Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 06-005

Protein Structure Prediction using String Kernels

Huzefa Rangwala, Kevin Deronne, and George Karypis

March 03, 2006

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
03 MAR 2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Protein Structure Prediction using String Kernels	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Minnesota,Department of Computer Science and Engineering,200 Union Street SE,Minneapolis,MN,55455-0159**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**20**	

Protein Structure Prediction using String Kernels

Huzefa Rangwala, Kevin DeRonne, George Karypis

Department of Computer Science & Engineering/Digital Technology Center
University of Minnesota, Minneapolis, MN 55455

rangwala@cs.umn.edu, deronne@cs.umn.edu, karypis@cs.umn.edu

Abstract

With recent advances in large scale sequencing technologies, we have seen an exponential growth in protein sequence information. Currently, our ability to produce sequence information far out-paces the rate at which we can produce structural and functional information. Consequently, researchers increasingly rely on computational techniques to extract useful information from known structures contained in large databases, though such approaches remain incomplete. As such, unraveling the relationship between pure sequence information and three dimensional structure remains one of the great fundamental problems in molecular biology.

In this report we aim to show several ways in which researchers try to characterize the structural, functional and evolutionary nature of proteins. Specifically, we focus on three common prediction problems, secondary structure prediction, remote homology and fold prediction. We describe a class of methods employing large margin classifiers with novel kernel functions for solving these problems, supplemented with a thorough evaluation study.

1 Introduction

The motivation behind the structural determination of proteins is based on the belief that structural information will ultimately result in a better understanding of intricate biological processes. Many methods exist to predict protein structure at different levels of granularity. Due to the interest from a wide range of research communities in this subject matter, a biennial competition, The Critical Assessment for Structure Prediction (CASP) [1] assesses the performance of current structure prediction methods. In this report we aim to show several ways in which researchers try to characterize the structural, functional and evolutionary nature of proteins.

Within each structural entity called a protein there lies a set of recurring substructures, and within these substructures are smaller substructures still. As an example, consider hemoglobin, the oxygen-carrying molecule in human blood. Hemoglobin has four domains that come together to form its quaternary structure. Each domain assembles (i.e. folds) itself independently to form a tertiary structure. These tertiary structures are comprised of multiple secondary structure elements–in hemoglobin's case α helices. Alpha helices (and their counterpart β sheets) have elegant repeating patterns dependent upon sequences of amino acids. These sequences form the primary structure of a protein, the smallest structural division aside from atoms. Hence, the linear ordering of amino acids forms secondary structure, arranging secondary structures yields tertiary structure, and the arrangement of tertiary structures forms quaternary structure. (See Figure 1). Research in computational structure prediction concerns itself mainly with predicting secondary and tertiary structure from known experimentally determined primary structure. This is due to the relative ease of determining primary structure and the complexity involved in quaternary structure. In this chapter we provide an overview of current secondary structure prediction techniques, followed by a breakdown of the tertiary structure prediction problem and descriptions of algorithms for each of several more restricted problems.

1.1 Secondary Structure Prediction

A sequence of characters representing the secondary structure of a protein describes the general three-dimensional form of local regions. These regions organize themselves into patterns of repeatedly occurring structural fragments independently from the rest of the protein. The most dominant local conformations of polypeptide chains are alpha helices and beta sheets. These local structures have a certain regularity in their form, attributed to the hydrogen bond interactions between various residues. An alpha helix has a coil-like structure, whereas a beta sheet consists of parallel strands of residues. (See Figure 1). In addition to regular secondary structure elements, irregular shapes form an important part of the structure and function of proteins. These elements are typically termed coil regions.

[1] http://predictioncenter.org/

Secondary structure can be divided into several types, though usually at least three classes (alpha-helix, coils and beta-sheet) are used. No unique method of assigning residues to a particular secondary structure state from atomic coordinates exists, though the most widely accepted protocol is based on the DSSP algorithm [25]. DSSP uses the following structural classes: H (α-helix), G (3_{10}-helix), I (π-helix), E (β-strand), B (isolated β-bridge), T (turn), S (bend), and – (other). Several other secondary structure assignment algorithms use a reduction scheme that converts this eight-state assignment down to three states by assigning H and G to the helix state (H), E and B to a the strand state (E), and the rest (I, T, S, and –) to a coil state (C). This is the format generally used in structure databases. Within the secondary structure prediction problem, the task is to learn a model that assigns a secondary structure state to each residue of an input sequence in the absence of atomic coordinates.

1.2 Protein Tertiary Structure

One of the biggest goals in structural bioinformatics is the prediction of the three-dimensional (3D) structure of a protein from its one-dimensional (1D) protein sequence. The goal is to be able to determine the shape (known as a fold) that a given amino acid sequence will adopt. The problem is further divided based on whether the sequence will adopt a new fold or bear resemblance to an existing fold (template) in some protein structure database. Fold recognition is easy when the sequence in question has a high degree of sequence similarity to a sequence with known structure [7]. If the two sequences share evolutionary ancestry they are said to be homologous. For such sequence pairs we can build the structure for the query protein by choosing the structure of the known homologous sequence as template. This is known as comparative modelling.

In the case where no good template structure exists for the query, one must attempt to build the protein tertiary structure from scratch. These methods are usually called *ab initio* methods. In a third fold prediction scenario, there may not necessarily be a good sequence similarity with a known structure, but a structural template may still exist for the given sequence. To clarify this case, if one were aware of the target structure then they could extract the template using structure-structure alignments of the target against the entire structural database. It is important to note that the target and template need not be homologous. These two cases define the fold prediction (homologous) and fold prediction (analogous) problems during the CASP competition.

1.2.1 Comparative Modeling

Comparative Modeling or homology modeling is used when there exists a clear relationship between the sequence of a query protein (unknown structure) to that of a sequence of a known structure. The most basic approach to structure prediction for such (query) proteins is to perform a pairwise sequence alignment against each sequence in protein sequence databases. This can be accomplished using sequence alignment algorithms such as Smith-Waterman [55] or sequence search algorithms (e.g. BLAST [3]). With a good sequence alignment in hand, the challenge in comparative modeling becomes how to best build a three-dimensional protein structure for a query protein using the template structure.

The heart of the above process is the selection of a suitable structural template based on sequence pair similarity. This is followed by the alignment of query sequence to the template structure selected to build the backbone of the query protein. Finally the entire modeled structure is refined by loop construction and side-chain modeling. Several comparative modeling methods, more commonly known as modeler programs, have been developed over the past several years [6, 13] focussing on various parts of the problem.

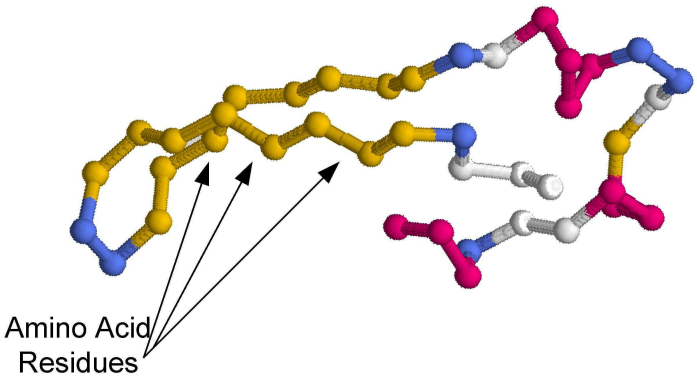
1.2.2 Fold Prediction (Homologous)

While satisfactory methods exist to detect homologs (proteins that share similar evolutionary ancestry) with high levels of similarity, accurately detecting homologs at low levels of sequence similarity (remote homology detection) remains a challenging problem. Some of the most popular approaches for remote homology prediction compare a protein with a collection of related proteins using methods such as PSI-BLAST [2], protein family profiles [15], hidden Markov models (HMMs) [30, 5], and SAM [26]. These schemes produce models that are generative, in the sense that they build a model for a set of related proteins and then check to see how well this model explains a candidate protein.

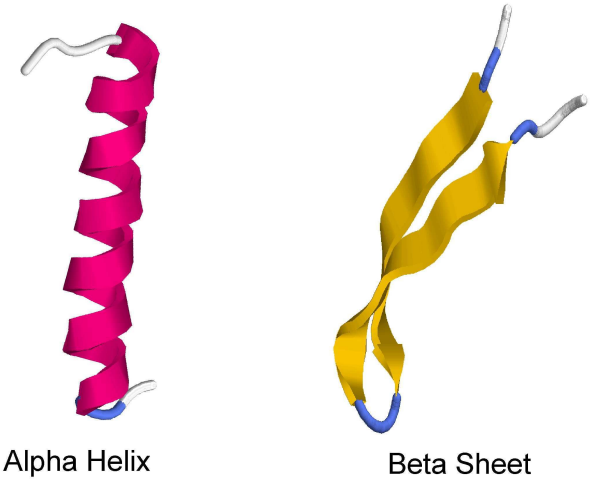
In recent years, the performance of remote homology detection has been further improved through the use of methods that explicitly model the differences between the various protein families (classes) by building discriminative models. In particular, a number of different methods have been developed that use support vector machines (SVM) [56] to produce results that are generally superior to those produced by either pairwise sequence comparisons or approaches based on generative models– provided there is sufficient training data. [19, 35, 33, 34, 17, 18, 52, 31].

1.2.3 Fold Prediction (Analogous)

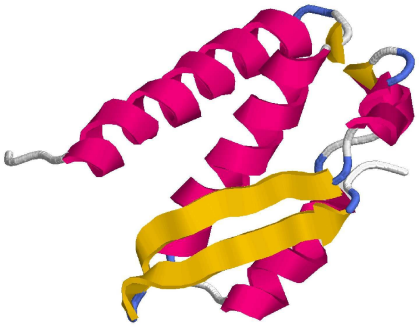
Occasionally a query sequence will have a native fold similar to another known fold in a database, but the two sequences will have no detectable similarity. In many cases the two proteins will lack an evolutionary relationship as well. As the definition of this problem relies on the inability of current methods to detect sequential similarity, the set of proteins falling into this category remains in flux. As new methods continue to improve at finding sequential similarities as a result of increasing database size and better techniques, the number of proteins in question decreases. Techniques to find structures for such query sequences revolve around mounting the query sequence on a series of template structures, in a process known as threading [21, 20, 8]. An objective energy function provides a score for each alignment, and the highest-scoring template is chosen. Obviously, if the correct template does not exist in the series then the method will not produce an accurate



Primary Protein Structure
- sequence of amino acid residues



Secondary Protein Structure
- local structures



Tertiary Protein Structure
- three dimensional atomic co-ordinates

Quaternary Protein Structure
- interaction of several protein chains

prediction. As a result of this limitation, predicting the structure of proteins in this category usually falls to new fold prediction techniques.

1.2.4 New Fold Techniques to predict novel protein structure have come a long way in recent years, though a definitive solution to the problem remains elusive. Research in this area can be roughly divided into fragment assembly [24, 28, 32] and first-principle based approaches, though occasionally the two are combined [9]. The former attempt to assign a fragment with known structure to a section of the unknown query sequence. The latter start with an unfolded conformation, usually surrounded by solvent, and allow simulated physical forces to fold the protein as would normally happen *in vivo*. Usually, algorithms from either class will use reduced representations of query proteins during initial stages to reduce the overall complexity of the problem.

2 Learning from Data

Supervised learning is the task of creating a function that maps a set of inputs to a particular set of outputs by examining labelled training data. This form of learning plays a vital role in several bioinformatic applications including protein structure prediction.

Several books [11, 56, 10] cover the foundations of supervised learning in detail. The general framework of a supervised learning problem is as follows. Given an input domain $\mathcal{X}$ and output domain $\mathcal{Y}$, learn a function mapping each element of $\mathcal{X}$ to an element in domain $\mathcal{Y}$. In formal terms, given some training data $(X_1, Y_1) \ldots (X_n, Y_n)$, we need to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ mapping each object $X_i \in \mathcal{X}$ to a classification label $Y_i \in \mathcal{Y}$.

It is assumed that there exists an underlying probability distribution $D(X, Y)$ over $\mathcal{X} \times \mathcal{Y}$. This distribution remains unchanged for the training and test samples, but this distribution is unknown. The training and test samples are assumed to be drawn independently, identically distributed from $D(X, Y)$.

Classifiers can be categorized as parametric models and distribution free models. Parametric models attempt to solve the supervised learning problem by explicitly modeling the joint distribution $D(X, Y)$ or conditional distribution $D(Y|X)$ for all X. Bayesian and Hidden Markov Models are examples of parametric models. Distribution-free models make no attempt to learn the distribution, but rather choose a function in a selected hypothesis space for classification purposes. Margin based learners like support vector machines are distribution-free classifiers.

2.1 Kernel Methods

Given a set of positive training examples $\mathcal{S}^+$ and a set of negative training examples $\mathcal{S}^-$, a support vector machine (SVM) learns a classification function $f(X)$ of the form

$$f(X) = \sum_{X_i \in \mathcal{S}^+} \lambda_i^+ \mathcal{K}(X, X_i) - \sum_{X_i \in \mathcal{S}^-} \lambda_i^- \mathcal{K}(X, X_i), \tag{1}$$

where λ_i^+ and λ_i^- are non-negative weights that are computed during training by maximizing a quadratic objective function, and $\mathcal{K}(.,.)$ is called the *kernel* function, which is computed over all training-set and test-set instances. Given this function, a new instance X is predicted to be positive or negative depending on whether $f(X)$ is positive or negative. In addition, the value of $f(X)$ can be used to obtain a meaningful ranking of a set of instances, as it represents the strength by which they are members of the positive or negative class.

The kernel function, when computed over all pairs of training instances, produces a symmetric matrix. To ensure the validity of a kernel, it is necessary to ensure that it satisfies Mercer's conditions, which require the pairwise matrix generated by the kernel function to be positive semidefinite. Formally, any function can be used as a kernel so long as for any number n, and any possible set of distinct instances $\{X_1, \ldots, X_n\}$, the $n \times n$ Gram matrix defined by $K_{i,j} = \mathcal{K}(X_i, X_j)$ is symmetric positive semidefinite.

A symmetric function defined on the training set instances can be converted into a positive definite by adding to the diagonal of the training Gram matrix a sufficiently large non-negative constant [52]. For example, the constant shift embedding kernelizing approach proposes the use of smallest negative eigenvalue to be subtracted from the main diagonal [58].

3 Structure Prediction - Capturing the right signals

Thus far we have looked at several problems within the larger context of protein structure prediction. An ideal solution to the structure prediction problem would correctly predict, from only sequence information, the complete native conformation of a protein in three-dimensional space. Due to the difficulty of developing such a grand solution, decomposing the problem has led to good solutions to smaller parts of the problem.

In the remainder of this chapter we focus on three common prediction problems, secondary structure prediction, remote homology and fold prediction. We also describe a class of methods employing large margin classifiers with novel kernel

functions for solving these problems.

One of the fundamental steps in building good classification models is selecting features that fit the classification task well. The input domain X for the protein structure prediction problems is the amino acid residues and their properties.

A protein sequence X of length n is represented by a sequence of characters $X = \langle a_1, a_2, \ldots, a_n \rangle$ such that each character corresponds to one of the 20 standard amino acids. Quite often, the learning and prediction algorithms segment the sequence into short contiguous segments called $w$mers. Specifically, given a sequence X of length n and a user-supplied parameter w, the $w\text{m}er$ at position i of X ($w < i \leq n - w$) is defined to be the $(2w + 1)$-length subsequence of X centered at position i. That is, the $w\text{m}er$ contains a_i, the w amino acids before, and the w amino acids after a_i. We will denote this subsequence as $w\text{m}er_X(i)$.

It is widely believed that a sequence of amino acids encodes a structural signal [4], and this belief forms the underlying premise of the protein structure prediction problem. Working under this assumption, researchers have tried to encapsulate protein sequence information in various forms for structure analysis. One common way to incorporate more information about the structure of a sequence is to consider similar (and hopefully, therefore, related) sequences. Using multiple sequence alignments one can infer structural information about conserved regions. Many classifiers take as input profiles constructed from such alignments.

The profile of a sequence X of length n can be represented by two $n \times 20$ matrices. The first is its position-specific scoring matrix PSSM_X that is computed directly by PSI-BLAST using the scheme described in [2]. The rows of this matrix correspond to the various positions in X and the columns correspond to the 20 distinct amino acids. The second matrix is its position-specific *frequency* matrix PSFM_X that contains the frequencies used by PSI-BLAST to derive PSSM_X. These frequencies (also referred to as *target frequencies* [38]) contain both the sequence-weighted observed frequencies (also referred to as *effective frequencies* [38]) as well as the BLOSUM62 [16] derived-pseudocounts [2].

We use the notations defined above to illustrate the machine learning methods used for secondary structure prediction, remote homology detection and fold recognition.

4 Secondary Structure Prediction

A large number of secondary structure prediction algorithms have been developed, and since their inception prediction accuracy has been continuously improved. Many algorithms can currently achieve a sustained three-state prediction accuracy in the range of 77%–78%, and combinations of them can sometimes further improve the accuracy by one to two percentage points. These improvements have been well-documented [51], and are attributed to an ever-expanding set of experimentally determined tertiary structures, the use of evolutionary information, and to algorithmic advances.

The secondary structure prediction approaches in use today can be broadly categorized into three groups: neighbor-based, model-based, and meta-predictor-based. The neighbor-based approaches [53, 14, 23] predict the secondary structure by identifying a set of similar sequence-fragments with known secondary structure; the model-based approaches [49, 22, 44, 42], employ sophisticated machine learning techniques to learn a predictive model trained on sequences of known structure; whereas the meta-predictor-based approaches [12, 41] predict based on a combination of the results of various different neighbor and/or model-based techniques. The near real-time evaluation of many of these methods performed by the EVA server [48] shows that the model-based approaches tend to produce statistically better results than the neighbor-based schemes, which are further improved by some of the more recently developed meta-predictor-based approaches [41].

Historically, the most successful model-based approaches such as PHD [49], PSIPRED [22], and SSPro [42], were based on neural network (NN) learning techniques. However, in recent years, a number of researchers have also developed secondary structure prediction algorithms based on support vector machines.

In the remainder of this section we present one such SVM-based secondary structure prediction algorithm called YASSPP that shows exemplary performance [29].

4.1 YASSPP Overview

The overall structure of YASSPP is similar to that used by many existing secondary structure prediction algorithms like PHD and PSIPRED. The approach is illustrated in Figure 2 It consists of two models, referred to as L_1 and L_2, that are connected together in a cascaded fashion. The L_1 model assigns to each position a weight for each of the three secondary structure elements $\{C, E, H\}$, which are provided as input to the L_2 model to predict the actual secondary structure class of each position. The L_1 model treats each position of the sequence as an independent prediction problem, and the purpose of the L_2 model is to determine the structure of a position by taking into account the predicted structure of adjacent positions. YASSPP splits the training set equally between the L_1 and L_2 models.

Both the L_1 and L_2 models consist of three binary SVM classifiers ($\{M_1^{C/\bar{C}}, M_1^{E/\bar{E}}, M_1^{H/\bar{H}}\}$ and $\{M_2^{C/\bar{C}}, M_2^{E/\bar{E}}, M_2^{H/\bar{H}}\}$, respectively) trained to predict whether or not a position belongs to a particular secondary structure state or not (i.e., one-vs-rest models). The output values of the L_1 model are the raw functional outputs of these binary classifiers (i.e., $M_1^{C/\bar{C}}$, $M_1^{E/\bar{E}}$, and

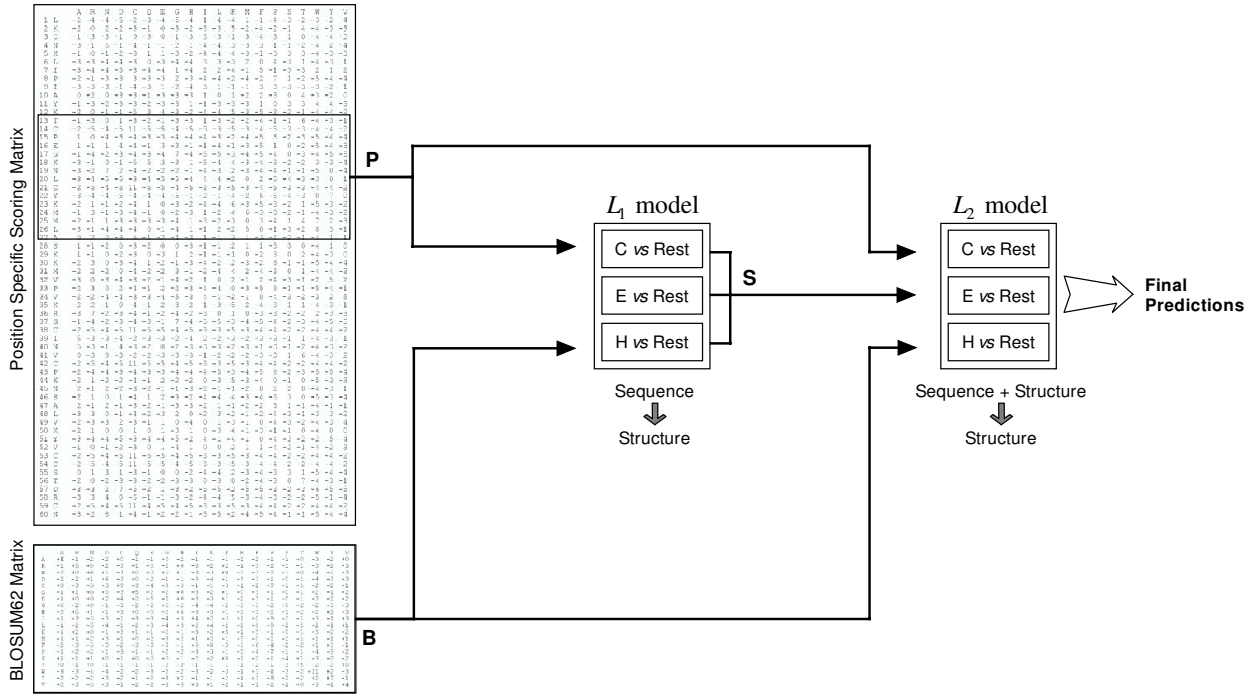


Figure 2: The general architecture of YASSPP's learning framework

$M_1^{H/\bar{H}}$), whereas the predicted secondary state of the L_2 model corresponds to the state whose corresponding binary classifier achieves the maximum value. That is,

$$\text{Predicted state} = \underset{x \in \{C,E,H\}}{\text{argmax}} \ (M_2^{x/\bar{x}}). \tag{2}$$

During training, for each position i that belongs to one of the three secondary structure states (i.e., classes) of a sequence X, the input to the SVM is a $(2w+1)$-length subsequence $w\text{mer}$ of X. The proper value for the parameter w is determined experimentally. During secondary structure prediction, a similar approach is used to construct a $w\text{mer}$ around each position i of a *query* sequence X with unknown secondary structure.

4.2 Input Sequence Coding

For the input sequence coding there are two different approaches for the L_1 model and two different approaches for the L_2 model. L_1's first coding scheme represents each $w\text{mer}$ x as a $(2w+1) \times 20$ matrix P_x, whose rows are obtained directly from the rows of the PSSM for each position. The second coding scheme augments this PSSM-based representation by adding another $(2w+1) \times 20$ matrix B_x, whose rows are the rows of the BLOSUM62 matrix corresponding to each position's amino acid. These schemes are referred as the P and the PB coding schemes, respectively.

By augmenting the $w\text{mer}$ coding scheme to contain both PSSM- as well as BLOSUM62-based information, the SVM can learn a model that is also partially based on the non-position specific information. This information will remain valid even in cases in which PSI-BLAST could not or failed to generate correct alignments.

The two coding schemes for the L_2 model are derived from the corresponding coding schemes of L_1 by including the predictions computed by L_1's three binary classifiers. This is done by adding another $(2w+1) \times 3$ matrix S_x,whose columns store the raw functional predictions of the $M_1^{C/\bar{C}}$, $M_1^{E/\bar{E}}$, and $M_1^{H/\bar{H}}$ models, respectively. Thus, the first coding scheme consists of matrices P_x and S_x, and the second coding scheme consists of matricesP_x, B_x, and S_x. These coding schemes are novel compared to the existing methods.

4.3 Profile-Based Kernel Functions

YASSPP shows a methodology for designing and evaluation various kernel functions for use by binary SVM classifiers of the L_1 and L_2 models. It develops kernel functions that are derived by combining a normalized second-order kernel, in which the contribution of each position decreases based on how far away it is from the central residue, along with an exponential function.

The general structure of the kernel functions used in YASSPP is given by

$$\mathcal{K}(x,y) = \exp\left(1.0 + \frac{\mathcal{K}_1(x,y)}{\sqrt{\mathcal{K}_1(x,x)\,\mathcal{K}_1(y,y)}}\right), \tag{3}$$

where x and y are two $w$mers, $\mathcal{K}_1(x,y)$ is given by

$$\underset{1}{\mathcal{K}}(x,y) = \overset{cs}{\underset{2}{\mathcal{K}}}(x,y) + (\overset{cs}{\underset{2}{\mathcal{K}}}(x,y))^2, \tag{4}$$

and $\mathcal{K}_2^{cs}(x,y)$ is a kernel function that depends on the choice of the particular input coding scheme cs, and for each one of the P, PB, PS, and PBS coding schemes is defined as follows:

$$\overset{P}{\underset{2}{\mathcal{K}}}(x,y) = \sum_{j=-w}^{j=w} \frac{P_x(j,:)P_y^t(j,:)}{1+|j|}, \tag{5}$$

$$\overset{PB}{\underset{2}{\mathcal{K}}}(x,y) = \overset{P}{\underset{2}{\mathcal{K}}}(x,y) + \sum_{j=-w}^{j=w} \frac{B_x(j,:)B_y^t(j,:)}{1+|j|}, \tag{6}$$

$$\overset{PS}{\underset{2}{\mathcal{K}}}(x,y) = \overset{P}{\underset{2}{\mathcal{K}}}(x,y) + \gamma \sum_{j=-w}^{j=w} \frac{S_x(j,:)S_y^t(j,:)}{1+|j|}, \tag{7}$$

$$\overset{PBS}{\underset{2}{\mathcal{K}}}(x,y) = \overset{PB}{\underset{2}{\mathcal{K}}}(x,y) + \gamma \sum_{j=-w}^{j=w} \frac{S_x(j,:)S_y^t(j,:)}{1+|j|}. \tag{8}$$

The various terms involving the rows of the P, B, and S matrices (e.g., $P_x(j,:)P_y^t(j,:)$) correspond to the dot-products of the rows corresponding to the jth positions of the $w$mers (indexed from $-w$ to $+w$). We do not delve into the various characteristics that are coded in the constructions of the kernel functions but direct the reader to the report [29] for further details.

4.4 Performance Evaluation

For assessing the performance of YASSPP a wide variety of datasets were used. A thorough parameter study was done to study the impact of the various coding schemes, kernel choices and the best parameters. We show some of the comparative performance study results for YASSPP.

The prediction accuracy is assessed using four widely used performance measures. These are the three-state per-residue accuracy (Q_3), the segment overlap measure (SOV), the per-state Matthews correlation coefficients (C_C, C_E, C_H), and the information index (Info). Q_3 is a measure of the overall three-state prediction accuracy and is defined as the percentage of residues whose structural class is predicted correctly [49]. The SOV is a segment-level measure of the overall prediction accuracy. This measure is initially introduced in [50] and subsequently refined in [54]. Matthews correlation coefficients [37] provide a per-state measure of prediction performance and for a particular state $i \in \{C, E, H\}$ it is given by

$$C_i = \frac{p_i n_i - u_i o_i}{\sqrt{(p+i+u_i)(p_i+o_i)(n_i+u_i)(n_i+o_i)}}, \tag{9}$$

where p_i is the number of correctly predicted residues in state i, n_i is the number of residues that were correctly rejected (true negatives), u_i is the number of residues that were incorrectly rejected (false negatives), and o_i is the number of residues that were incorrectly predicted to be in state i (false positives). Finally, the information index [49] is an entropy-related measure that merges the observed and the predicted state-specific accuracy measures into a single number with all these elements contributing equally.

Table 1 compares the performance achieved by YASSPP against that achieved by PHDpsi [44], PSIPRED [22], SAM-T99sec [27], PROFsec [47], SCRATCH [42], SSPro4 [42], and SABLE2 [43]. These schemes represent some of the best performing schemes currently evaluated by the EVA server, and their results were obtained directly from EVA. Since EVA did not use all the methods to predict all the sequences of EVAc4, Table 1 presents four different sets of results for YASSPP$_{P+PS}$ and YASSPP$_{PB+PBS}$ (indicated by the superscripts 1–4), each obtained by averaging the various performance assessment methods over the common subset. These common subsets contained 165, 134, 86, and 115 sequences, respectively.

These results show that both YASSPP$_{P+PS}$ and YASSPP$_{PB+PBS}$ achieve better prediction performance than that achieved by any of the other schemes across all the different performance assessment measures. In particular, for the entire dataset, YASSPP$_{PB+PBS}$ achieves a Q_3 score of 79.34%, which is 1.7 percentage points higher than the second best-performing scheme in terms of Q_3 (SAM-T99sec), and an SOV score of 78.65%, which is 2.6 percentage points higher than the second

Table 1: Performance on the EVAc4 dataset.

Scheme	Q_3	SOV	Info	C_C	C_E	C_H
PHDpsi	74.52	70.69	0.346	0.529	0.685	0.665
PSIPRED	77.62	76.05	0.375	0.561	0.735	0.696
SAM-T99sec	77.64	75.05	0.385	0.578	0.721	0.675
PROFsec	76.54	75.39	0.378	0.562	0.714	0.677
[1]YASSPP$_{P+PS}$	78.35	77.20	0.407	0.589	0.746	0.708
ErrSig	0.86	1.21	0.015	0.015	0.021	0.017
[1]YASSPP$_{PB+PBS}$	79.34	78.65	0.419	0.608	0.747	0.722
ErrSig	0.82	1.16	0.015	0.015	0.021	0.016
SCRATCH	75.75	71.38	0.357	0.545	0.690	0.659
[2]YASSPP$_{P+PS}$	78.39	77.69	0.406	0.586	0.750	0.711
ErrSig	0.97	1.36	0.016	0.017	0.023	0.018
[2]YASSPP$_{PB+PBS}$	79.31	78.75	0.416	0.602	0.751	0.722
ErrSig	0.94	1.29	0.016	0.017	0.023	0.018
SSPro4	77.96	72.73	0.385	0.559	0.711	0.696
[3]YASSPP$_{P+PS}$	79.21	78.60	0.418	0.590	0.749	0.723
ErrSig	1.19	1.67	0.021	0.023	0.030	0.022
[3]YASSPP$_{PB+PBS}$	80.03	79.00	0.430	0.605	0.751	0.736
ErrSig	1.18	1.68	0.022	0.024	0.030	0.022
SABLE2	76.85	73.55	0.376	0.546	0.725	0.682
[4]YASSPP$_{P+PS}$	78.70	78.09	0.417	0.596	0.766	0.715
ErrSig	1.00	1.42	0.018	0.018	0.025	0.019
[4]YASSPP$_{PB+PBS}$	79.85	79.71	0.432	0.615	0.768	0.730
ErrSig	0.97	1.39	0.018	0.019	0.025	0.019

YASSPP$_{P+PS}$ uses the $P+PS$ input coding and the YASSPP$_{PB+PBS}$ uses the $PB+PBS$ input coding and were obtained using $w=7$ (i.e., $w$mers of size 15). The [1]YASSPP are the averages over the set of sequences in common with PHDpsi, PSIPRED, SAM-T99sec, and PROFsec. The [2]YASSPP are the averages over the set of sequences in common with SCRATCH. The [3]YASSPP are the averages over the set of sequences in common with SSPro4. The [4]YASSPP are the averages over the set of sequences in common with SABLE2.

Table 2: Comparative performance of YASSPP against other secondary structure prediction servers.

RS126 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
PSIPRED	81.01	76.24	0.45	0.65	0.70	0.77
PHD	76.92	72.57	0.38	0.57	0.63	0.73
Prof	76.95	71.70	0.38	0.58	0.63	0.73
SSPro	77.01	70.24	0.38	0.58	0.61	0.72
YASSPP$_{P+PS}$	79.81	74.41	0.42	0.61	0.70	0.76
ErrSig	0.80	1.28	0.02	0.02	0.02	0.02
YASSPP$_{PB+PBS}$	80.29	75.65	0.43	0.61	0.70	0.75
ErrSig	0.79	1.25	0.02	0.02	0.02	0.02
CB513 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
PSIPRED	79.95	76.48	0.43	0.63	0.68	0.76
PHD	77.61	74.98	0.39	0.59	0.65	0.73
Prof	77.13	73.74	0.39	0.58	0.64	0.73
SSPro	79.07	74.39	0.42	0.61	0.65	0.76
YASSPP$_{P+PS}$	80.52	77.39	0.45	0.62	0.70	0.74
ErrSig	0.40	0.60	0.01	0.01	0.01	0.01
YASSPP$_{PB+PBS}$	80.99	77.86	0.45	0.63	0.70	0.75
ErrSig	0.39	0.60	0.01	0.01	0.01	0.01

YASSPP$_{P+PS}$ uses the $P + PS$ input coding and the YASSPP$_{PB+PBS}$ uses the $PB + PBS$ input coding. Both schemes use $w$mers of length 15 ($w = 7$). The results for PSIPRED, PHD, Prof, and SSPro were obtained from [46].
ErrSig is the significant difference margin for each score (to distinguish between two methods) and is defined as the standard deviation divided by the square root of the number of proteins ($\sigma/\sqrt{N}$).

best performing scheme in terms of SOV (PSIPRED).

Table 2 compares the performance achieved by YASSPP's production server with that achieved by other model-based servers such as PSIPRED, PHD, Prof, and SSPro [46]. These results show that the performance achieved by YASSPP$_{P+PS}$ and YASSPP$_{PB+PBS}$ is in general higher than that achieved by the other servers. YASSPP$_{PB+PBS}$'s performance is one to four percentage points higher in terms of Q_3 and SOV. The only exception is the RS126 dataset for which PSIPRED achieves somewhat better prediction performance than either YASSPP$_{P+PS}$ or YASSPP$_{PB+PBS}$ (PSIPRED achieves a Q_3 score of 81.01 vs 80.29 for YASSPP$_{PB+PBS}$). However, as measured by *ErrSig*, this performance difference is not statistically significant. Also, as was the case with the previous results, YASSPP$_{PB+PBS}$ achieves better prediction performance than that achieved by YASSPP$_{P+PS}$.

5 Remote Homology and Fold Prediction

Both remote homology detection and fold recognition are central problems in computational biology and bioinformatics, with the aim of classifying protein sequences into structural and functional groups or classes.

Pairwise sequence comparison methods (e.g., sequence alignment algorithms like Smith-Waterman [55] and sequence database search tools like BLAST [1]) are able to detect homologous sequences with a high percentage sequence identity. However, as the percent identity between sequence pairs decreases, the problem of finding the correct homologous pairs becomes increasingly difficult.

Some of the better performing schemes in this domain use profile information to compare a query protein with a collection of related proteins. Profiles for a sequence can be defined in terms of a multiple sequence alignment of a query sequence

with its statistically significant homologs (as computed by PSI-BLAST [2]) or in the form of hidden markov model (HMM) states [30, 5]. The models built in this fashion are examples of generative models.

The current state-of-the-art methods employ discriminative based modelling techniques and have a large advantage over generative models in this domain. Support vector machines have been the popular choice of discriminative learners.

One of the early attempts at using a feature-space-based approach is the SVM-Fisher method [19], in which a profile HMM model is estimated on a set of proteins belonging to the positive class. This HMM is then used to extract a vector representation for each protein. Another approach is the SVM-pairwise scheme [35], which represents each sequence as a vector of pairwise similarities between all sequences in the training set. A relatively simpler feature space that contains all possible short subsequences ranging from 3–8 amino acids ($k$mers) is explored in a series of papers (Spectrum kernel [33], Mismatch kernel [34], and Profile kernel [31]). All three of these methods represent a sequence X as a vector in this simpler feature space, but differ in the scheme they employ to actually determine if a particular dimension u (i.e., $k$mer) has a non-zero weight in X's vector or not. The Spectrum kernel considers u to be present if X contains u as a substring, the Mismatch kernel considers u to be present if X contains a substring that differs with u in at most a predefined number of positions (i.e., mismatches), whereas the Profile kernel considers u to be present if X contains a substring whose PSSM–based ungapped alignment score with u is above a user-supplied threshold. An entirely different feature space is explored by the SVM-Isites [17] and SVM-HMMSTR [18] methods that take advantage of a set of local structural motifs (SVM-Isites) and their relationships (SVM-HMMSTR).

An alternative to measuring pairwise similarity through a dot-product of vector representations is to calculate an explicit protein similarity measure. The recently developed LA-Kernel method [52] represents one such example of a *direct kernel function*. This scheme measures the similarity between a pair of protein sequences by taking into account all the optimal gapped local alignment scores between all possible subsequences of the pair. The experiments presented in [52] show that this kernel is superior to previously developed schemes that do not take into account sequence profiles and that the overall classification performance improves by taking into account all possible local alignments.

5.1 Profile-Based Kernel Functions

Recently, a set of direct profile-based kernel functions were developed and tested to show very good performance [45]. The first class, referred to as window-based, determines the similarity between a pair of sequences by combining ungapped alignment scores of fixed-length subsequences. The second, referred to as local alignment-based, determines the similarity between a pair of sequences using Smith-Waterman alignments and a position independent affine gap model, optimized for the characteristics of the scoring system. Both kernel classes utilize profiles constructed automatically via PSI-BLAST and employ a profile-to-profile scoring scheme that extend a recently introduced profile alignment method [38].

One way of computing the profile-to-profile scores would be to take the dot product between the profile columns for the two positions, shown in Equation 10

$$S_{X,Y}(i,j) = \sum_{k=1}^{20} \text{PSSM}_X(i,k) * \text{PSSM}_Y(j,k), \tag{10}$$

Another example of such a scoring function [45] is given by Equation 11. This particular scoring function captures the similarity between the two profile positions using both the position specific scoring matrices and position specific frequency matrices. This scoring function can be defined as,

$$\begin{aligned} S_{X,Y}(i,j) \quad = \quad & \sum_{k=1}^{20} \text{PSFM}_X(i,k)\, \text{PSSM}_Y(j,k) + \\ & \sum_{k=1}^{20} \text{PSFM}_Y(j,k)\, \text{PSSM}_X(i,k), \end{aligned} \tag{11}$$

5.1.1 Smith-Waterman based Kernel Functions As explained in section 2.1, the choice of kernel function plays a critical role in the performance of a classifier. A simple Smith-Waterman based alignment scoring scheme can be used as a kernel function provided steps are followed to ensure its validity–specifically, that it follows Mercer's conditions.

The Smith-Waterman based kernel computes the similarity between a pair of sequences X and Y by finding an optimal alignment between them that optimizes a particular scoring function. Given two sequences X and Y of lengths n and m, respectively, the SW-PSSM kernel computes their similarity as the score of the optimal local alignment. In this alignment, the similarity between two sequence positions is determined using the profile-to-profile scoring scheme of Equation 11, and a position independent affine gap model.

Within this local alignment framework, the similarity score between a pair of sequences depends on the gap-opening (*go*)

and gap-extension (*ge*) costs, and the intrinsic characteristics of the profile-to-profile scoring scheme. A scoring system whose average score is positive will tend to produce very long alignments, potentially covering segments of low biologically relevant similarity. On the other hand, if the scoring system cannot easily produce alignments with positive scores, then it may fail to identify any non-empty similar subsequences. In order to obtain meaningful local alignments, the scoring scheme that is used should produce alignments whose score must on average be negative with the maximum score being positive [55].

To ensure that the SW-PSSM kernel can correctly account for the characteristics of the scoring system, the profile-to-profile scores calculated from Equation 11 are modified by adding a constant value. This scheme, commonly referred to as *zero-shifting* [57], ensures that the resulting alignments have scores that are negative on the average, while allowing for positive maximum scores.

5.1.2 Window-based Kernel Functions The local alignment based kernels capture the similarity between sequence pairs by combining the ungapped alignment scores of $w\text{mer}$ subsequences between the various positions of the sequences. Based on the combination of fixed and varied length $w$mers for different pair positions between sequences, [45] introduces three novel window-based kernel functions.

The ungapped alignment score between two $w$mers is computed using the profile-to-profile scoring method of Equation 11 as follows:

$$wscore_{X,Y}(i,j) = \sum_{k=-w}^{w} S_{X,Y}(i+k, j+k). \tag{12}$$

The All Fixed-width $w$mers (AF-PSSM) kernel computes the similarity between a pair of sequences X and Y by adding-up the alignment scores of all possible $w$mers between X and Y that have a positive ungapped alignment score. Specifically, if the ungapped alignment score between two $w$mers at positions i and j of X and Y, respectively is denoted by $wscore_{X,Y}(i,j)$, n and m are the lengths of X and Y, respectively, and $\mathcal{P}_w$ is the set of all possible $w\text{mer}$-pairs of X and Y with a positive ungapped alignment score, i.e,

$$\mathcal{P}_w = \{(w\text{mer}_X(i), w\text{mer}_Y(j)) \mid wscore_{X,Y}(i,j) > 0\}, \tag{13}$$

for $w+1 \leq i \leq n-w$ and $w+1 \leq j \leq m-w$, then the AF-PSSM kernel computes the similarity between X and Y as

$$\text{AF-PSSM}_{X,Y}(w) = \sum_{(w\text{mer}_X(i), w\text{mer}_Y(j)) \in \mathcal{P}_w} wscore_{X,Y}(i,j). \tag{14}$$

The Best Fixed-width $w$mer (BF-PSSM) kernel improves on the AF-PSSM kernel by selecting a subset $\mathcal{P}'_w$ of $\mathcal{P}_w$ (as defined in Equation 13) such that (i) each position of X and each position of Y is present in at most one $w\text{mer}$-pair and (ii) the sum of the $w$scores of the selected pairs is maximized. Given $\mathcal{P}'_w$, the similarity between the pair of sequences is then computed as follows:

$$\text{BF-PSSM}_{X,Y}(w) = \sum_{(w\text{mer}(X,i), w\text{mer}(Y,j)) \in \mathcal{P}'_w} wscore_{X,Y}(i,j). \tag{15}$$

The relation between $\mathcal{P}'_w$ and $\mathcal{P}_w$ can be better understood if the possible $w\text{mer}$-pairs in $\mathcal{P}_w$ are viewed as forming an $n \times m$ matrix, whose rows correspond to the positions of X, columns to the positions of Y, and values correspond to their respective $w$scores. Within this context, $\mathcal{P}'_w$ corresponds to a matching of the rows and columns [40] whose weight is high (bipartite graph matching problem). Since the selection forms a matching, each position of X (or Y) contributes a single $w\text{mer}$ in Equation 15, and as such, eliminates the multiplicity present in the AF-PSSM kernel. At the same time, the BF-PSSM kernel attempts to select the *best* $w$mers for each position.

In fixed-width $w\text{mer}$-based kernels the width of the $w$mers is fixed for all pairs of sequences and throughout the entire sequence. As a result, if w is set to a relatively high value, it may fail to identify positive scoring subsequences whose length is smaller than $2w+1$, whereas if it is set too low, it may fail to reward sequence-pairs that have relatively long similar subsequences.

The Best Variable-width $w$mer (BV-PSSM) kernel overcomes this problem by using variable length $w$mers. It is derived from the BF-PSSM kernel, where, for a given a user-supplied width w, the BV-PSSM kernel considers the set of all possible $w\text{mer}$-pairs whose length ranges from one to a maximum w, i.e.,

$$\mathcal{P}_{1\ldots w} = \mathcal{P}_1 \cup \ldots \cup \mathcal{P}_w, \tag{16}$$

From this set $\mathcal{P}_{1\ldots w}$ the BV-PSSM kernel uses the greedy scheme employed by BF-PSSM to select a subset $\mathcal{P}'_{1\ldots w}$ of

$w$mer-pairs that form a high weight matching. The similarity between the pair of sequences is then computed as follows:

$$\text{BV-PSSM}_{X,Y}(w) = \sum_{(w\text{mer}(X,i),w\text{mer}(Y,j))\in\mathcal{P}'_{1\ldots w}} w\text{score}_{X,Y}(i,j). \tag{17}$$

Since for each position of X (and Y), $\mathcal{P}'_{1\ldots w}$ is constructed by including the highest scoring $w$mer for i that does not conflict with the previous selections, this scheme can automatically select the highest scoring $w$mer whose length can vary from one up to w; thus, achieving the desired effect.

Table 3: Comparison against different schemes for the superfamily-level classification problem.

Kernel	ROC	ROC50	mRFP
SVM-Fisher	0.773	0.250	0.204
SVM-Pairwise	0.896	0.464	0.084
LA-eig($\beta = 0.2$)	0.923	0.661	0.064
LA-eig($\beta = 0.5$)	0.925	0.649	0.054
LA-ekm(β = 0.5)	0.929	0.600	0.052
SVM-HMMSTR-Ave	–	0.640	0.038
SVM-HMMSTR-Max	–	0.618	0.043
SVM-HMMSTR-Hybrid	–	0.617	0.048
Mismatch	0.872	0.400	0.084
Profile(4,6)	0.974	0.756	0.013
Profile(5,7.5)	0.980	0.794	0.010
AF-PSSM(2)	0.978	0.816	0.013
BF-PSSM(2)	0.980	0.854	0.015
BV-PSSM(2)	0.973	0.855	0.018
SW-PSSM(3.0,0.750,1.50)	0.982	**0.904**	0.015
AF-GSM(6)	0.926	0.549	0.048
BF-GSM(6)	0.934	0.669	0.053
BV-GSM(6)	0.930	0.666	0.052
SW-GSM(B62,5.0,1,0.5)	0.948	0.711	0.039

The SVM-Fisher, SVM-Pairwise, LA-Kernel, and Mismatch results were obtained from [52]. The SVM-HMMSTR results were obtained from [18]and correspond to the best-performing scheme (the authors did not report ROC values). The Profile results were obtained locally by running the publicly available implementation of the scheme obtained from the authors. The ROC50 value of the best performing scheme has been underlined.

5.2 Performance Evaluation

The fold prediction algorithms can be evaluated using the sets of sequences obtained from the SCOP database [39]. The SCOP database is a manually curated protein structure database assigning proteins into hierarchically defined classes. The fold prediction problem in the context of SCOP can be defined as assigning a protein sequence to its correct fold. On a similar basis the remote homology problem can be defined as predicting the correct superfamily for a protein.

To evaluate the above techniques, remote homology detection is simulated by formulating it as a superfamily classification problem within the context of the SCOP database. The same dataset and classification problems[2] have been used in a number of earlier studies [35, 18, 52] allowing for direct comparisons of the relative performance of the various schemes. The data

[2]The dataset and classification problem definitions are available at http://www.cs.columbia.edu/compbio/svm-pairwise.

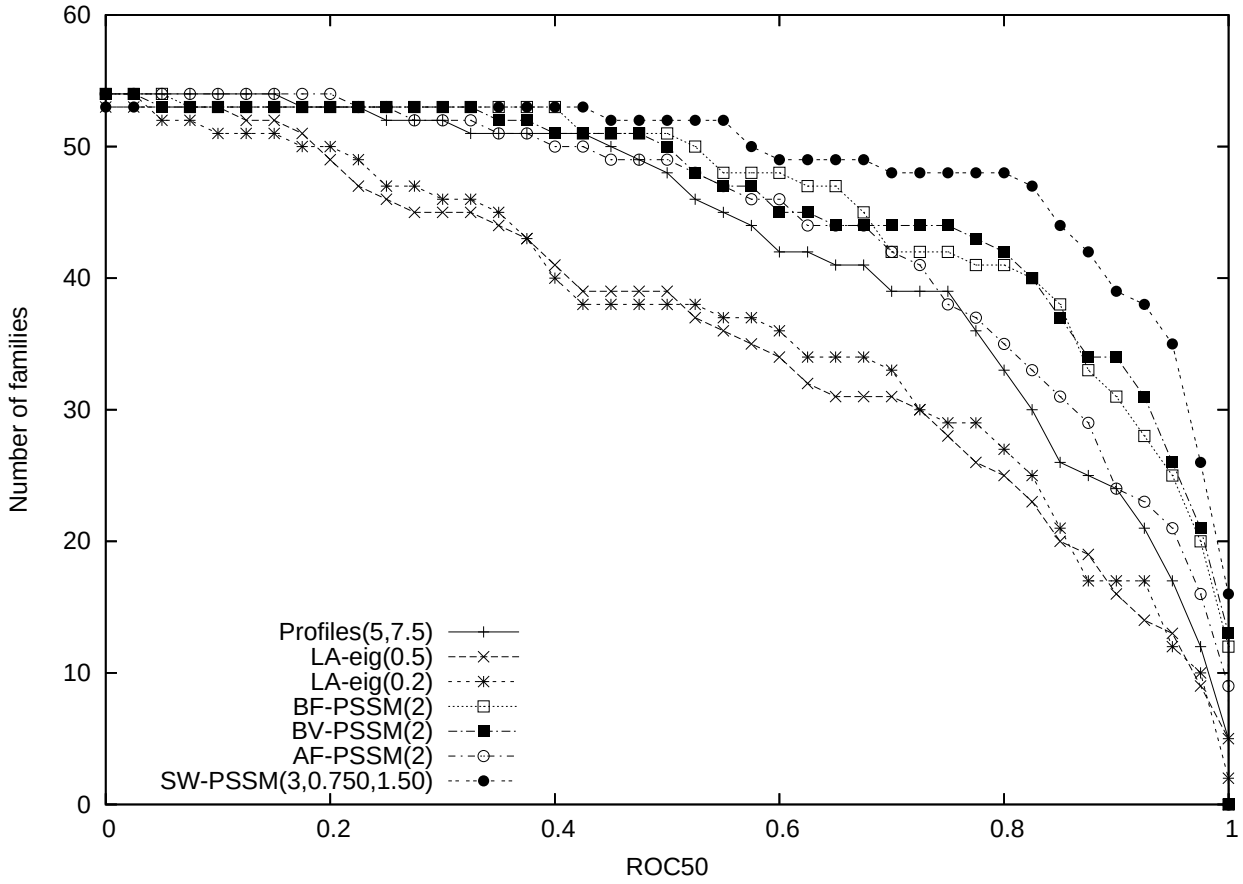


Figure 3: Comparison of the different SVM-based methods for remote homology detection on the SCOP 1.53 benchmark dataset. The graph plots the total number of families for which a given method exceeds the ROC-50 score threshold along the x-axis.

consists of 4352 sequences from SCOP version 1.53 extracted from the Astral database, grouped into families and superfamilies. The dataset is processed so that it does not contain any sequence pairs with an E-value threshold smaller than 10^{-25}. For each family, the protein domains within the family are considered positive test examples, and protein domains within the superfamily but outside the family are considered positive training examples. This yields 54 families with at least 10 positive training examples and 5 positive test examples. Negative examples for the family are chosen from outside of the positive sequences' fold, and are randomly split into training and test sets in the same ratio as the positive examples.

Employing the same dataset and overall methodology as in remote homology detection, we simulate fold detection by formulating it as a fold classification problem within the context of SCOP's hierarchical classification scheme. In this setting, protein domains within the same superfamily are considered positive test examples, and protein domains within the same fold but outside the superfamily are considered positive training examples. This yields 23 superfamilies with at least 10 positive training and 5 positive test examples. Negative examples for the superfamily are chosen from outside of the positive sequences' fold and split equally into test and training sets[3]. Since the positive test and training instances are members of different superfamilies within the same fold, this new problem is significantly harder than remote homology detection, as the sequences in the different superfamilies do not have any apparent sequence similarity [39]. The quality of these methods is evaluated by using the receiver operating characteristic (ROC) scores, the ROC50 scores, and the median rate of false positives (mRFP).

Table 3 and Table 4 compare the performance of the various kernel functions developed in this paper against that achieved by a number of previously developed schemes for the superfamily- and fold-level classification problems, respectively. In the case of the superfamily-level classification problem, the performance is compared against SVM-Fisher [19], SVM-Pairwise [35], and different instances of the LA-Kernel [52], SVM-HMMSTR [18], Mismatch [34], and Profile [31].

The results in these tables show that both the window- and local alignment-based kernels derived from sequence profiles (i.e., AF-PSSM, BF-PSSM, BV-PSSM, and SW-PSSM) lead to results that are in general better than those obtained by existing schemes. The performance advantage of these direct kernels is greater over existing schemes that rely on sequence information

[3]The classification problem definitions are available at http://bioinfo.cs.umn.edu/supplements/remote-homology/.

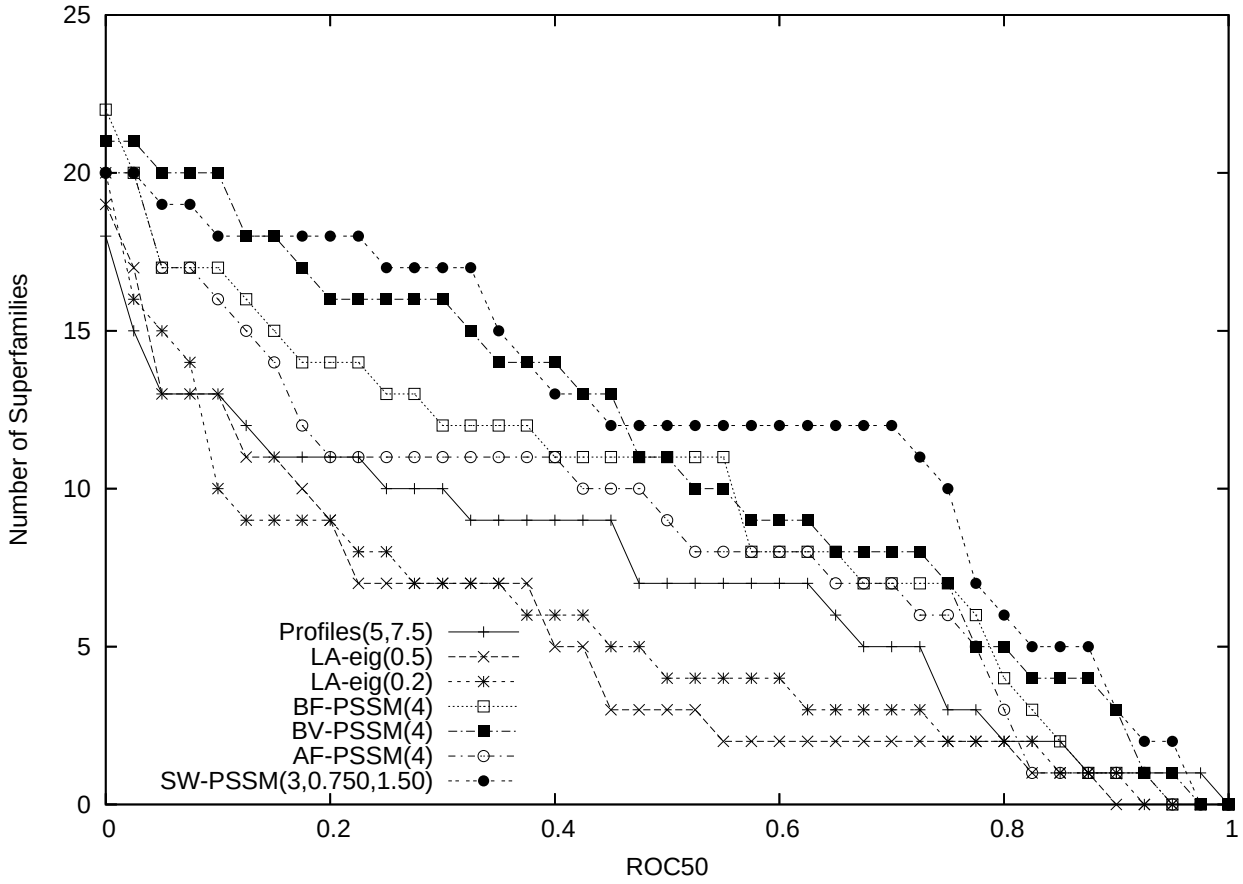


Figure 4: Comparison of the different SVM-based methods for fold detection on the SCOP 1.53 benchmark dataset. The graph plots the total number of superfamilies for which a given method exceeds the ROC-50 score threshold along the x-axis.

alone (e.g., SVM-Pairwise, LA-Kernels), but still remains significant when compared against schemes that either directly take into account profile information (e.g., SVM-Fisher, Profile) or utilize higher-level features derived by analyzing sequence-structure information (e.g., SVM-HMMSTR). Also, the relative advantage of profile-based methods over existing schemes is greater for the much harder fold-level classification problem over the superfamily-level classification problem. For example, the SW-PSSM scheme achieves ROC50 values that are 13.8% and 81.8% better than the best values achieved by existing schemes for the superfamily- and fold-level classification problems, respectively.

To get a better understanding of the relative performance of the various schemes across the different classes, Figures 3 and 4 plot the number of classes whose ROC50 are greater than a given threshold that ranges from 0 to 1. Specifically, Figure 3 shows the results for the remote homology detection problem, whereas Figure 4 shows the results for the fold detection problem. (Note that these figures contain only results for the schemes that we are able to run locally.) These results show that our profile-based methods lead to higher ROC50 values for a greater number of classes than either the Profile or LA-kernels, especially for larger ROC50 values (e.g. in the range of 0.6 to 0.95). Also, the SW-PSSM tends to consistently outperform the rest of the profile-based direct kernel methods.

In addition, the results for the BF-GSM, BV-GSM, and SW-GSM kernels that rely on the BLOSUM scoring matrices show that these kernel functions are capable of producing results that are superior to all of the existing non-profile-based schemes. In particular, the properly optimized SW-GSM scheme is able to achieve significant improvements over the best LA-Kernel-based scheme (7.6% higher ROC50 value) and the best SVM-HMMSTR-based scheme (15.1% higher ROC50 value).

From the evaluation of direct profile-based kernels for fold classification, three major observations can be made. First, as was the case with a number of studies on the accuracy of protein sequence alignment [38, 57, 36], the proper use of sequence profiles leads to dramatic improvements in the overall ability to detect remote homologs and identify proteins that share the same structural fold. Second, kernel functions that are constructed by directly taking into account the similarity between the various protein sequences tend to outperform schemes that are based on a feature-space representation (where each dimension of the space is constructed as one of k-possibilities in a k-residue long subsequence or using structural motifs (Isites) in the case of SVM-HMMSTR). This is especially evident by comparing the relative advantage of the window-based kernels over the Profile kernel. Third, time-tested methods for comparing protein sequences based on optimal local alignments (as well as

Table 4: Comparison against different schemes for the fold-level classification problem.

Kernel	ROC	ROC50	mRFP
LA-eig($\beta = 0.2$)	0.847	0.212	0.129
LA-eig($\beta = 0.5$)	0.771	0.172	0.193
Profile(4,6)	0.912	0.305	0.071
Profile(5,7.5)	0.924	0.314	0.069
AF-PSSM(4)	0.911	0.374	0.067
BF-PSSM(4)	0.918	0.414	0.060
BV-PSSM(4)	0.941	0.481	0.043
SW-PSSM(3.0,0.750,2.0)	0.936	**0.571**	0.054
AF-GSM(6)	0.770	0.197	0.217
BF-GSM(6)	0.822	0.240	0.157
BV-GSM(7)	0.845	0.244	0.133
SW-GSM(B62,5,1.0,0.5)	0.826	0.223	0.176

The results for the LA-Kernel were obtained using the publicly available kernel matrices that are available at the author's website. The Profile results were obtained locally by running the publicly available implementation of the scheme obtained from the authors. The ROC50 value of the best performing scheme has been underlined.

global and local-global alignments), when properly optimized for the classification problem at hand, lead to kernel functions that are in general superior to those based on either short subsequences (e.g., Spectrum, Mismatch, Profile, or window-based kernel functions) or local structural motifs (e.g., SVM-HMMSTR). The fact that these widely used methods produce good results in the context of SVM-based classification is reassuring as to the validity of these approaches and their ability to capture biologically relevant information.

6 Concluding Remarks

Predicting protein structure from primary sequence information is a challenging problem that has attracted and continues to attract attention from several fields of research. The current challenges within this problem stem from two factors. First, we still do not have a complete understanding of the basic physical principles that govern protein folding. Second, the number of experimentally resolved 3D protein structures remains small compared to the number of known proteins. Despite these obstacles, recent advances in applying machine learning to evolutionary analysis have significantly improved the quality of current structural predictions.

In this chapter we provided a brief overview of some of these machine learning techniques. Specifically, we examined the design of state-of-the-art kernel functions within a discriminative learning framework for secondary structure prediction, remote homology detection and fold recognition. We have given a flavor of string kernels along with the use of evolutionary information in our methods. Hopefully, increasingly better solutions to subproblems within complete structure prediction will lead to an accurate method for native fold prediction from sequence.

Acknowledgment

This work was supported by NSF EIA-9986042, ACI-0133464, IIS-0431135, NIH RLM008 713A, the Army High Performance Computing Research Center contract number DAAD19-01-2- 0014, and by the Digital Technology Center at the University of Minnesota.

References

[1] S. F. Altschul, W. Gish, E. W. Miller, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[2] S. F. Altschul, L. T. Madden, A. A. Schffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–402, 1997.

[3] Stephen Altschul, Warren Gish, Webb Miller, Eugene Myers, and David Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[4] C. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181:223–230, 1973.

[5] P. Baldi, Y. Chauvin, T. Hunkapiller, and M. McClure. Hidden markov models of biological primary sequence information. *PNAS*, 91:1053–1063, 1994.

[6] P. A. Bates and M. J. E Sternberg. Model building by comparison at casp3: Using expert knowledge and computer automation. *Proteins: Structure, Functions and Genetics*, 3:47 –54, 1999.

[7] P. Bourne and H. Weissig. *Structural Bioinformatics*. John Wiley & Sons, 2003.

[8] J. U. Bowie, R. Luethy, and D. Eisenberg. A method to identify protein sequences that fold into a known three-dimensional structure. *Science*, 253:797–815, 1991.

[9] K. M. S. Misura C. A. Rohl, C. E. M. Strauss and D. Baker. Protein structure prediction using rosetta. *Methods in Enzymology*, 383:66–93, 2004.

[10] Michael Collins. Parameter estimation for statistical parsing models: Theory and practice of distribution-free methods. In *New Developments in Parsing Technology*, pages 1–38. Kluwer, 2001.

[11] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

[12] J. A. Cuff and G. J. Barton. Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *PROTEINS: Structure, Function, and Genetics*, 34:508–519, 1999.

[13] A. Fiser, R. K. Do, and A.Sali. Modeling of loops in protein structures. *Protein Science*, 9:1753 – 1773, 2000.

[14] D. Frishman and P. Argos. Seventy-five percent accuracy in protein secondary structure prediction. *PROTEINS: Structure, Function, and Genetics*, 27:329–335, 1997.

[15] M. Gribskov, A. D. McLachlan, and D. Eisenberg. Profile analysis: detection of distantly related proteins. *PNAS*, 84:4355–4358, 1987.

[16] S. Henikoff and J. G. Henikoff. Amino acid subsitution matrices from protein blocks. *PNAS*, 89:10915–10919, 1992.

[17] Y. Hou, W. Hsu, M. L. Lee, and C. Bystroff. Efficient remote homology detection using local structure. *Bioinformatics*, 19(17):2294–2301, 2003.

[18] Y. Hou, W. Hsu, M. L. Lee, and C. Bystroff. Remote homolog detection using local sequence-structure correlations. *Proteins:Structure,Function and Bioinformatics*, 57:518–530, 2004.

[19] T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 7(1):95–114, 2000.

[20] D. T. Jones. Genthreader: an efficient and reliable protein fold recognition method for genomic sequences. *Journal of Molecular Biology*, 287:797–815, 1999.

[21] D. T. Jones, W. R. Taylor, and J. M. Thorton. A new approach to proetin fold recognition. *Nature*, 358:86–89, 1992.

[22] David T. Jones. Protein secondary structure prediction based on position-specific scoring matricies. *J. Mol. Biol.*, 292:195–202, 1999.

[23] K. Joo, J. Lee, S. Kim, I. Kum, J. ee, and S. Lee. Profile-based nearest neighbor method for pattern recognition. *J. of the Korean Physical Society*, 54(3):599–604, 2004.

[24] E. Huang K. T. Simons, C. Kooperberg and D. Baker. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. *Journal of Molecular Biology*, 268:209–225, 1997.

[25] W. Kabsch and C. Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

[26] K. Karplus, C. Barrett, and R. Hughey. Hidden markov models for detecting remote protein homologies. *Bioinformatics*, 14:846–856, 1998.

[27] K. Karplus, C. Barrett, and R. Hughey. Hidden markov models for detecting remote protein homologies. *Bioinformatics*, 14:846–856, 1998.

[28] K. Karplus, R. Karchin, J. Draper, J. Casper, Y. Mandel-Gutfreund, M. Diekhans, and R. Hughey. Combining local-structure, fold-recognition, and new fold methods for protein structure prediction. *PROTEINS: Structure, Function and Genetics*, 53:491–496, 2003.

[29] George Karypis. Better kernels and coding schemes lead to improvements in svm-based secondary structure prediction. Technical Report 05-028, Department of Computer Science, University of Minnesota, 2005.

[30] A. Krogh, M. Brown, I. Mian, K. Sjolander, and D. Haussler. Hidden markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.

[31] R. Kuang, E. Ie, K. Wang, K. Wang, M. Siddiqi, Y. Freund, and C. Leslie. Profile-based string kernels for remote homology detection and motif extraction. *Computational Systems Bioinformatics*, pages 152–160, 2004.

[32] J. Lee, S. Kim, K. Joo, I. Kim, and J. Lee. Prediction of protein tertiary structure using profesy, a novel method based on fragment assembly and conformational space annealing. *PROTEINS: Structure, function and bioinformatics*, 56:704–714, 2004.

[33] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for svm protein classification. *Proceedings of the Pacific Symposium on Biocomputing*, pages 564–575, 2002.

[34] C. Leslie, E. Eskin, W. S. Noble, and J. Weston. Mismatch string kernels for svm protein classification. *Advances in Neural Information Processing Systems*, 20(4):467–476, 2003.

[35] L. Liao and W. S. Noble. Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. *Proc. of the Intl. Conf. on Research in Computational Molecular Biology*, pages 225–232, 2002.

[36] M. Marti-Renom, M. Madhusudhan, and A. Sali. Alignment of protein sequences by their profiles. *Protein Science*, 13:1071–1087, 2004.

[37] F. S. Matthews. The structure, function and evolution of cytochromes. *Prog. Biophys. Mol. Biol.*, 45:1–56, 1975.

[38] D. Mittelman, R. Sadreyev, and N. Grishin. Probabilistic scoring measures for profile-profile comparison yield more accurate short seed alignments. *Bioinformatics*, 19(12):1531–1539, 2003.

[39] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.

[40] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[41] G. Pollastri and A. McLysaght. Porter: a new, accurate server for protein secondary structure prediction. *Bioinformatics*, 21:1719–1720, 2005.

[42] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *PROTEINS: Structure, Function, and Genetics*, 47:228–235, 2002.

[43] A. Porollo, R. Adamczak, M. Wagner, and J Meller. Maximum feasibility approach for consensus classifiers: Applications to protein structure prediction. In *CIRAS*, 2003.

[44] D. Przybylski and B. Rost. Alignments grow, secondary structure prediction improves. *PROTEINS: Structure, Function, and Genetics*, 46:197–205, 2002.

[45] H. Rangwala and G. Karypis. Profile based direct kernels for remote homology detection and fold recognition. *Bioinformatics*, 21(23):4239–4247, 2005.

[46] V. Robles, Pedro Larranaga, Jose M. Pena, Ernestinae Menasalvas, Maria S. Perez, Vanessa Herves, and Anita Wasilewska. Bayesian network multi-classifiers for protein secondary structure prediction. *Artificial Intelligence in Medicine*, 31:117–136, 2004.

[47] B. Rost. unpublished.

[48] B. Rost and V. A. Eyrich. EVA: Large-scale analysis of secondary structure prediction. *PROTEINS: Structure, Function, and Genetics*, Suppl. 5:192–199, 2001.

[49] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.*, 232:584–599, 1993.

[50] B. Rost, C. Sander, and R. Schneider. Redefining the goals of protein secondary structure prediction. *J. Mol. Biol.*, 235:13–26, 1994.

[51] Burkhard Rost. Review: Protein secondary structure prediction continues to rise. *Journal of Structural Biology*, 134:204–218, 2001.

[52] H. Saigo, J. P. Vert, N. Ueda, and T. Akutsu. Protein homology detection using string alignment kernels. *Bioinformatics*, 20(11):1682–1689, 2004.

[53] A. A. Salamov and V. V. Solovyev. Protein secondary structure prediction using local alignments. *J. Mol. Biol.*, 268:31–36, 1997.

[54] A. Semla, C. Venclovas, Krzysztof Fidelis, and B. Rost. A modified definition of sov, a segment-based measure for protein secondary structure prediction assessment. *PROTEINS: Structure, Function, and Genetics*, 34:220–223, 1999.

[55] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[56] V. Vapnik. *Statistical Learning Theory*. John Wiley, New York, 1998.

[57] G. Wang and R. L. Dunbrack JR. Scoring profile-to-profile sequence alignments. *Protein Science*, 13:1612–1626, 2004.

[58] Y. Wu and E. Y. Chang. Distance-function design and fusion for sequence data. *Proc. of the Thirteenth ACM conference on Information and knowledge management*, pages 324–333, 2004.