is reduced to 55.48%, 45.46%, and 61.39%, respectively, of the number of binders in MHCBN-UPDS datasets. Similarly, the average number of non-binders in similarity-reduced datasets, MHCBN-SRDS1, MHCBN-SRDS2, and MHCBN-SRDS3, is reduced to 67.55%, 64.24%, and 87.47%, respectively, of the number of non-binders in MHCBN-UPDS datasets.

Datasets derived from IEDB

Table 3 summarizes the number of binders and non-binders in IEDB-UPDS, IEDB-SRDS1, IEDB-SRDS2, and IEDB-SRDS3 datasets derived for 12 HLA alleles. We observed that the average number of binders in similarity-reduced datasets, IEDB-SRDS1, IEDB-SRDS2, and

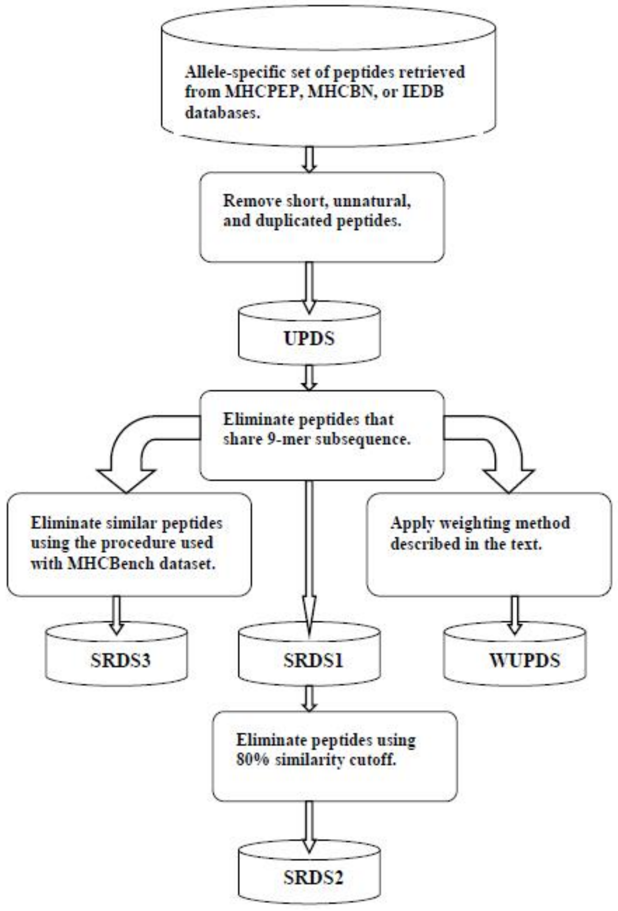

Allele-specific set of peptides retrieved from MHCPEP, MHCBN, or IEDB databases.
Remove short, unnatural, and duplicated peptides.
UPDS
Eliminate peptides that share 9-mer subsequence.
Eliminate similar peptides using the procedure used with MHCBench dataset.
Apply weighting method described in the text.
SRDS3
SRDS1
WUPDS
Eliminate peptides using 80% similarity cutoff.
SRDS2

Figure 6 An overview of the process used for generating five different versions of each allele dataset using the different similarity-reduction methods described in the text.

IEDB-SRDS3, is reduced to 51.17%, 47.66%, and 63.5%, respectively, of the number of binders in MHCBN-UPDS datasets. Similarly, the average number of non-binders in similarity-reduced datasets, IEDB-SRDS1, IEDB-SRDS2, and IEDB-SRDS3, is reduced to 60.86%, 59.38%, and 82.9%, respectively, of the number of non-binders in MHCBN-UPDS datasets.

Independent blind set

Recently, Wang et al. (2008) introduced a comprehensive dataset of previously unpublished MHC-II peptide binding affinities and utilized it to assessing the performance of nine publicly available MHC-II prediction methods. The dataset covers 14 HLA alleles and two Mouse alleles. Out of the 14 HLA allele-specific datasets, five datasets are used in our experiments as independent blind test data to evaluate the performance of the classifiers trained using the corresponding MHCBN allele-specific datasets. Table 19 shows the number of test peptides in each allele-specific dataset and the number of binders and non-binders obtained using an IC50 cutoff of 500 nM employed to categorize peptides into binders and non-binders (Nielsen et al., 2007).

Table 19 Five allele-specific blind test set obtained Wang et al. (2008). Peptides are categorized into binders and non-binders using an IC50 cutoff 500 nM.

Allele	peptides	binders	non-binders
HLA-DRB1-0101	3882	2579	1303
HLA-DRB1-0301	502	209	293
HLA-DRB1-0401	512	286	226
HLA-DRB1-0701	505	358	147
HLA-DRB1-1101	520	317	203

Prediction methods

Our experiments focused on two approaches for training MHC-II binding peptide predictors from variable-length MHC-II peptides have been recently proposed in (Cui et al., 2006a; Salomon and Flower, 2006) and a method based on k-spectrum kernel (Leslie et al., 2002) that is designed to rely on the presence of high degree of sequence similarity between training and test peptides (and hence is expected to perform well on redundant datasets but poorly on similarity-reduced datasets). We implemented the three methods in java using Weka machine learning workbench (Witten and Frank, 2005). Brief descriptions of each of the three prediction methods are included below.

Composition-Transition-Distribution (CTD)

The basic idea of this approach is to map each variable-length peptide into a fixed-length feature vector such that standard machine learning algorithms are applicable. This method was used and explained in details in (Cai et al., 2003; Cui et al., 2006a). 21 features are extracted from each peptide sequence as follows:

- First, each peptide sequence p is mapped into a string s_p defined over an alphabet of three symbols, $\{1, 2, 3\}$. The mapping is performed by grouping amino acids into three groups using a physico-chemical property of amino acids (see Table 20). For example the peptide (AIRHIPRRIR) is mapped into (2312321131) using the hydrophobicity division of amino acids into three groups (see Table 20).
- Second, for each peptide string s_p, three descriptors are derived as follows:
  - Composition (C): three features representing the percent frequency of the symbols, $\{1, 2, 3\}$, in the mapped peptide sequence.
  - Transition (T): three features representing the percent frequency of i followed by j or j followed by i, for $i, j \in \{1, 2, 3\}$.
  - Distribution (D): five features per symbol representing the fractions of the entire sequence where the first, 25, 50, 75, and 100% of the candidate symbol are contained in s_p . A total of 15 features are derived from each peptide.

Table 20 shows division of the 20 amino acids into three groups based on hydrophobicity, polarizability, polarity, and Van der Waal's volume properties. Using these four properties, we derived 84 CTD features from each peptide sequence. In our experiments, we trained SVM classifiers using RBF kernel and peptide sequences represented using their amino acid sequence composition (20 features) and CTD descriptors (84 features).

Table 20 Categorization of amino acids into three groups for a number of physicochemical properties Chinnasamy et al. (2004).

Property	Group 1	Group 2	Group 3
Hydrophobicity	RKEDQN	GASTPHY	CVLIMFW
Polarizability	GASCTPD	NVEQIL	MHKFRYW
Polarity	LIFWCMVY	PATGS	HQRKNED
Van der Waal's volume	GASDT	CPNVEQIL	KMHFRYW

Local alignment (LA) kernel

Local alignment (LA) kernel (Saigo et al., 2004) is a string kernel designed for biological sequence classification problems. The LA kernel measures the similarity between two sequences by adding up the scores obtained from local alignments with gaps of the sequences. This kernel has several parameters: the gap opening and extension penalty parameters d and e, the amino acid mutation matrix s, and the factor β which controls the influence of suboptimal alignments in the kernel value. Saigo et al. (2004) used the BLOSUM62 substitution matrix, gap opening and extending parameters equal 11 and 1, respectively, and β ranges from 0.2 to 0.5. In our experiments, we tried a range of values for gap opening/extension and β parameters and got the best performance out of LA kernel using BLOSUM62 substitution matrix, gap opening and extending parameters equal 10 and 1, respectively, and $\beta = 0.5$. Detailed formulation of the LA kernel and a dynamic programming implementation of the kernel are provided in (Saigo et al., 2004).

k-spectrum kernel

Intuitively, k-spectrum kernel (Leslie et al., 2002) captures a simple notion of string similarity: two strings are deemed similar (i.e., have a high k-spectrum kernel value) if they share many of the same k-mer substrings. We used the k-spectrum with relatively large k value, $k = 5$. As noted earlier, the choice of a relatively large value for k was motivated by the desire to construct a predictor that is expected to perform well in settings where the peptides in the test set share significant similarity with one or more peptides in the training set.

Performance evaluation

The prediction accuracy (ACC), sensitivity (S_n), specificity (S_p), and correlation coefficient (CC) are often used to evaluate prediction algorithms (Baldi et al., 2000). The CC measure has a value in the range from -1 to +1 and the closer the value to +1, the better the predictor. The S_n and S_p summarize the accuracies of the positive and negative predictions respectively.

ACC, S_n, S_p, and CC are defined as follows:

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$S_n = \frac{TP}{TP + FN} \; and \; S_p = \frac{TN}{TN + FP} \tag{2}$$

$$CC = \frac{TP \times TN - FP \times FN}{\sqrt{(TN + FN)(TN + FP)(TP + FN)(TP + FP)}} \tag{3}$$

where TP, FP, TN, FN are the numbers of true positives, false positives, true negatives, and false negatives respectively.

Although these metrics are widely used to assess the performance of machine learning methods, they all suffer from an important limitation of being threshold-dependent. Threshold-dependent metrics describe the classifier performance at a specific threshold value. It is often possible to increase the number of true positives (equivalently sensitivity) of the classifier at the expense of an increase in false positives (equivalently false alarm rate). Receiver Operating Characteristic (ROC) curve describes the performance of the classifier over all possible thresholds. The ROC curve is obtained by plotting the true positive rate as a function of the false positive rate or, equivalently, sensitivity versus (1-specificity) as the discrimination threshold of the binary classifier is varied. Each point on the ROC curve describes the classifier at a certain threshold value and hence a particular choice of tradeoff between true positive rate and false negative rate. The area under ROC curve (AUC) is a useful summary statistic for comparing two ROC curves. AUC is defined as the probability that a randomly chosen positive example will be ranked higher than a randomly chosen negative example. An ideal classifier will have an AUC = 1, while a classifier performs no better than random will have an AUC = 0.5, any classifier performing better than random will have an AUC value that lies between these two extremes.

Implementation and SVM parameter optimization

We used Weka machine learning workbench (Witten and Frank, 2005) for implementing the spectrum, and LA kernels (RBF kernel is already implemented in Weka). For the SVM classifier, we used the Weka implementation of the SMO algorithm (Platt, 1998). For k-spectrum and LA kernels, the default value of the cost parameter, $C = 1$, was used for the SMO classifier. For the RBF kernel, we found that tuning the SMO cost parameter C and the RBF kernel parameter γ is necessary to obtain satisfactory performance. We tuned these parameters using a two dimensional grid search over the range $C = 2^{-5}, 2^{-3}, \ldots, 2^{3}, \gamma = 2^{-15}, 2^{-13}, \ldots, 2^{3}$.

Supporting Information

The following Supporting information can be freely downloaded from PLoS ONE [1] or requested from the author:

Data S1: Detailed results on MHCPEP, MHCBN, and IEDB datasets

Dataset S1: Datasets derived from MHCPEP database

Dataset S2: Datasets derived from MHCBN database

Dataset S3: Datasets derived from IEDB database

Code S1: Java programs implementing the similarity reduction and peptide weighting methods

Author Contributions

Conceived and designed the experiments: YEM. Performed the experiments: YEM. Analyzed the data: YEM DD VH. Contributed reagents/materials/analysis tools: YEM DD VH. Wrote the paper: YEM DD VH.

Acknowledgment

This research was supported in part by a doctoral fellowship from the Egyptian Government to Yasser El-Manzalawy and a grant from the National Institutes of Health (GM066387) to Vasant Honavar and Drena Dobbs.

[1] http://www.plosone.org/article/info:doi%2F10.1371%2Fjournal.pone.0003268

CHAPTER 6. PREDICTING MHC-II BINDING AFFINITY USING MULTIPLE INSTANCE REGRESSION

A paper submitted to the Journal of IEEE/ACM Transactions on Computational Biology and Bioinformatics

Yasser EL-Manzalawy, Drena Dobbs, Vasant Honavar

Abstract

Reliably predicting the ability of antigen peptides to bind to major histocompatibility complex class II (MHC-II) molecules is an essential step in developing new vaccines. Uncovering the amino acid sequence correlates of MHC-II peptides binding affinity can contribute to a deeper understanding of how pathogens cause disease and how the immune system responds to them. Unfortunately, the variable length of MHC-II binding peptides complicates the prediction task. Most existing computational methods for predicting MHC-II binding peptides rely on methods for identifying a nine amino acids core region in each binding peptide. We formulate the problems of qualitatively and quantitatively predicting flexible length MHC-II peptides as multiple instance learning and multiple instance regression problems, respectively. Based on this formulation, we introduce MHCMIR, a novel method for predicting MHC-II binding affinity using multiple instance regression. We present results of experiments using a benchmark dataset covering 13 HLA-DR and three H2-IA alleles that show that MHCMIR is competitive with the state-of-the-art methods for predicting MHC-II binding peptides. An online web server that implements the MHCMIR method for MHC-II binding affinity prediction is freely accessible at http://ailab.cs.iastate.edu/mhcmir.

Introduction

T-cells, a major type of the immune system cells, play a central role in the cell-mediated immunity (Janeway et al., 2004). Cytotoxic T-cells attack cells that have certain foreign or abnormal molecules on their surfaces. They have also been implicated in transplant rejection. Helper T-cells, or CD4+ T-cells, coordinate immune responses by communicating with other cells. Once activated, they divide rapidly and secrete cytokines that regulate the immune response. T-cells are also targets of HIV infection, with the loss of CD4+ T-cells being associated with the appearance of AIDS symptoms. Regulatory T-cells are believed to be crucial for the maintenance of immunological tolerance. T-cells epitopes are short linear peptides that are generated by the cleavage of antigenic proteins. The identification of T-cell epitopes in protein sequences is important for understanding disease pathogenesis, for identifying potential autoantigens, and for designing vaccines and immune-based cancer therapies. Predicting whether a given peptide will bind to a specific major histocompatibility complex (MHC) molecule (and the binding affinity) is an important step in identifying potential T-cell epitopes. Consequently, predicting MHC binding peptides is an important and challenging task in immunoinformatics (Korber et al., 2006; Gowthaman and Agrewala, 2008).

There are two classes of MHC molecules: MHC class I (MHC-I) molecules that are characterized by short binding peptides, usually consisting of 9 amino acid residues; and MHC class II (MHC-II) molecules that bind to peptides of variable length. MHC-II binding peptides typically vary from 11 to 30 amino acids in length, although shorter and longer MHC-binding peptides are not entirely uncommon (Rammensee et al., 1995). MHC-II molecules allow variable length peptides to bind because the binding groove of MHC-II molecule is open at both ends. However, it has been reported that a 9-mer core region is essential for MHC-II binding activity of peptides (Madden, 1995; Rammensee et al., 1995). Because the precise location of the 9-mer core region of the MHC-II binding peptide is unknown, predicting MHC-II binding peptides is more challenging than predicting MHC-I binding peptides.

The computational methods that are currently available for predicting MHC-II peptides can be categorized into two major categories:

- Quantitative MHC-II binding prediction methods that attempt to predict the binding affinities (e.g., IC50 values); Examples of such methods include PLS-ISC (Doytchinova and Flower, 2003), MHCPred (Hattotuwagama et al., 2004), SVRMHC (Liu et al., 2006), ARB (Bui et al., 2005), and NetMHCII (Nielsen et al., 2007).

- Qualitative MHC-II binding prediction methods that simply classify MHC peptides into binders and non-binders; Examples of such methods include: (i) methods that use a position weight matrix to model ungapped multiple sequence alignment of MHC binding peptides (Reche et al., 2004; Singh and Raghava, 2001; Nielsen et al., 2004; Rajapakse et al., 2007; Nielsen et al., 2007), or rely on Hidden Markov Models (HMMs) (Mamitsuka, 1998; Noguchi et al., 2002); (ii) supervised machine learning methods based on Artificial Neural Networks (ANN) (Nielsen et al., 2003; Buus et al., 2003) or Support Vector Machines (SVMs) (Donnes and Kohlbacher, 2006; Bhasin and Raghava, 2004; Cui et al., 2006a; Salomon and Flower, 2006); and (iii) semi-supervised machine learning methods (Murugan and Dai, 2005; Hertz and Yanover, 2006).

Most of the currently available MHC-II binding prediction methods focus on identifying a putative 9-mer MHC-II binding core region, e.g., based on the degree of match with a 9-mer MHC-II binding motif, typically constructed using one of the motif finding algorithms. For example, MEME (Bailey and Elkan, 1995), Gibbs sampling (Lawrence et al., 1993), matrix optimization techniques (MOTs) (Singh and Raghava, unpublished data), evolutionary algorithms (Fonseca and Fleming, 1993), Mont Carlo (MC) search (Metropolis et al., 2004), and linear programming (Bennett and Mangasarian, 1992) form the basis of MHC-II binding peptide prediction methods RankPEP (Reche et al., 2004), Gibbs (Nielsen et al., 2004), HLA-DR4Pred (Bhasin and Raghava, 2004), MOEA (Rajapakse et al., 2007), NetMHCII (Nielsen et al., 2007), and LP (Murugan and Dai, 2005), respectively. The success of these MHC-II prediction methods in identifying MHC-II peptides relies on the effectiveness of the corresponding motif-finding methods in recognizing the motif that characterizes the 9-mer core of MHC-II binding peptides. An inherent limitation of such MHC-II prediction methods is their inability to make use of any potentially useful signals that lie outside the 9-mer core region (Chang et al., 2006; Nielsen et al., 2007). Moreover, Wang et al. (Wang et al., 2008) showed that existing MHC-II prediction tools lack consistency in identifying the 9-mer binding cores.

Recently, two methods (Cui et al., 2006a; Salomon and Flower, 2006) for predicting flexible length MHC-II peptides have been proposed. Both methods use the entire sequences of MHC-II peptides (as opposed to only the 9-mer cores) for training MHC-II binding peptide predictors. The first method (Cui et al., 2006a) maps a variable length peptide into a fixed length feature vector obtained from sequence-derived structural and physicochemical properties of the peptide. The second method (Salomon and Flower, 2006) uses a sequence kernel that defines the pair-wise similarity of variable-length peptides as the average score of all possible local alignments between the corresponding amino acid sequences.

Against this background, the main contributions of this paper to the state-of-the-art in predicting flexible length MHC-II peptides are as follows:

(a) Novel multiple instance learning (MIL) and multiple instance regression (MIR) formulations (respectively) of the flexible length MHC-II binding prediction problem and the MHC-II binding affinity prediction problem. In this setting, a peptide sequence, regardless of its length, is represented by a *bag* of 9-mer subsequences. The resulting bags are labeled with a binary class label (indicating whether or not the peptide sequence binds to an MHC-II molecule) or a binding affinity (respectively). This avoids the need to know the precise location of the 9-mer MHC-II binding core within the peptide sequence prior to training the corresponding classifier or regressor. The 9-mer binding cores are instead determined as a result of training.

(b) MILESreg, an adaptation of MILES (Chen et al., 2006) for multiple instance regression on bags of amino acid sequences.

(c) MHCMIR, a novel method for predicting the binding affinity of flexible lenghth MHC-II peptides using MILESreg. The performance of MHCMIR evaluated on a benchmark dataset covering 13 HLA-DR and three H2-IA alleles demonstrates the feasibility of our proposed approach and shows that the proposed MHCMIR method is competitive with the state-of-the-art methods for predicting MHC-II binding peptides on a majority of MHC-II alleles. An implementation of MHCMIR as an online web server for predicting MHC-II binding affinity is freely accessible at `http://ailab.cs.iastate.edu/mhcmir`.

Multiple instance learning

The multiple instance learning (MIL) problem, first introduced by Dietterich et al. (Dietterich et al., 1997) was motivated by a challenging classification task in drug discovery where the goal is to determine whether or not a given molecule will bind to a desired protein binding site (Dietterich et al., 1997). In this task, each molecule can adopt multiple shapes (conformations) as a consequence of rotation of some internal bonds. A good drug candidate is one that has one or more conformations that bind tightly to the desired binding site on a target protein whereas a poor drug candidate is one that has no conformations that bind tightly to the desired binding site on the target protein. A multiple instance learning (MIL) formulation of this problem (Dietterich et al., 1997) involves representing each candidate molecule by a *bag* of instances, with each instance in the bag representing a unique conformation assumed by the molecule. Under the so-called *standard MIL assumption*, a molecule (i.e., the corresponding bag of conformations) is labeled positive if and only if at least one of the conformations in the bag binds tightly to the desired binding site on the target protein; Otherwise, it is labeled negative. More generally, a bag is labeled positive if it contains at least one positive instance, and negative otherwise. During classification, the MIL classifier is given a bag of instances to be assigned a positive or negative label based on the instances in the bag. What makes the MIL problem challenging is the fact that the learning algorithm has access to the makeup of, and the label assigned to, each bag; but not the specific instance(s) in a positively labeled bag that are responsible for the positive label.

In the standard (single instance) supervised classifier learning scenario, typically, each instance (input to the classifier) is represented by an ordered tuple of attribute values in the instance space $I = D_1 \times D_2 \times \ldots \times D_n$, where D_i is the domain of the i^{th} attribute. The output of the classifier is a class label drawn from a set C of mutually exclusive classes. A training example is a labeled instance in the form $\langle X_i, c(X_i)\rangle$ where $X_i \in I$ and $c : I \rightarrow C$ is unknown function that assigns to an instance X_i its corresponding class label $c(X_i)$. For simplicity we consider only the binary classification problem in which $C = \{-1, 1\}$. Given a collection of training examples, $E = \{\langle X_1, c(X_1)\rangle, \ldots, \langle X_n, c(X_n)\rangle\}$, the goal of the (single instance) learner to learn a function c^* that approximates c as well as possible (as measured by some pre-specified performance criterion, e.g., accuracy of classification).

The MIL problem involves training a classifier to label *bags* of instances (as opposed to individual instances as is usually the case in the standard supervised learning scenario). Let $B = \{B_1, B_2, \ldots, B_m\}$ be a collection of bags. Let $B_i = \{X_{i1}, X_{i2}, \ldots, X_{ik_i}\}$ denote a bag of k_i instances ($k_i \geq 1$). The set of multiple instance training examples, E_{MI}, is a collection of ordered pairs $\langle B_i, f(B_i)\rangle$ where f is unknown function that assigns to each bag B_i a class label $f(B_i) \in \{-1, 1\}$. Under the standard multiple instance learning assumption (Dietterich et al., 1997), $f(B_i) = -1$ iff $\forall j \in \{1 \cdots k_i\}$, $c(X_{ij}) = -1$; and $f(B_i) = 1$ iff $\exists j \in \{1 \cdots k_i\}$, such that $c(X_{ij}) = 1$. Given E_{MI}, a collection of MI training examples, the goal of an multiple instance learner is to learn as good an approximation of the function f as possible (as measured by some pre-specified performance measure e.g., accuracy of classification of bags).

Dietterich et al. (Dietterich et al., 1997) proposed a solution to the MIL problem under the standard MIL assumption using a hypothesis space of axis-parallel rectangles (see Figures 1 and 2). Figure 1 (adapted from (Dietterich et al., 1997)) shows a schematic diagram of the MIL problem wherein instances are represented as points in a two dimensional Euclidean instance space. Instances that belong to the same bag are shown using the same shape. Unfilled shapes represent instances that belong to the positively labeled bags; filled shapes represent instances that belong to the negatively labeled bags. An axis parallel rectangle is used to classify bags as follows: a bag is assigned a positive label if at least one of its instances is contained within the rectangle; and a negative label otherwise. In this setting, given a set of labeled bags, the goal of the MIL algorithm is to identify an axis parallel rectangle that includes at least one unfilled point of each shape (i.e., at least one positively labeled instance from each positively labeled bag) and does not include any filled points (i.e., instances from negatively labeled bags). Such

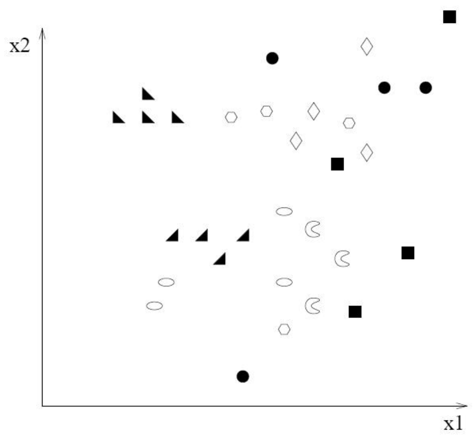


Figure 1 A multiple instance classifier learning problem. Unfilled shapes represent instances from positively labeled bags; filled shapes represent instances from negatively labeled bags. Instances extracted from the same bag are shown using the same shape. This figure is adapted from Figure 14 in (Dietterich et al., 1997).

a solution is shown in Figure 2.

Subsequently, many solutions to the MIL problem and its variants have been investigated in the literature. Ramon and De Raedt (Ramon and De Raedt, 2000) introduced a variant of the back-propagation algorithm for training a neural network for multiple instance classification problem. Wang and Zucker (Wang and Zucker, 2000) proposed variants of the k-nearest neighbor (k-NN) algorithm. Maron and Lozano-Perez (Maron and Lozano-Perez, 1998) introduced the *diverse density* (DD) framework for solving multiple instance classifier learning problems. The basic idea behind the DD method is to locate a point in the feature space that is close to at least one instance from every positive bag and as far away as possible from instances in the negative bags. Zhang and Goldman proposed EM-DD (Zhang and Goldman, 2001) which improves on DD by using Expectation Maximization (EM). The difficulty of MIL comes from the ambiguity of not knowing which of the instances in a bag is most likely to be responsible for its positive label. EM-DD models the mapping of instances to labels assigned to the bag using a set of *hidden variables*, which are estimated using the EM. EM-DD starts with an initial guess of the solution (obtained using original DD algorithm), and refines the

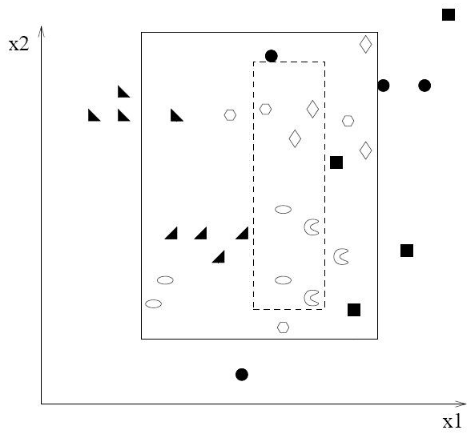


Figure 2 Solving the MIL problem using axis parallel rectangles (APR). The solid rectangle represents the initial solution, a rectangle that covers all instances that belong to positively labeled bags. The dashed rectangle represents the final APR solution, a rectangle that covers at least one positive instance from each positively labeled bag and no instances from negatively labeled bags. This figure is adapted from Figure 15 in (Dietterich et al., 1997).

guess by applying EM. Andrews et al. (Andrews et al., 2003) and Gartner et al. (Gartner et al., 2002) have proposed adaptations of support vector machines that involve changing the objective function or the kernel function to suit the multiple instance classification problem. Ray and Craven (Ray and Craven, 2005) compared several multiple instance classifier learning algorithms as well as their standard supervised learning counterparts. Scott et al. (Scott et al., 2005) introduced a generalization of the multiple instance learning model in which all of the instances in a bag are used to determine its label. Tao et al. (Tao et al., 2008) have explored kernel functions for the generalized multiple instance learning problem.

In addition to the drug discovery problem (Dietterich et al., 1997), MIL algorithms have been used, with varying degrees of success, on a number of practical problems including: content-based image retrieval (CBIR) (Maron and Ratan, 1998; Zhang et al., 2002) in which each image is viewed as a bag of objects (image regions) and an image is assigned a label based on the presence or absence of specific objects; web page classification (Zhou et al., 2005) in which each web page is modeled by a bag of pages that it links to, and is labeled positive based on the user's interest in at least one of the pages that a given page links to; and computer-aided diagnosis (Fung et al., 2007) in which each medical case is modeled by a bag of medical images (e.g., CT scans, X-ray, MRI etc) and is labeled positively if at least one of these medical images indicate malignant tumors and lesions.

The multiple instance regression (MIR) problem is a generalization of the MIL problem where each bag is labeled with a real number (as opposed to a discrete class label). Several MIR algorithms have been reported in the literature including (Ray and Page, 2001; Zhang and Goldman, 2001; Goldman and Scott, 2003).

MIL formulation of the MHC-II binding peptide prediction problem

We now proceed to introduce an MIL formulation of the variable length MHC-II binding peptide prediction problem.

Recall that a 9-mer core region is believed to be essential for MHC-II binding (Madden, 1995; Rammensee et al., 1995). We represent each variable length MHC-II peptide sequence by a bag of all 9-mer subsequences extracted from it. Under the *standard MIL assumption*, we assign a positive label to a bag of 9-mers extracted from an MHC-II binding peptide; and a negative label to a bag of 9-mers extracted from a non MHC-II binding peptide. Figure 3 shows an example of an MHC-II binding peptide and its mapping into a bag of 9-mer subsequences. It should be noted that labels are associated with bags of 9-mers, and not individual 9-mers. Consequently, in preparing the training data, we do not need to know the 9-mer binding cores.

The proposed MIL formulation of the problem of predicting MHC-II binding peptide offers several advantages over most of the existing MHC-II prediction methods:

(a) Unlike in the case of MHC-II binding prediction methods designed that work with fixed length peptides, there is no need to know the precise location of 9-mer binding cores in MHC-II binding peptides in the training data.

(b) Unlike MHC-II binding prediction methods that use only the positively labeled data to determine a 9-mer binding core motif, our method makes use of both positive (i.e., 9-mers extracted from binders) and negatively labeled data (i.e., 9-mers extracted from

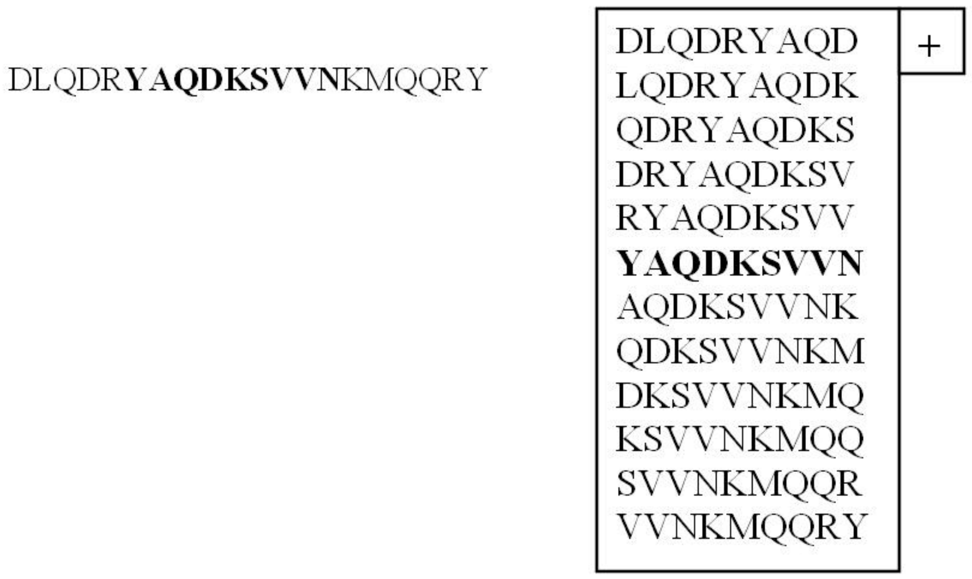


Figure 3 An example of an MHC-II binding peptide and its corresponding multiple instance bag. Bold subsequence indicates the 9-mer binding core. Mapping the peptide sequence into a bag does not require the identification of the 9-mer binding core because no labels are associated with the instances of the bag.

non-binders) to determine such binding core such that the performance of the classifier is optimize.

(c) The resulting classifier can predict the MHC-II binding activity of a peptide of virtually any length (because the input to the classifier is a bag (multiset) of all 9-mers extracted from the peptide).

(d) The resulting classifier not only predicts the MHC-II binding activity of flexible length MHC-II peptides but also predicts the 9-mer binding core.

The problem of learning to predict the MHC-II binding affinities of flexible length peptides can be formulated as a multiple instance regression problem in a manner similar to that described above for the classification setting, simply by mapping each peptide to a bag of 9-mers and substituting the class labels with the measured real-valued binding affinities for each peptide.

In summary, both qualitative and quantitative predictions of the MHC-II binding activity of peptides can be obtained using predictive models based on the multiple instance formulations of the corresponding classification and regression problems (respectively). The resulting multiple instance problems can be solved using the corresponding multiple instance learning algorithms or multiple instance regression algorithms available in the literature. In this paper, we focus on the quantitative prediction of the binding activity of MHC-II peptides using a multiple instance regression algorithm.

Materials and Methods

Data

We used the IEDB benchmark dataset introduced in (Nielsen et al., 2007) in our experiments. The dataset consists of peptides along with their IC50 binding affinities for 14 HLA-DR and three H2-IA alleles (hereafter referred to as IEDB dataset for short). Details of the IEDB benchmark dataset are summarized in Table 1. Because each peptide is labeled with its binding affinity (IC50) value, peptides were categorized into binders and non-binders using a binding affinity threshold of 500 nM (Nielsen et al., 2007). To avoid overly optimistic estimates of the performance of MHC-II binding peptide prediction methods, it is important to ensure that the peptide sequences used to evaluate the performance of the predictor do not share a high degree of sequence overlap (or similarity) with peptide sequences in the training set used to train the predictor. Nielsen et al. (2007) have partitioned each IEDB allele dataset into five subsets so as to minimize the degree of sequence overlap between any pair of subsets. In our experiments, we used this partitioning of each MHC-II allele dataset in our 5-fold cross validation experiments. Following (Rajapakse et al., 2007), we excluded the DRB3-0101 MHC-II allele dataset from our experiments because of its highly skewed distribution (only 3 binders as opposed to 99 non-binders). Recall that in 5-fold cross validation, the dataset has to be partitioned into 5 non-overlapping subsets. With only 3 MHC-II binders in the dataset, at least 2 of the 5 subsets will have 0 positive instances. Consequently, the computation of AUC is meaningless in such a setting. Therefore, if we were to report AUC estimated using 5-fold cross-validation on the DRB3-0101 dataset we would need to either exclude the 2 subsets with 0 positive instances or by combine the predictions over the 5 subsets and compute a single value for the AUC.

MHCMIR method

In order to explore the feasibility of predicting MHC-II binding activity of peptides based on the proposed multiple instance regression formulation, we developed MHCMIR, a novel method for predicting the binding affinity of MHC-II peptides using multiple instance regression. Given a dataset of MHC-II peptides where each peptide is labeled with its experimentally determined binding affinity (IC50 value), MHCMIR maps each peptide to its corresponding bag of 9-mers and uses the data in its multiple instance representation to train a multiple instance regression model. The learned multiple instance regression model can be used to predict the affinity of any query peptide by providing as input to the model the bag of 9-mers representation of the query peptide sequence.

In this study, we chose to adapt an existing multiple instance learning algorithm, MILES (multiple instance learning via embedded selection) (Chen et al., 2006), to work in the regression (as opposed to classification) setting. The original algorithm, MILES, maps each bag of instances into a meta instance constructed by applying an Euclidean distance based similarity measure to instances within each bag. Then, a 1-norm SVM classifier (Zhu et al., 2004) is trained on the resulting dataset of meta instances. The competitive performance of MILES, and its low computational cost of training are some of its main advantages relative to other MIL algorithms (Chen et al., 2006).

Adapting the MILES algorithm for training a multiple instance classifier into a multiple instance regression algorithm is rather straightforward. All we need to do is to replace the 1-norm SVM classifier by a support vector regression (SVR) model (Shevade et al., 2000). Because in our application, the bags to be labeled are comprised of 9-mers over the amino-acid alphabet, we replaced the Euclidean distance used in MILES for transforming a bag of instances into a meta instance by a distance function that is customized for calculating the distance between amino acid sequences. This distance function is based on the BLOSUM62 amino acid substitution matrix (Henikoff and Henikoff, 1992).

The pseudocode shown in Algorithm 1 summarizes MILESreg, our proposed multiple instance regression algorithm. The function $dist(s1, s2)$ computes the distance between two 9-mers, $s1$ and $s2$. Note that $BLOSUM62(aa1, aa2)$ is the corresponding BLOSUM62 matrix entry for the amino acids $aa1$ and $aa2$ and $s[i]$ denotes the amino acid in the i^{th} position in the sequence s.

Algorithm 1 Training MILESreg

```
1: Input : $B = \{\langle B_1, y_1 \rangle, \dots \langle B_m, y_m \rangle\}$ set of training bags
2: Let $C = \{x^1, \dots, x^n\}$ be set of all instances extracted from $B$
3: for all $i$ such that $\langle B_i, y_i \rangle \in B$ do
4:     Let $I_i$ be a new instance of $n$ attributes
5:     for all $k$ such that instance $x^k \in C$ do
6:         Set $k^{th}$ attribute in $I_i$ to $min_j\ dist(x_{ij}, x^k)$
7:     end for
8: end for
9: Build an SVR model using meta instances $I$
10:
11: Function: dist
12: Parameters : $s1$ and $s2$ two 9-mer subsequences
13: for $i = 1$ to 9 do
14:     $d+ = BLOSUM62(s1[i], s2[i])$
15: end for
16: if $(d \leq 0)$ then
17:     return 1
18: else
19:     return $\frac{1}{d}$
20: end if
```

Predicting the label of a test bag B_i is performed in two steps. First, B_i is mapped into a meta instance using the set of training instances C and the procedure described in lines 3 to 6 in the pseudocode. Then, a predicted real value is assigned to the meta-instance using the learned support vector regression model.

Results and Discussion

We compared the predictive performance of MHCMIR with that of several MHC-II binding peptide prediction methods reported in the literature: Gibbs sampler (Nielsen et al., 2004), TEPITOPE (Sturniolo et al., 1999), SVRMHC (Liu et al., 2006), MHCPred (Hattotuwagama et al., 2004), ARB (Bui et al., 2005), NetMCHII (Nielsen et al., 2007), and MOEA (Rajapakse et al., 2007). Because most reports of MHC-II binding activity prediction methods in the literature focus on qualitative prediction of MHC-II binding activity, although MHCMIR is able to produce both quantitative and qualitative predictions of MHC-II binding activity (the latter by comparing the predicted binding affinity value with a threshold), our comparisons focus on qualitative predictions of MHC-II binding activity. Specifically, we compared the estimated area under ROC curve (AUC) (Swets, 1988) for the different methods. In the case of MHCMIR, Gibbs sampler, NetMHCII (Nielsen et al., 2007), and MOEA (Rajapakse et al., 2007) the performance estimates were obtained using 5-fold cross validation on the partitioning of each MHC-II allele dataset into 5 subsets, ensuring minimal sequence overlap between the different subsets provided by Nielsen et al. (Nielsen et al., 2007). Because the codes for the SVRMHC, MHCPred, and ARB methods are not readily available, estimates of the performance of these methods were obtained by submitting the data to the online web servers that implement the respective methods (using the default parameter setting for each server). As noted in (Nielsen et al., 2007), the reported performance of ARB method should be interpreted with caution because the ARB method has been trained on data from IEDB database (Peters et al., 2005), which, because of the overlap between the training and test data, gives it an unfair advantage over other methods.

Table 2 compares the predictive performance, in terms of AUC, of the different MHC-II peptide prediction methods. For a better visualization of the pairwise comparisons between MHCMIR and other methods reported in Table 2, we used "*" to indicate results that are outperformed by MHCMIR method. "-" indicates unavailable information either because the online server does not support the corresponding allele (e.g., SVRMHC, MHCPred, and ARB on a number of allele datasets) or because the data was not provided in the published work (e.g., detailed results of Gibbs method on the three mouse allele datasets were not provided in (Nielsen et al., 2007)). The results show that MHCMIR has AUC that is superior to that of the other methods on a majority of the MHC-II allele datasets. For example, MHCMIR outperforms both NetMHCII and MOEA, the latest MHC-II binding peptide prediction method reported in the literature, on 11 out of the 16 allele datasets. In addition, MHCMIR outperforms Gibbs Sampler method on 11 out of 13 allele datasets. It is worth noting that the MHCMIR is the only method with an estimated AUC that is greater than 0.7 on each of the datasets.

In addition to the AUC, Table 3 compares the performance of the different MHC-II peptide prediction methods as estimated by the Pearson's correlation coefficient (Urdan, 2005) between the predicted and actual labels. MOEA has not been included in this comparison because its performance has been reported using only AUC (Rajapakse et al., 2007). MHCMIR has a better correlation coefficient than NetMHCII on 8 out of the 13 HLA-DR allele datasets. Overall, the results show that MHCMIR is very competitive with the state-of-the-art methods for the same problem.

Statistical analysis

In comparing two classifiers, statistical tests can be employed to determine whether the difference in performance between the two classifiers is significant or not. For comparing multiple classifiers on multiple datasets, we followed the procedure that has recently been recommended by Demšar (2006) which involves comparing the average rank of the classifiers across the different datasets.

The statistical analysis of the performance comparisons was limited to NetMHCII, MOEA, and MHCMIR methods because these are only the methods with reported performance (AUC) on each of the allele datasets. First, the different classifiers are ranked on the basis of their observed performance on each data set (see Table 4). Then we used the Friedman test to determine whether the measured average ranks are significantly different from the mean rank under the null hypothesis. We found that at 0.05 level of significance the null hypothesis could

Table 1 Summary of the IEDB benchmark dataset. Binding peptides were identified using an IC50 binding threshold of 500 nM.

Dataset	Binders	Non-binders
DRB1-0101	920	283
DRB1-0301	65	409
DRB1-0401	209	248
DRB1-0404	74	94
DRB1-0405	88	83
DRB1-0701	125	185
DRB1-0802	58	116
DRB1-0901	47	70
DRB1-1101	95	264
DRB1-1302	101	78
DRB1-1501	188	177
DRB3-0101	3	99
DRB4-0101	74	107
DRB5-0101	112	231
H2-IAb	43	33
H2-IAd	56	286
H2-IAs	35	91

not be rejected. Hence, we concluded that the reported performances of the three methods are not significantly different.

MHCMIR has the best average rank of 1.63 while the average ranks for MOEA and NetMHCII are 2.06 and 2.31, respectively (As noted by Demšar, the average ranks of the classifiers provide a reasonably fair comparison of the classifiers (Demšar, 2006)).

Multiple Instance Learning Approaches to MHC-II prediction methods

The multiple instance learning literature offers three broad classes of approaches to MIL based on different assumptions regarding the relation between the label assigned to a bag and the labels of instances contained in the corresponding bag. In what follows, we demonstrate that the different MIL methods have parallels among the existing approaches for predicting MHC-II binding peptides. We further argue that the MIL formulation of the MHC-II binding peptide prediction problem offers several advantages over existing methods.

The first class of MIL methods is the witness-based methods (Maron and Lozano-Perez, 1998; Zhang and Goldman, 2001; Dietterich et al., 1997; Andrews et al., 2003; Mangasarian and Wild, 2005; Ray and Page, 2001) which search for a single positive instance (witness) within each positive bag. Existing MHC-II prediction methods that pre-determine a single 9-mer core within each binding peptide (Reche et al., 2004; Nielsen et al., 2004; Bui et al., 2005; Liu et al., 2006; Donnes and Kohlbacher, 2006; Nielsen et al., 2007; Rajapakse et al., 2007) can be seen as the counterparts of the witness-based MIL methods. However, the MIL formulation of flexible length MHC-II peptides proposed in this study offers two major advantages over this family of MHC-II binding peptide prediction methods:

- First, our approach does not require the pre-identification of the binding cores; the binding cores are identified as a side-effect of learning an MIR model that optimizes the predictive performance on the MHC-II binding affinity prediction task. In contrast, the performance of the MHC-II prediction methods that first identify 9-mer binding cores and then use them to construct a predictor (e.g., using weight matrix) is critically dependent on the correct identification of the 9-mer binding cores.

- Second, our approach uses training data obtained from both the binders and non-binders to identify the binding cores that maximize the predictive performance of the learned model whereas the MHC-II binding peptide prediction methods that rely on pre-identified 9-mer binding cores typically use only the data from the binders. As shown by several studies (Murugan and Dai, 2005; Nielsen et al., 2007; Rajapakse et al., 2007) including the one reported in this paper, using data from both the binders and non-binders to identify the MHC-II binding cores yields predictive performance that is superior to that of methods that rely only on the binders.

The second broad class of MIL methods is the generalized MIL methods that operate under the assumption that all instances within a bag contribute the bag label (Ray and Craven, 2005; Zhou and Zhang, 2007; Gartner et al., 2002). Two recently proposed MHC-II prediction methods (Cui et al., 2006a) and (Salomon and Flower, 2006) which train their classifiers using the entire peptide sequence can be seen as variants of the generalized MIL methods. In principle, these two SVM-based qualitative MHC-II binding peptide prediction methods can be easily adapted to obtain quantitative MHC-II predictions by replacing the SVM classifiers with support vector regression models but such an adaptation remains to be explored.

The third broad class of MIL methods is the generalized MIL methods which operate under the assumption that only a subset of the instances within a bag contribute the bag label (Chen et al., 2006; Weidmann et al., 2003). The iterative approach for predicting MHC-II peptides (Murugan and Dai, 2005) can be seen as an exemplar of this third class of MIL methods. However, the application of several additional variants of such generalized MIL methods to the problem of MHC-II binding peptide or binding affinity prediction remain to be explored.

In summary, the multiple instance based formulations of the flexible length MHC-II binding peptide and binding affinity prediction problems introduced in this paper open up, in light of the parallels between MIL methods and MHC-II binding peptide prediction methods, the possibility of adapting several other MIL and MIR based methods in this setting.

Conclusion

We have introduced a novel formulation of the problem of learning to predict variable length MHC-II binding peptides as an instance of a multiple instance learning problem. Unlike many existing MHC-II binding peptide prediction methods, the proposed method does not require the pre-identification of the 9-mer core region in each binding peptide prior to training the model. Moreover, the 9-mer binding cores are determined (during the training of the multiple instance learning algorithm) from both binding and non-binding training data such that the predictive performance of the learned model is maximized.

Based on our proposed formulation, we introduced a novel method, MHCMIR, for predicting the binding affinity of variable length MHC-II peptides. MHCMIR utilizes MILESreg, our adaptation of MILES MIL classification method, for performing multiple instance regression on bags of amino acid sequences. Our experiments on the largest available benchmark dataset covering 13 HLA-DR alleles and three H2-IA alleles have shown that MHCMIR is quite competitive with the state-of-the-art methods for predicting MHC-II binding peptides. Our formulation of the problems of qualitatively and quantitatively predicting flexible length

MHC-II peptides as multiple instance learning and multiple instance regression problems, respectively, has opened up the possibility of adapting a broad range of multiple instance methods for classification and regression in this setting.

We have made our implementation of MHCMIR freely available to the scientific community in the form of an online web server for predicting the binding affinity of MHC-II peptides. The server can be accessed at http://ailab.cs.iastate.edu/mhcmir.

Acknowledgment

This work was supported in part by a doctoral fellowship from the Egyptian Government to Yasser EL-Manzalawy and a grant from the National Institutes of Health (GM066387) to Vasant Honavar and Drena Dobbs.

Table 2 Comparison of AUC values for the different MCH-II prediction methods on the IEDB benchmark dataset. Results for Gibbs, TEPITOPE, SVRMHC, MHCPred, ARB, and NetMHCII are taken from (Nielsen et al., 2007). Results of MOEA are obtained from (Rajapakse et al., 2007). The results of the best-performing method are highlighted in bold. "-" indicates that the corresponding entry is currently unavailable. "*" indicates results that are outperformed by MHCMIR.

Dataset	Gibbs	TEPITOPE	SVRMHC	MHCpred	ARB	NetMHCII	MOEA	MHCMIR
DRB1-0101	0.676*	0.647*	0.623*	0.565*	0.666*	0.716*	0.651*	**0.780**
DRB1-0301	0.722*	0.734*	-	-	**0.799**	0.765*	0.778	0.772
DRB1-0401	0.759*	0.754*	0.739*	0.606*	0.737*	0.758*	0.725*	**0.774**
DRB1-0404	0.743*	**0.829**	-	-	0.788*	0.785*	0.786*	0.806
DRB1-0405	0.724	**0.790**	0.701*	-	0.724	0.735	0.756	0.715
DRB1-0701	0.695*	0.768	-	0.647*	0.749	**0.787**	0.735*	0.744
DRB1-0802	0.721*	0.769*	-	-	**0.803**	0.756*	0.773*	0.779
DRB1-0901	0.734	-	-	-	0.711*	**0.775**	0.712*	0.713
DRB1-1101	0.715*	0.710*	-	-	0.727*	0.734*	**0.759**	0.758
DRB1-1302	0.716*	0.720*	-	-	**0.917**	0.818*	0.820*	0.850
DRB1-1501	0.672*	0.726*	0.730*	-	**0.792**	0.736*	0.743*	0.781
DRB4-0101	0.742*	-	-	-	0.800*	0.736*	0.759*	**0.821**
DRB5-0101	0.618*	0.653*	0.649*	-	0.677*	0.664*	0.660*	**0.708**
H2-IAb	-	-	-	-	0.662*	0.908*	0.919*	**0.924**
H2-IAd	-	-	-	-	0.819	0.818	**0.855**	0.791
H2-IAs	-	-	-	-	-	**0.898**	0.889	0.843

Table 3 Comparison of Pearson's correlation coefficient values for the different MCH-II prediction methods on the IEDB benchmark dataset. Results for Gibbs, TEPITOPE, SVRMHC, MHCPred, ARB, and NetMHCII are obtained from (Nielsen et al., 2007). The results of the best-performing method are highlighted in bold. "-" indicates that the corresponding entry is currently unavailable. "*" indicates results that are outperformed by MHCMIR.

Dataset	Gibbs	TEPITOPE	SVRMHC	MHCpred	ARB	NetMHCII	MHCMIR
DRB1-0101	0.260*	0.333*	0.213*	0.146*	0.376*	0.413*	**0.510**
DRB1-0301	0.453*	0.227*	-	-	**0.506**	0.466*	0.487
DRB1-0401	0.482*	0.508*	0.461*	0.176*	0.434*	0.499*	**0.513**
DRB1-0404	0.433*	**0.609**	-	-	0.529	0.481	0.456
DRB1-0405	0.428	**0.542**	0.409*	-	0.420	0.417	0.412
DRB1-0701	0.353*	0.460*	-	0.309*	0.410*	**0.531**	0.463
DRB1-0802	0.375*	0.472	-	-	**0.517**	0.461	0.456
DRB1-0901	0.398	-	-	-	0.440	**0.487**	0.378
DRB1-1101	0.385*	0.382*	-	-	0.421*	0.426*	**0.454**
DRB1-1302	0.400*	0.411*	-	-	**0.763**	0.594*	0.622
DRB1-1501	0.305*	0.453*	0.481*	-	**0.561**	0.461*	0.552
DRB4-0101	0.417*	-	-	-	0.507*	0.403*	**0.531**
DRB5-0101	0.288*	0.313*	0.322*	-	0.330*	0.347*	**0.362**
H2-IAb	-	-	-	-	-	-	0.700
H2-IAd	-	-	-	-	-	-	0.398
H2-IAs	-	-	-	-	-	-	0.535

Table 4 AUC values for NetMHCII, MOEA, and MHCMIR methods evaluated on IEDB benchmark datasets. For each dataset, the rank of each classifier is shown in parentheses.

Dataset	NetMHCII	MOEA	MHCMIR
DRB1-0101	0.716(2)	0.651(3)	0.780(1)
DRB1-0301	0.765(3)	0.778(1)	0.772(2)
DRB1-0401	0.758(2)	0.725(3)	0.774(1)
DRB1-0404	0.785(3)	0.786(2)	0.806(1)
DRB1-0405	0.735(2)	0.756(1)	0.715(3)
DRB1-0701	0.787(1)	0.735(3)	0.744(2)
DRB1-0802	0.756(3)	0.773(2)	0.779(1)
DRB1-0901	0.775(1)	0.712(3)	0.713(2)
DRB1-1101	0.734(3)	0.759(1)	0.758(2)
DRB1-1302	0.818(3)	0.820(2)	0.850(1)
DRB1-1501	0.736(3)	0.743(2)	0.781(1)
DRB4-0101	0.736(3)	0.759(2)	0.821(1)
DRB5-0101	0.664(2)	0.660(3)	0.708(1)
H2-IAb	0.908(3)	0.919(2)	0.924(1)
H2-IAd	0.818(2)	0.855(1)	0.791(3)
H2-IAs	0.898(1)	0.889(2)	0.843(3)
Avg. ranks	0.774(2.31)	0.770(2.06)	0.785(1.63)

CHAPTER 7. GENERAL CONCLUSIONS

Summary and discussion

Computational methods for reliably identifying potential vaccine candidates (i.e., epitopes that invoke strong response from both T-cells and B-cells) are highly desirable. Unfortunately, the predictive performance of such prediction tool is still far from ideal. Machine learning offers one of the most cost-effective and widely used approaches to developing epitope prediction tools. We have proposed several machine learning based methods for epitope prediction using only amino acid sequence information. First, we have introduced a method, BCPred, for predicting linear B-cell epitopes using the subsequence kernel. Our results have shown that BCPred significantly outperforms several other SVM based classifiers and a number of existing linear B-cell epitope prediction methods.

One of the challenges for developing reliable linear B-cell epitope predictors is how to deal with the large variability in the length of the epitopes which ranges from 3 to 30 amino acids in length. Previous machine learning based linear B-cell epitope prediction methods, including BCPred, require training and testing the classifier using sequences of fixed length. We have constructed the first flexible length linear B-cell epitopes and explored two different approaches for training classifiers using variable length amino acid sequences. Based on our results, we have proposed FBCPred, a novel method for predicting flexible length linear B-cell epitopes using the subsequence kernel. Unlike other linear B-cell epitope prediction methods, FBCPred can predict linear B-cell epitopes of virtually any specified length.

For predicting MHC-I binding peptides, matrix based methods are fairly believed to be less efficient than machine learning based methods due to the inability of many matrix based methods to modeling the correlations between different positions in the learned model. We have presented a comparative study where we have directly compared an extensive number of machine learning and matrix based MHC-I predictors. Unlike previous comparison studies comparing different MHC-I prediction servers (note that these servers have been developed using different training data), our study provides a direct comparison of different prediction methods using a unified experimental setup (e.g., all methods were trained and evaluated using the same training and test sets, respectively). The results have shown that AOMM and SMMBin, two matrix based methods that we have proposed in this study, were highly competitive with a broad class of machine learning methods for predicting MHC-I peptides.

For predicting MHC-II binding peptides, we have shown that the performance of many MHC-II binding peptide prediction methods reported in the literature is substantially overly-optimistic because the performance of such methods had been estimated using data sets of unique peptides without applying any similarity reduction procedure to eliminate highly similar peptides. Because MHC-II peptides have lengths that vary over a broad range, similarity reduction of MHC-II peptides is not a straightforward task. We have shown that the previously reported similarity reduction methods may not eliminate highly similar peptides, i.e., peptides that share $> 80\%$ sequence identity still pass the similarity test. We have proposed a two-step similarity reduction procedure that is much more stringent than those currently in use for similarity reduction with MHC-II benchmark datasets. We have introduced three similarity-reduced MHC-II benchmark data sets derived from MHCPEP (Brusic et al., 1998), MHCBN (Bhasin et al., 2003), and IEDB (Peters et al., 2005) databases and have utilized them in our experiments to show the pitfalls of these commonly used data sets for evaluating the performance of machine learning approaches to MHC-II peptide binding predictions. Finally, we have formulated the problems of qualitatively and quantitatively predicting flexible length MHC-II peptides as multiple instance learning and multiple instance regression problems, respectively. Based on this formulation, we have introduced MHCMIR, a novel method for predicting MHC-II binding affinity using multiple instance regression. We have presented results of experiments using a benchmark dataset covering 13 HLA-DR and three H2-IA alleles that showed that MHCMIR is competitive with the state-of-the-art methods for predicting MHC-II binding peptides.

It is our hope that the results of this dissertation and our freely available benchmark data sets and prediction servers will contribute to a better understanding of the dynamics of the adaptive immune system and will facilitate more advances in the epitope prediction problem which is a major challenge in Immunoinformatics.

Contributions

This dissertation has provided several contributions that can be categorized into three categorizies:

- Algorithmic
  - **AUC optimized matrix method (AOMM)**, an algorithm for finding a position specific scoring matrix (PSSM) that maximizes the AUC over the training data.
  - **Modified PSSM (MPSSM)**, a variant of the PSSM method (Henikoff and Henikoff, 1996) that utilizes motif and non-motif sequences in building the PSSM from the provided training data.
  - **Qualitative via quantitative (QVQ)**, an approach for building a qualitative scoring matrix using a quantitative matrix method.
  - **A formalization of the problem of qualitatively and quantitatively predicting flexible length major histocompatibility complex class II (MHC-II) peptides as multiple instance learning and multiple instance regression problems, respectively**.
  - **MILESreg**, an adaptation of MILES (Chen et al., 2006) for multiple instance regression over bags of amino acid sequences.
- Prediction servers and software

  - **BCPREDS**, a web server for predicting linear B-cell epitopes. The current implementation supports three B-cell epitope prediction methods: (i) BCPred (EL-Manzalawy et al., 2008d); (ii) FBCPred (EL-Manzalawy et al., 2008b); (iii) AAP (Chen et al., 2007). The server is freely accessible at `http://ailab.cs.iastate.edu/bcpreds/`
  - **MHCIPREDS**, a web server for predicting MHC-I peptides using a number of qualitative and quantitative MHC-I peptide prediction methods. The current implementation of the server provides predictions for the 22 MHC-I HLA alleles. The server is freely accessible at `http://ailab.cs.iastate.edu/mhcipreds/`
  - **MHCMIR**, a web server for predicting MHC-II binding affinities using multiple instance regression. The current version supports 13 HLA-DR alleles and three mouse H2-IA alleles. The server is freely accessible at `http://ailab.cs.iastate.edu/mhcmir/`
  - **WLSVM**, a wrapper for integrating LibSVM (Chang and Lin, 2001) into Weka framework (Witten and Frank, 2005). WLSVM has been contributed and integrated into Weka since version 3.5.2.
  - **MPSSM**, a Java program implementing our proposed MPSSM method. The program is available upon request from the author.
  - **AOMM**, a Java program implementing our proposed AOMM method. The program is available upon request from the author.

- Benchmark data sets
  - **BCPred data sets**, 10 homology-reduced data sets used in the evaluation and implementation of BCPred method. To the best of our knowledge these are the first and only available similarity-reduced linear B-cell epitope data sets. The data sets can be downloaded from the BCPREDS server.
  - **FBCPred data sets**, 2 flexible length linear B-cell epitope data sets used in the evaluation and implementation of FBCPred method. To the best of our knowledge

these are the first flexible length linear B-cell epitope data sets. The data sets can be downloaded from the BCPREDS web server.

- **MHC-I data sets**, 22 HLA MHC-I allele-specific similarity-reduced data sets. The data is available in two version, qualitative and quantitative, and can be downloaded from the MHCIPREDS web server.
- **MHC-II data sets**, three benchmark data sets derived from MHCPEP (Brusic et al., 1998), MHCBN (Bhasin et al., 2003), and IEDB (Peters et al., 2005) databases using different similarity reduction methods. The data sets and the similarity reduction scripts can be downloaded from PLoS ONE web site, `http://www.plosone.org/article/info:doi%2F10.1371%2Fjournal.pone.0003268#s5`, or be requested from the author.

Future work

This dissertation has provided new machine learning based methods for attacking three important epitope prediction related problems, predicting linear B-cell epitopes and predicting both MHC-I and MHC-II binding peptides. From our study and several other related studies, it seems that the problems of predicting MHC-II binding peptides and linear B-cell epitopes are more challenging than the problem of predicting MHC-I binding peptides. The following are some potential directions for future studies.

Predicting conformational B-cell epitopes

In this dissertation, we have focused on predicting linear B-cell epitopes using amino acid sequence information. Although it is believed that a large majority of B-cell epitopes are discontinuous (Walter, 1986), experimental epitope identification has focused primarily on linear B-cell epitopes (Flower, 2007). Because the number of available antigen-antibody complexes in protein data bank (PDB) is limited, only few methods for predicting conformational B-cell epitopes using structure information have been proposed. As enough data becomes available, the development of reliable conformational B-cell epitope prediction tools is expected to gain more interest.

Predicting sub-types of linear B-cell epitopes

One approach of improving the performance of linear B-cell epitopes is to develop predictors that focus on a sub-type of linear B-cell epitopes (e.g., predicting protective linear B-cell epitopes (Söllner et al., 2008; EL-Manzalawy et al., 2008c)).

Improved prediction of MHC-II binding peptides using MIL/MIR methods

Our formulation of MHC-II binding peptide prediction problem as a multiple instance learning problem has opened up the possibility of adapting a broad range of multiple instance learning methods for classification and regression in this setting. Several avenues for further improving the performance of MHCMIR could be explored: i) Expanding the coverage to more MHC-II alleles; ii) Incorporating feature selection, feature abstraction, and dimensionality reduction methods to reduce redundant and irrelevant features from the meta instance data used to build the support vector regression model; iii) Exploring other regression methods (e.g. Gaussian process (MacKay, 1998)) for building the regression model from the meta instance data.

Exploring the application of the approaches and methods presented in this study in several other Bioinformatics problems

Examples may include: i) The application of the scoring matrix methods, MPSSM, AOMM, and SMMBin, in any Bioinformatics application where the scoring matrix approach is applicable (e.g., predicting post translational modification sites (Caragea et al., 2007; Yang, 2007; Xue et al., 2008)); ii) Implementing each residue in a protein sequence as a bag of its spatial residues, where each spatial residue could be represented using a set of structure and physicochemical features. Using this representation, multiple instance learning methods may be applied for predicting functional sites using structure and sequence information (e.g., predicting conformational B-cell epitopes (Kulkarni-Kale et al., 2005; Haste Andersen et al., 2006) or protein-RNA interface residues (Terribilini et al., 2006)).

APPENDIX SUPPLEMETARY MATERIALS FOR CHAPTER 2

Table A.1 Performance of different methods on our BCP18 *homology-reduced* data set using 5-fold cross validation. BCPred method denotes $K^{sub}_{(4,0.5)}$.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
K^{spct}_1	55.08	53.12	57.05	0.102	0.588
K^{spct}_2	59.28	60.57	57.99	0.186	0.636
K^{spct}_3	64.70	65.99	63.41	0.294	0.675
$K^{msmtch}_{(3,1)}$	46.75	46.34	47.15	-0.065	0.465
$K^{msmtch}_{(4,1)}$	57.79	57.72	57.86	0.156	0.599
$K^{msmtch}_{(5,1)}$	64.77	61.92	67.62	0.296	0.691
$K^{msmtch}_{(5,2)}$	55.01	56.50	53.52	0.100	0.568
LA	64.43	62.87	65.99	0.289	0.691
$K^{sub}_{(2,0.5)}$	61.52	61.92	61.11	0.230	0.668
$K^{sub}_{(3,0.5)}$	66.80	**69.38**	64.23	0.337	0.726
BCPred	**69.04**	65.72	72.36	**0.382**	**0.751**
RBF	56.98	57.99	55.96	0.140	0.601
AAP	66.94	56.91	**76.96**	0.346	0.699

Table A.2 Performance of different methods on our BCP16 *homology-reduced* data set using 5-fold cross validation. BCPred method denotes $K^{sub}_{(4,0.5)}$.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
K^{spct}_{1}	60.01	60.22	59.81	0.200	0.652
K^{spct}_{2}	58.99	59.54	58.45	0.180	0.612
K^{spct}_{3}	61.17	62.53	59.81	0.224	0.645
$K^{msmtch}_{(3,1)}$	47.55	47.00	48.09	-0.049	0.460
$K^{msmtch}_{(4,1)}$	54.36	52.86	55.86	0.087	0.569
$K^{msmtch}_{(5,1)}$	64.24	61.58	66.89	0.285	0.667
$K^{msmtch}_{(5,2)}$	54.70	55.18	54.22	0.094	0.563
LA	63.15	63.49	62.81	0.263	0.686
$K^{sub}_{(2,0.5)}$	63.76	63.62	63.90	0.275	0.681
$K^{sub}_{(3,0.5)}$	65.53	67.98	63.08	0.311	0.718
BCPred	**65.94**	**74.93**	56.95	**0.324**	**0.730**
RBF	57.29	56.81	57.77	0.146	0.594
AAP	65.05	60.90	**69.21**	0.302	0.689

Table A.3 Performance of different methods on our BCP14 *homology-reduced* data set using 5-fold cross validation. BCPred method denotes $K^{sub}_{(4,0.5)}$.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
K^{spct}_{1}	55.66	54.76	56.56	0.113	0.582
K^{spct}_{2}	57.07	56.43	57.71	0.141	0.597
K^{spct}_{3}	65.75	66.71	64.78	0.315	0.675
$K^{msmtch}_{(3,1)}$	50.06	49.23	50.90	0.001	0.506
$K^{msmtch}_{(4,1)}$	57.07	55.53	58.61	0.142	0.596
$K^{msmtch}_{(5,1)}$	63.11	65.04	61.18	0.262	0.649
$K^{msmtch}_{(5,2)}$	55.98	54.24	57.71	0.120	0.574
LA	62.98	62.47	63.50	0.260	0.671
$K^{sub}_{(2,0.5)}$	60.48	60.41	60.54	0.210	0.647
$K^{sub}_{(3,0.5)}$	63.88	66.71	61.05	0.278	0.697
BCPred	**64.78**	**73.14**	56.43	**0.300**	**0.733**
RBF	57.65	58.35	56.94	0.153	0.603
AAP	61.38	55.40	**67.35**	0.229	0.665

Table A.4 Performance of different methods on our BCP12 *homology-reduced* data set using 5-fold cross validation. BCPred method denotes $K^{sub}_{(4,0.5)}$.

Method	ACC(%)	Sn(%)	Sp(%)	CC	AUC
K^{spct}_1	55.60	53.80	57.40	0.112	0.591
K^{spct}_2	57.21	58.17	56.24	0.144	0.606
K^{spct}_3	61.00	62.03	59.97	0.220	0.636
$K^{msmtch}_{(3,1)}$	44.98	45.05	44.92	-0.100	0.450
$K^{msmtch}_{(4,1)}$	53.47	53.80	53.15	0.070	0.548
$K^{msmtch}_{(5,1)}$	56.89	65.51	48.26	0.140	0.594
$K^{msmtch}_{(5,2)}$	53.41	52.38	54.44	0.068	0.535
LA	61.20	61.13	61.26	0.224	0.662
$K^{sub}_{(2,0.5)}$	59.91	61.00	58.82	0.198	0.643
$K^{sub}_{(3,0.5)}$	62.74	63.96	61.52	0.255	0.687
BCPred	**65.83**	**53.80**	77.86	**0.326**	**0.709**
RBF	59.33	60.36	58.30	0.187	0.620
AAP	64.22	69.50	**58.94**	0.286	0.663

Table A.5 BCPred predictions on RBD of SRAS-CoV S protein.

Position	Epitope	Score
142	PFSPDGKPCTPPALNC	1
159	WPLNDYGFYTTTGIGY	0.992
105	AWNTRNIDATSTGNYN	0.974
18	PSVYAWERKKISNCVA	0.946

Table A.6 AAP predictions on RBD of SRAS-CoV S protein.

Position	Epitope	Score
68	DSFVVKGDDVRQIAPG	1
140	NVPFSPDGKPCTPPAL	1
17	FPSVYAWERKKISNCV	1
166	FYTTTGIGYQPYRVVV	1
114	TSTGNYNYKYRYLKHG	1

Table A.7 Bepipred predictions on RBD of SARS-CoV S protein.

No.	Start Position	End Position	Peptide	Peptide Length
1	17	17	F	1
2	72	72	V	1
3	75	88	DDVRQIAPGQTGVI	14
4	93	95	YKL	3
5	110	120	NIDATSTGNYN	11
6	133	133	P	1
7	136	154	RDISNVPFSPDGKPCTPPA	19
8	166	167	FY	2
9	171	174	GIGY	4

Table A.8 ABCPred predictions on RBD region of SARS-CoV S protein.

Rank	Sequence	Start position	Score
1	MGCVLAWNTRNIDATS	100	0.93
2	TTTGIGYQPYRVVVLS	168	0.87
3	TSTGNYNYKYRYLKHG	114	0.86
4	DVRQIAPGQTGVIADY	76	0.83
4	PALNCYWPLNDYGFYT	153	0.83
5	TNLCPFGEVFNATKFP	3	0.82
5	DGKPCTPPALNCYWPL	146	0.82
5	DISNVPFSPDGKPCTP	137	0.82
5	TRNIDATSTGNYNYKY	108	0.82
6	KFPSVYAWERKKISNC	16	0.8
7	TGVIADYNYKLPDDFM	85	0.79
7	CFSNVYADSFVVKGDD	61	0.79
8	YRYLKHGKLRPFERDI	123	0.78
9	NCVADYSVLYNSTFFS	30	0.77
10	FSTFKCYGVSATKLND	44	0.76
11	ATKLNDLCFSNVYADS	54	0.74
12	FVVKGDDVRQIAPGQT	70	0.73
13	LNDYGFYTTTGIGYQP	161	0.72
14	SVLYNSTFFSTFKCYG	36	0.71
15	GEVFNATKFPSVYAWE	9	0.7
16	RVVVLSFELLNAPATV	178	0.65

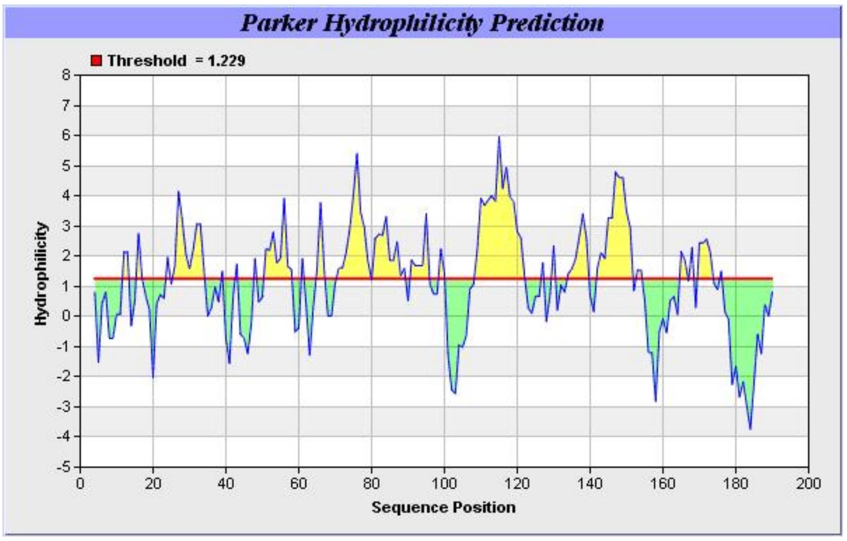
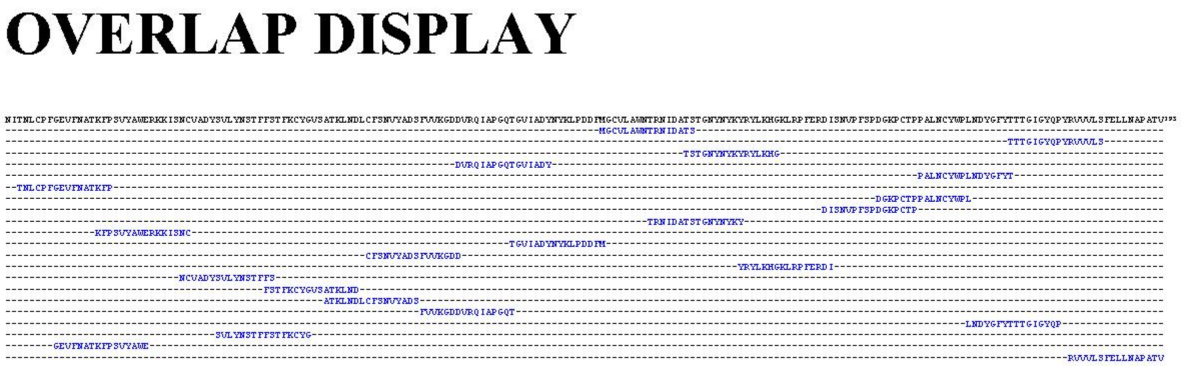


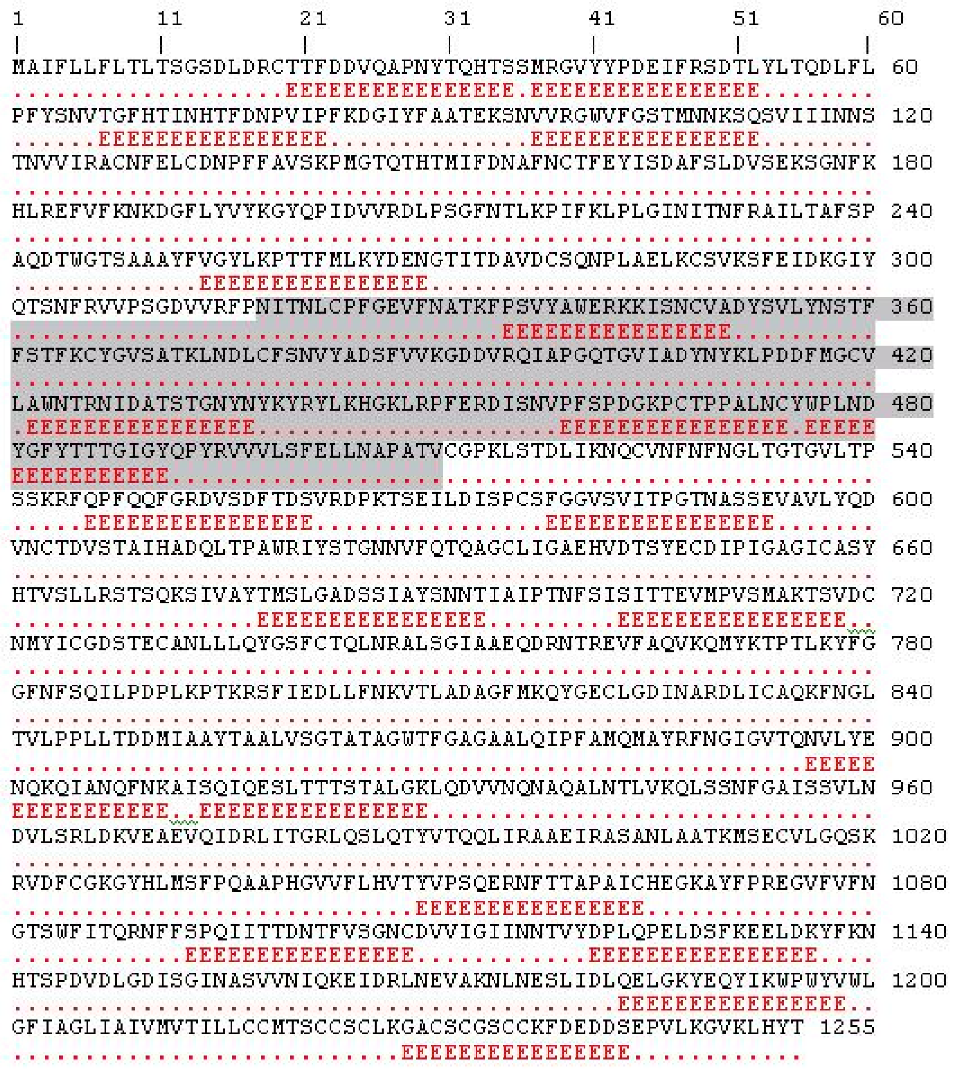
Figure A.1 Analysis of RBD of SARS-CoV S protein using Parker's hydrophilic scale.



OVERLAP DISPLAY

```
NITNLCPFGEVFNATKFPSVYAWERKKISNCVADYSVLYNSTFFSTFKCYGVSATKLNDLCFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRNIDATSTGNYNYKYRYLKHGKLRPFERDISNVPFSPDGKPCTPPALNCYWPLNDYGFYTTTGIGYQPYRVVVLSFELLNAPATV193
---------------------------------------------------------------------------------------------------MGCVLAWNTRNIDATS------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------TTTGIGYQPYRVVVLS----------
-----------------------------------------------------------------------------------------------------------------TSTGNYNYKYRYLKHG----------------------------------------------------------------
---------------------------------------------------------------------------DVRQIAPGQTGVIADY------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------PALNCYWPLNDYGFYT-------------------------
--TNLCPFGEVFNATKFP-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------DGKPCTPPALNCYWPL--------------------------------
----------------------------------------------------------------------------------------------------------------------------------------DISNVPFSPDGKPCTP-----------------------------------------
-----------------------------------------------------------------------------------------------------------TRNIDATSTGNYNYKY----------------------------------------------------------------------
---------------KFPSVYAWERKKISNC------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------TGVIADYNYKLPDDFM---------------------------------------------------------------------------------------------
------------------------------------------------------------CFSNVYADSFVVKGDD---------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------YRYLKHGKLRPFERDI-------------------------------------------------------
-----------------------------NCVADYSVLYNSTFFS----------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------FSTFKCYGVSATKLND--------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------ATKLNDLCFSNVYADS----------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------FVVKGDDVRQIAPGQT------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------LNDYGFYTTTGIGYQP-----------------
-----------------------------------SVLYNSTFFSTFKCYG----------------------------------------------------------------------------------------------------------------------------------------------
--------GEVFNATKFPSVYAWE---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------RVVVLSFELLNAPATV
```

Figure A.2 ABCPred predictions on RBD region of SARS-CoV S protein.



```
1         11        21        31        41        51         60
|         |         |         |         |         |         |
MAIFLLFLTLTSGSDLDRCTTFDDVQAPNYTQHTSSMRGVYYPDEIFRSDTLYLTQDLFL 60
...................EEEEEEEEEEEEEEEE.EEEEEEEEEEEEEEEE........
PFYSNVTGFHTINHTFDNPVIPFKDGIYFAATEKSNVVRGWVFGSTMNNKSQSVIIINNS 120
......EEEEEEEEEEEEEEEE..............EEEEEEEEEEEEEEEE........
TNVVIRACNFELCDNPFFAVSKPMGTQTHTMIFDNAFNCTFEYISDAFSLDVSEKSGNFK 180
............................................................
HLREFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNTLKPIFKLPLGINITNFRAILTAFSP 240
............................................................
AQDTWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVDCSQNPLAELKCSVKSFEIDKGIY 300
.............EEEEEEEEEEEEEEEE...............................
QTSNFRVVPSGDVVRFPNITNLCPFGEVFNATKFPSVYAWERKKISNCVADYSVLYNSTF 360
..................................EEEEEEEEEEEEEEEE..........
FSTFKCYGVSATKLNDLCFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCV 420
............................................................
LAWNTRNIDATSTGNYNYKYRYLKHGKLRPFERDISNVPFSPDGKPCTPPALNCYWPLND 480
.EEEEEEEEEEEEEEEE.....................EEEEEEEEEEEEEEEE.EEEEE
YGFYTTTGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVNFNFNGLTGTGVLTP 540
EEEEEEEEEEE.................................................
SSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPCSFGGVSVITPGTNASSEVAVLYQD 600
.....EEEEEEEEEEEEEEEE................EEEEEEEEEEEEEEEE.......
VNCTDVSTAIHADQLTPAWRIYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASY 660
............................................................
HTVSLLRSTSQKSIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDC 720
.................EEEEEEEEEEEEEEEE.........EEEEEEEEEEEEEEEE..
NMYICGDSTECANLLLQYGSFCTQLNRALSGIAAEQDRNTREVFAQVKQMYKTPTLKYFG 780
............................................................
GFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGECLGDINARDLICAQKFNGL 840
............................................................
TVLPPLLTDDMIAAYTAALVSGTATAGWTFGAGAALQIPFAMQMAYRFNGIGVTQNVLYE 900
.......................................................EEEEE
NQKQIANQFNKAISQIQESLTTTSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLN 960
EEEEEEEEEEE..EEEEEEEEEEEEEEEE...............................
DVLSRLDKVEAEVQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSK 1020
............................................................
RVDFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFPREGVFVFN 1080
............................EEEEEEEEEEEEEEEE................
GTSWFITQRNFFSPQIITTDNTFVSGNCDVVIGIINNTVYDPLQPELDSFKEELDKYFKN 1140
............EEEEEEEEEEEEEEEE............EEEEEEEEEEEEEEEE....
HTSPDVDLGDISGINASVVNIQKEIDRLNEVAKNLNESLIDLQELGKYEQYIKWPWYVWL 1200
..........................................EEEEEEEEEEEEEEEE..
GFIAGLIAIVMVTILLCCMTSCCSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
...........................EEEEEEEEEEEEEEEE............
```

Figure A.3 BCPred predictions over entire SARS CoV S protein. “E” indicates that the corresponding amino acid residue lies in a predicted linear B-cell epitope. Shaded region represents the RBD region.

BIBLIOGRAPHY

Alix, A. (1999). Predictive estimation of protein linear epitopes by using the program PEOPLE. *Vaccine*, 18:311–4.

Andrews, S., Tsochantaridis, I., and Hofmann, T. (2003). Support vector machines for multiple-instance learning. In *Adv. Neural. Inf. Process. Syst. 15*, pages 561–568. Cambridge, MA:MIT Press.

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

Bairoch, A. and Apweiler, R. (2000). The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Res.*, 28:45–48.

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., and Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16:412–424.

Barlow, D., Edwards, M., Thornton, J., et al. (1986). Continuous and discontinuous protein antigenic determinants. *Nature*, 322:747–748.

Beniac, D., Andonov, A., Grudeski, E., and Booth, T. (2006). Architecture of the SARS coronavirus prefusion spike. *Nat. Struct. Mol. Biol.*, 13:751–752.

Bennett, K. and Mangasarian, O. (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optim. Methods Softw.*, 1:23–34.

Bhasin, M. and Raghava, G. (2004). SVM based method for predicting HLA-DRB1 0401 binding peptides in an antigen sequence. *Bioinformatics*, 20:3.

Bhasin, M., Singh, H., and Raghava, G. (2003). MHCBN: a comprehensive database of MHC binding and non-binding peptides. *Bioinformatics*, 19:665–666.

Björklund, Å., Soeria-Atmadja, D., Zorzet, A., Hammerling, U., and Gustafsson, M. (2005). Supervised identification of allergen-representative peptides for in silico detection of potentially allergenic proteins. *Bioinformatics*, 21:39–50.

Blythe, M. and Flower, D. (2005). Benchmarking B cell epitope prediction: underperformance of existing methods. *Protein Sci.*, 14:246–248.

Brefeld, U. and Scheffer, T. (2005). AUC maximizing support vector learning. *Preceedings of ICML 2005 workshop on ROC Analysis in Machine Learning.*

Breiman, L. (1996). Bagging predictors. *Mach. Learn.*, 24:123–140.

Brusic, V., Rudy, G., Harrison, L., and Journals, O. (1998). MHCPEP, a database of MHC-binding peptides: update 1997. *Nucleic Acids Res.*, 26:368–371.

Bui, H., Sidney, J., Peters, B., Sathiamurthy, M., Sinichi, A., Purton, K., Mothé, B., Chisari, F., Watkins, D., and Sette, A. (2005). Automated generation and evaluation of specific MHC binding predictive tools: ARB matrix applications. *Immunogenetics*, 57:304–314.

Bulashevska, A. and Eils, R. (2006). Predicting protein subcellular locations using hierarchical ensemble of Bayesian classifiers based on Markov chains. *BMC Bioinformatics*, 7:298.

Burden, F. and Winkler, D. (2006). Predictive Bayesian neural network models of MHC class II peptide binding. *J. Mol. Graph. Model.*, 2005:481–9.

Buus, S., Lauemoller, S., Worning, P., Kesmir, C., Frimurer, T., Corbet, S., Fomsgaard, A., Hilden, J., Holm, A., and Brunak, S. (2003). Sensitive quantitative predictions of peptide-MHC binding by a'Query by Committee' artificial neural network approach. *Tissue Antigens*, 62:378–384.

Cai, C., Han, L., Ji, Z., Chen, X., and Chen, Y. (2003). SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Res.*, 31:3692–3697.

Caragea, C., Sinapov, J., Silvescu, A., Dobbs, D., and Honavar, V. (2007). Glycosylation site prediction using ensembles of support vector machine classifiers. *BMC Bioinformatics*, 8:438.

Chang, C.-C. and Lin, C.-J. (2001). *LIBSVM: a library for support vector machines.* Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chang, S., Ghosh, D., Kirschner, D., and Linderman, J. (2006). Peptide length-based prediction of peptide-MHC class II binding. *Bioinformatics*, 22:2761.

Chen, J., Liu, H., Yang, J., and Chou, K. (2007). Prediction of linear B-cell epitopes using amino acid pair antigenicity scale. *Amino Acids*, 33:423–428.

Chen, Y., Bi, J., and Wang, J. (2006). MILES: multiple-instance learning via embedded instance selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28:1931–1947.

Chinnasamy, A., Sung, W., and Mittal, A. (2004). Protein structure and fold prediction using tree-augmented naive Bayesian classifier. *Pac. Symp. Biocomput.*, 387:98.

Clark, A., Florencio, C., Watkins, C., and Serayet, M. (2006). Planar languages and learnability. *International Colloquium on Grammatical Inference (ICGI06)*, pages 148–160.

Cui, J., Han, L., Lin, H., Tan, Z., Jiang, L., Cao, Z., and Chen, Y. (2006a). MHC-BPS: MHC-binder prediction server for identifying peptides of flexible lengths from sequence-derived physicochemical properties. *Immunogenetics*, 58:607–613.

Cui, J., Han, L., Lin, H., Tang, Z., Jiang, L., Cao, Z., and Chen, Y. (2006b). MHC-BPS: MHC-binder prediction server for identifying peptides of flexible lengths from sequence-derived physicochemical properties. *Immunogenetics*, 58:607–13.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

Dietterich, T. G., Lathrop, R. H., and Lozano-Perez, T. (1997). Solving the multiple-instance problem with axis parallel rectangles. *Artif. Intell.*, 89:31–71.

Dimitrov, D. (2003). The secret life of ACE2 as a receptor for the SARS virus. *Cell*, 115:652–653.

Dobson, P. and Doig, A. (2003). Distinguishing enzyme structures from non-enzymes without alignments. *J. Mol. Biol.*, 330:771–783.

Donnes, P. and Kohlbacher, O. (2006). SVMHC: a server for prediction of MHC-binding peptides. *Nucleic Acids Res.*, 34:W194.

Doytchinova, I. and Flower, D. (2003). Towards the in silico identification of class II restricted T-cell epitopes: a partial least squares iterative self-consistent algorithm for affinity prediction. *Bioinformatics*, 19:2263–2270.

Drosten, C., Gunther, S., Preiser, W., van der Werf, S., Brodt, H., Becker, S., Rabenau, H., Panning, M., Kolesnikova, L., Fouchier, R., et al. (2003). Identification of a novel coronavirus in patients with severe acute respiratory syndrome. *N. Engl. J. Med.*, 348:1967.

Eisenhaber, F., Frommel, C., and Argos, P. (1996). Prediction of secondary structural content of proteins from their amino acid composition alone. II. The paradox with secondary structural class. *Proteins*, 25:169–79.

EL-Manzalawy, Y., Dobbs, D., and Honavar, V. (2008a). On Evaluating MHC-II Binding Peptide Prediction Methods. *PLoS ONE*, 3.

EL-Manzalawy, Y., Dobbs, D., and Honavar, V. (2008b). Predicting flexible length linear B-cell epitopes. *7th International Conference on Computational Systems Bioinformatics*, pages 121–131.

EL-Manzalawy, Y., Dobbs, D., and Honavar, V. (2008c). Predicting linear B-cell epitopes using evolutionary information. *IEEE International Conference on Bioinformatics and Biomedicine.*

EL-Manzalawy, Y., Dobbs, D., and Honavar, V. (2008d). Predicting linear B-cell epitopes using string kernels. *J. Mol. Recognit.*, 21:243–255.

Emini, E., Hughes, J., Perlow, D., and Boger, J. (1985). Induction of hepatitis A virus-neutralizing antibody by a virus-specific synthetic peptide. *J. Virol.*, 55:836–839.

Fisher, R. (1973). *Statistical methods and scientific inference.* Hafner Press, New York.

Flower, D. (2007). *Immunoinformatics: predicting immunogenicity in silico.* Quantum distributor, 1st edition.

Fonseca, C. and Fleming, P. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. *Proceedings of the Fifth International Conference on Genetic Algorithms*, 423:416–423.

Fouchier, R., Kuiken, T., Schutten, M., van Amerongen, G., van Doornum, G., van den Hoogen, B., Peiris, M., Lim, W., Stöhr, K., and Osterhaus, A. (2003). Koch's postulates fulfilled for SARS virus. *Nature*, 423:240.

Frank, E., Wang, Y., Inglis, S., Holmes, G., and Witten, I. (1998). Using model trees for classification. *Mach. Learn.*, 32:63–76.

Freund, Y. and Mason, L. (1999). The alternating decision tree learning algorithm. In *Proceedings of the Sixteenth International Conference on Machine Learning table of contents*, pages 124–133. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.

Freund, Y. and Schapire, R. (1996). Experiments with a new boosting algorithm. In *Proceedings of 13th International Conference in Machine Learning*, pages 148–156.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Stat.*, 11:86–92.

Fung, G., Dundar, M., Krishnapuram, B., and Rao, R. (2007). Multiple instance learning for computer aided diagnosis. In *Adv. Neural Inf. Process. Syst. 19*, pages 425–432. MIT Press.

Garcia, J., Fierro, R., Puentes, A., Cortes, J., Bermudez, A., Cifuentes, G., Vanegas, M., and Patarroyo, M. (2007). Monosaccharides modulate HCV E2 protein-derived peptide biological properties. *Biochem. Biophys. Res. Commun.*, 355:409–418.

Gartner, T., Flach, P., Kowalczyk, A., and Smola, A. (2002). Multi-instance kernels. *Proceedings of the 19th International Conference on Machine Learning*, pages 179–186.

Gärtner, T., Flach, P., and Wrobrl, S. (2003). On graph kernels: Hardness results and efficient alternatives. *Lect. Notes Comput. Sci.*, 2777:129–143.

Goldman, S. and Scott, S. (2003). Multiple-instance learning of real-valued geometric patterns. *Ann. Math. Artif. Intell.*, 39:259–290.

Gowthaman, U. and Agrewala, J. (2008). In silico tools for predicting peptides binding to HLA-class II molecules: more confusion than conclusion. *J. Proteome Res.*, 7:154–63.

Greenbaum, J., Andersen, P., Blythe, M., Bui, H., Cachau, R., Crowe, J., Davies, M., Kolaskar, A., Lund, O., Morrison, S., et al. (2007). Towards a consensus on datasets and evaluation metrics for developing B-cell epitope prediction tools. *J. Mol. Recognit.*, 20:75–82.

Haste Andersen, P., Nielsen, M., and Lund, O. (2006). Prediction of residues in discontinuous B-cell epitopes using protein 3D structures. *Protein Sci.*, 15:2558.

Hattotuwagama, C., Guan, P., Doytchinova, I., Zygouri, C., and Flower, D. (2004). Quantitative online prediction of peptide binding to the major histocompatibility complex. *J. Mol. Graph. Model.*, 22:195–207.

Haussler, D. (1999). Convolution kernels on discrete structures. *UC Santa Cruz Technical Report UCS-CRL-99-10.*

Henikoff, J. and Henikoff, S. (1996). Using substitution probabilities to improve position-specific scoring matrices. *Bioinformatics*, 12:135–143.

Henikoff, S. and Henikoff, J. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 89:10915–10919.

Herschtal, A. and Raskutti, B. (2004). Optimising area under the ROC curve using gradient descent. In *Proceedings of the twenty-first international conference on Machine learning*, pages 49–56. ACM Press.

Hertz, T. and Yanover, C. (2006). PepDist: a new framework for protein-peptide binding prediction based on learning peptide distance functions. *BMC Bioinformatics*, 7:S1–S3.

Hua, S. and Sun, Z. (2001). Support vector machine approach for protein subcellular localization prediction. *Bioinformatics*, 17:721–728.

Janeway, C., Travers, P., et al. (2004). *Immunobiology: the immune system in health and disease*. Garland Pub, 6th edition.

Karplus, P. and Schulz, G. (1985). Prediction of chain flexibility in proteins: a tool for the selection of peptide antigen. *Naturwiss.*, 72:21–213.

Korber, B., LaBute, M., and Yusim, K. (2006). Immunoinformatics comes of age. *PLoS Comput. Biol.*, 2:e71.

Ksiazek, T., Erdman, D., Goldsmith, C., Zaki, S., Peret, T., Emery, S., Tong, S., Urbani, C., Comer, J., Lim, W., et al. (2003). A novel coronavirus associated with severe acute respiratory syndrome. *N. Engl. J. Med.*, 348:1953.

Kulkarni-Kale, U., Bhosle, S., and Kolaskar, A. (2005). CEP: a conformational epitope prediction server. *Nucleic Acids Res.*, 33:W168.

Langeveld, J., martinez Torrecuadrada, J., boshuizen, R., Meloen, R., and Ignacio, C. (2001). Characterisation of a protective linear B cell epitope against feline parvoviruses. *Vaccine*, 19:2352–2360.

Larsen, J., Lund, O., and Nielsen, M. (2006). Improved method for predicting linear B-cell epitopes. *Immunome Res.*, 2:2.

Lawrence, C., Altschul, S., Boguski, M., Liu, J., Neuwald, A., and Wootton, J. (1993). Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 262:208–214.

Le Cessie, S. and Van Houwelingrn, J. (1992). Ridge estimators in logistic regression. *Appl. Stat.*, 41:191–201.

Leslie, C., Eskin, E., Cohen, A., Weston, J., and Noble, W. (2004). Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20:467–476.

Leslie, C., Eskin, E., and Noble, W. (2002). The spectrum kernel: a string kernel for SVM protein classification. *Proceedings of the Pacific Symposium on Biocomputing*, 7:566–575.

Li, W., Jaroszewski, L., and Godzik, A. (2002). Tolerating some redundancy significantly speeds up clustering of large protein databases. *Bioinformatics*, 18:77–82.

Li, W., Moore, M., Vasilieva, N., Sui, J., Wong, S., Berne, M., Somasundaran, M., Sullivan, J., Luzuriaga, K., Greenough, T., et al. (2003). Angiotensin-converting enzyme 2 is a functional receptor for the SARS coronavirus. *Nature*, 426:450–454.

Lien, S., Shih, Y., Chen, H., Tsai, J., Leng, C., Lin, M., Lin, L., Liu, H., Chou, A., Chang, Y., et al. (2007). Identification of synthetic vaccine candidates against SARS CoV infection. *Biochem. Biophys. Res. Commun.*, 358:716–721.

Lin, H., Ray, S., Tongchusak, S., Reinherz, E., and Brusic, V. (2008). Evaluation of MHC class I peptide binding prediction servers: applications for vaccine research. *BMC Immunol.*, 9:8.

Liu, W., Meng, X., Xu, Q., Flower, D., and Li, T. (2006). Quantitative prediction of mouse class I MHC peptide binding affinity using support vector machine regression (SVR) models. *BMC Bioinformatics*, 7:182.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C. (2002). Text classification using string kernels. *J. Mach. Learn. Res.*, 2:419–444.

Luo, R., Feng, Z., and Liu, J. (2002). Prediction of protein structural class by amino acid and polypeptide composition. *FEBS J.*, 269:4219–4225.

MacKay, D. (1998). Introduction to Gaussian processes. *Neural Netw. Mach. Learn.*, 168:133–165.

Madden, D. (1995). The three-dimensional structure of peptide-MHC complexes. *Annu. Rev. Immunol.*, 13:587–622.

Mallios, R. (2003). A consensus strategy for combining HLA-DR binding algorithms. *Hum. Immunol.*, 64:852–6.

Mamitsuka, H. (1998). Predicting peptides that bind to MHC molecules using supervised learning of Hidden Markov Models. *PROTEINS: Structure, Function, and Genetics*, 33:460–474.

Mangasarian, O. and Wild, E. (2005). Multiple instance classification via successive linear programming. *Data Mining Institute Technical Report 05-02.*

Mann, H. and Whitney, D. (1947). On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Stat.*, 18:50–60.

Maron, O. and Lozano-Perez, T. (1998). A framework for multiple-instance learning. *Adv. Neural. Inf. Process. Syst.*, 10:570–576.

Maron, O. and Ratan, A. (1998). Multiple-instance learning for natural scene classification. *Proceedings of the Fifteenth International Conference on Machine Learning table of contents*, pages 341–349.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (2004). Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087.

Mitchell, T. (1997). Machine Learning. *McGraw Hill*.

Murugan, N. and Dai, Y. (2005). Prediction of MHC class II binding peptides based on an iterative learning model. *Immunome Res.*, 1:6.

Nadeau, C. and Bengio, Y. (2003). Inference for the generalization error. *J. Mach. Learn. Res.*, 52:239–281.

Neter, J., Wasserman, W., Kutner, M., et al. (1985). *Applied linear statistical models*. Irwin Homewood, IL.

Nielsen, M., Lundegaard, C., and Lund, O. (2007). Prediction of MHC class II binding affinity using SMM-align, a novel stabilization matrix alignment method. *BMC Bioinformatics*, 8:238.

Nielsen, M., Lundegaard, C., Worning, P., Lauemøller, S., Lamberth, K., Buus, S., Brunak, S., and Lund, O. (2003). Reliable prediction of T-cell epitopes using neural networks with novel sequence representations. *Protein Sci.*, 12:1007–1017.

Nielsen, M., Lundegaard, C., Worning, P., Sylvester-Hvid, C., Lamberth, K., Buus, S., Brunak, S., and Lund, O. (2004). Improved prediction of MHC class I and II epitopes using a novel Gibbs sampling approach. *Bioinformatics*, 20:1388–97.

Noguchi, H., Kato, R., Hanai, T., Matsubara, Y., Honda, H., Brusic, V., and Kobayashi, T. (2002). Hidden Markov model-based prediction of antigenic peptides that interact with MHC class II molecules. *J. Biosci. Bioeng.*, 94:264–270.

Odorico, M. and Pellequer, J. (2003). BEPITOPE: predicting the location of continuous epitopes and patterns in proteins. *J. Mol. Recognit.*, 16:20–22.

O'Sullivan, D., Sidney, J., Del Guercio, M., Colon, S., and Sette, A. (1991). Truncation analysis of several DR binding epitopes. *J. Immunol.*, 146:1240–6.

Parker, J. and Guo, D and, H. R. (1986). New hydrophilicity scale derived from high-performance liquid chromatography peptide retention data: correlation of predicted surface residues with antigenicity and x-ray-derived accessible sites. *Biochemistry*, 25:5425–5432.

Parker, K., Bednarek, M., and Coligan, J. (1994). Scheme for ranking potential HLA-A2 binding peptides based on independent binding of individual peptide side-chains. *J. Immunol.*, 152:163–75.

Peiris, J., Lai, S., Poon, L., Guan, Y., Yam, L., Lim, W., Nicholls, J., Yee, W., Yan, W., Cheung, M., et al. (2003). Coronavirus as a possible cause of severe acute respiratory syndrome. *The Lancet*, 361:1319–1325.

Pellequer, J. and Westhof, E. (1993). PREDITOP: a program for antigenicity prediction. *J. Mol. Graph.*, 11:204–210.

Pellequer, J., Westhof, E., and Van Regenmortel, M. (1991). Predicting location of continuous epitopes in proteins from their primary structures. *Meth. Enzymol.*, 203:176–201.

Pellequer, J., Westhof, E., and Van Regenmortel, M. (1993). Correlation between the location of antigenic sites and the prediction of turns in proteins. *Immunol. Lett.*, 36:83–99.

Peters, B., Bui, H., Frankild, S., Nielsen, M., Lundegaard, C., and Rzhetsky, A. (2006). A community resource benchmarking predictions of peptide binding to MHC-I molecules. *PLoS Comput. Biol.*, 2:e65.

Peters, B. and Sette, A. (2005). Generating quantitative models describing the sequence specificity of biological processes with the stabilized matrix method. *BMC Bioinformatics*, 6:132.

Peters, B., Sidney, J., Bourne, P., Bui, H., Buus, S., Doh, G., Fleri, W., Kronenberg, M., Kubo, R., Lund, O., et al. (2005). The immune epitope database and analysis resource: from vision to blueprint. *PLoS Biology*, 3.

Pier, G., Lyczak, J., and Wetzler, L. (2004). *Immunology, infection, and immunity.* ASM Press, 1st edition.

Platt, J. (1998). *Fast training of support vector machines using sequential minimal optimization.* MIT Press.

Prabakaran, P., Gan, J., Feng, Y., Zhu, Z., Choudhry, V., Xiao, X., Ji, X., and Dimitrov, D. (2006). Structure of severe acute respiratory syndrome coronavirus receptor-binding domain complexed with neutralizing antibody. *J. Biol. Chem.*, 281:15829.

Quinlan, J. (1992). Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348.

Raghava, G. (2004). *MHCBench: evaluation of MHC binding peptide prediction algorithms.* datasets available at http://www.imtech.res.in/raghava/mhcbench/.

Rajapakse, M., Schmidt, B., Feng, L., and Brusic, V. (2007). Predicting peptides binding to MHC class II molecules using multi-objective evolutionary algorithms. *BMC Bioinformatics*, 8:459.

Rammensee, H., Bachmann, J., Emmerich, N., Bachor, O., and Stevanović, S. (1999). SYFPEITHI: database for MHC ligands and peptide motifs. *Immunogenetics*, 50:213–219.

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

Ramon, J. and De Raedt, L. (2000). Multi instance neural networks. *Proceedings of the ICML-2000 Workshop on Attribute-Value and Relational Learning.*

Rangwala, H., DeRonne, K., Karypis, G., and of Computer Science, M. U. M. D. (2006). *Protein structure prediction using string kernels.* Defense Technical Information Center.

Ray, S. and Craven, M. (2005). Supervised versus multiple instance learning: An empirical comparison. In *Proceedings of the Twentieth-Second International Conference on Machine Learning*, pages 697–704.

Ray, S. and Page, D. (2001). Multiple instance regression. *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 425–432.

Reche, P., Glutting, J., Zhang, H., and Reinherz, E. (2004). Enhancement to the RANKPEP resource for the prediction of peptide binding to MHC molecules using profiles. *Immunogenetics*, 56:405–419.

Saha, S., Bhasin, M., and Raghava, G. (2005). Bcipep: a database of B-cell epitopes. *BMC Genomics*, 6:79.

Saha, S. and Raghava, G. (2004). BcePred: Prediction of continuous B-cell epitopes in antigenic sequences using physico-chemical properties. *Artificial Immune Systems, Third International Conference (ICARIS 2004), LNCS*, 3239:197–204.

Saha, S. and Raghava, G. (2006a). ABCPred benchmarking datasets. available at http://www.imtech.res.in/raghava/abcpred/dataset.html.

Saha, S. and Raghava, G. (2006b). Prediction of continuous B-cell epitopes in an antigen using recurrent neural network. *Proteins*, 65:40–48.

Saigo, H., Vert, J., Ueda, N., and Akutsu, T. (2004). Protein homology detection using string alignment kernels. *Bioinformatics*, 20:1682–1689.

Sainz, J. B., Rausch, J., Gallaher, W., Garry, R., and Wimley, W. (2005). Identification and characterization of the putative fusion peptide of the severe acute respiratory syndrome-associated coronavirus spike protein. *Virology*, 79:7195–7206.

Salomon, J. and Flower, D. (2006). Predicting class II MHC-peptide binding: a kernel based approach using similarity scores. *BMC Bioinformatics*, 7:501.

Scott, S., Zhang, J., and Brown, J. (2005). On generalized multiple-instance learning. *Int. J. Comput. Intell. Appl.*, 5:21–35.

Seewald, A. and Kleedorfer, F. (2005). Lambda pruning: an approximation of the string subsequence kernel. Technical report, Technical Report, Osterreichisches Forschungsinstitut fur Artificial Intelligence, Wien, TR-2005-13.

Shevade, S., Keerthi, S., Bhattacharyya, C., and Murthy, K. (2000). Improvements to the SMO algorithm for SVM regression. *IEEE Trans. Neural. Netw.*, 11:1189.

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

Söllner, J., Grohmann, R., Rapberger, R., Perco, P., Lukas, A., Mayer, B., and Blythe, M. (2008). Analysis and prediction of protective continuous B-cell epitopes on pathogen proteins. *Immunome Res.*, 2008:1–17.

Söllner, J. and Mayer, B. (2006). Machine learning approaches for prediction of linear B-cell epitopes on proteins. *J. Mol. Recognit.*, 19:200–208.

Sturniolo, T., Bono, E., Ding, J., Raddrizzani, L., Tuereci, O., Sahin, U., Braxenthaler, M., Gallazzi, F., Protti, M., Sinigaglia, F., et al. (1999). Generation of tissue-specific and promiscuous HLA ligand databases using DNA microarrays and virtual HLA class II matrices. *Nat. Biotechnol.*, 17:555–561.

Sui, J., Li, W., Murakami, A., Tamin, A., Matthews, L., Wong, S., Moore, M., Tallarico, A., Olurinde, M., Choe, H., et al. (2004). Potent neutralization of severe acute respiratory syndrome (SARS) coronavirus by a human mAb to S1 protein that blocks receptor association. *Proc. Natl. Acad. Sci.*, 101:2536–2541.

Swets, J. (1988). Measuring the accuracy of diagnostic systems. *Science*, 240:1285–1293.

Tao, Q., Scott, S., Vinodchandran, N., Osugi, T., and Mueller, B. (2008). Kernels for generalized multiple-instance learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30:2084–2098.

Terribilini, M., Lee, J., Yan, C., Jernigan, R., Honavar, V., and Dobbs, D. (2006). Prediction of RNA binding sites in proteins from amino acid sequence. *RNA*, 12:1450–1462.

Trost, B., Bickis, M., and Kusalik, A. (2007). Strength in numbers: achieving greater accuracy in MHC-I binding prediction by combining the results from multiple prediction tools. *Immunome Res.*, 3:5.

Tsoumakas, G. and Katakis, I. (2007). Multi-label classification: an overview. *Int. J. Data Warehousing and Min.*, 3:1–13.

Urdan, T. (2005). *Statistics in plain english.* Lawrence Erlbaum Associates.

Vapnik, V. (2000). *The nature of statistical learning theory.* Springer, 2nd edition.

Walter, G. (1986). Production and use of antibodies against synthetic peptides. *J. Immunol. Methods*, 88:149–61.

Wang, J. and Zucker, J. D. (2000). Solving the multiple-instance problem: a lazy learning approach. In *Proceedings 17th International Conference on Machine Learning*, pages 1119–1125.

Wang, P., Sidney, J., Dow, C., Mothé, B., Sette, A., and Peters, B. (2008). A systematic assessment of MHC class II peptide binding predictions and evaluation of a consensus approach. *PLoS Comput. Biol.*, 4.

Wang, Y. and Witten, I. (1997). Induction of model trees for predicting continuous classes. In *Proceeding of European Conference on Machine Learning.*

Weidmann, N., Frank, E., and Pfahringer, B. (2003). A two-level learning method for generalized multi-instance problems. In *Proceedings of the European Conference on Machine Learning*, pages 468–479. Springer.

Witten, I. and Frank, E. (2005). *Data mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2nd edition.

Wu, F., Olson, B., Dobbs, D., and Honavar, V. (2006). Comparing kernels for predicting protein binding sites from amino acid sequence. *International Joint Conference on Neural Networks (IJCNN06)*, pages 1612–1616.

Wu, X., Shang, B., Yang, R., Yu, H., Hai, Z., Shen, X., Ji, Y., Lin, Y., Di Wu, Y., Lin, G., et al. (2004). The spike protein of severe acute respiratory syndrome (SARS) is cleaved in virus infected Vero-E6 cells. *Cell Res.*, 14:400–406.

Xue, Y., Ren, J., Gao, X., Jin, C., Wen, L., and Yao, X. (2008). GPS 2.0: Prediction of kinase-specific phosphorylation sites in hierarchy. *Mol. Cell Proteomics*, page M700574.

Yan, L., Dodier, R., Mozer, M., and Wolniewicz, R. (2003). Optimizing classifier performance via the Wilcoxon-Mann-Whitney statistics. *Proceedings of the International Conference on Machine Learning*, pages 848–855.

Yang, Z. (2007). Predicting palmitoylation sites using a regularised bio-basis function neural network. *Lect. Notes Comput. Sci.*, 4463:406.

Yu, C., Chen, Y., Lu, C., and Hwang, J. (2006a). Prediction of protein subcellular localization. *Proteins*, 64:643–651.

Yu, H., Zhu, X., and Huang, M. (2006b). Using string kernel to predict binding peptides for MHC class II molecules. *The 8th International Conference on Signal Processing*, 4.

Yu, K., Petrovsky, N., Schonbach, C., Koh, J., and Brusic, V. (2002). Methods for prediction of peptide binding to MHC molecules: a comparative study. *Mol. Med.*, 8:137–48.

Zaki, N., Deris, S., and Illias, R. (2005). Application of string kernels in protein sequence classification. *Appl. Bioinformatics*, 4:45–52.

Zhang, G., Bozic, I., Kwoh, C., August, J., and Brusic, V. (2007). Prediction of supertype-specific HLA class I binding peptides using support vector machines. *J. Immunol. Methods*, 320:143–54.

Zhang, Q., Goldman, S., Yu, W., and Fritts, J. (2002). Content-based image retrieval using multiple-instance learning. *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 682–689.

Zhang, Q. and Goldman, S. A. (2001). Em-dd: An improved multiple-instance learning technique. *Neural. Inf. Process. Syst.*, 14.

Zhou, Z., Jiang, K., and Li, M. (2005). Multi-instance learning based web mining. *Appl. Intell.*, 22:135–147.

Zhou, Z. and Zhang, M. (2007). Solving multi-instance problems with classifier ensemble based on constructive clustering. *Knowl. Inf. Syst.*, 11:155–170.

Zhu, J., Kosset, S., Hastie, T., and Tibshirani, R. (2004). 1-norm support vector machines. *Adv. Neural Inf. Process. Syst. 16.*