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 05-028

Better Kernels and Coding Schemes Lead to Improvements in SVM-based Secondary Structure Prediction

George Karypis

July 29, 2005

TECHNICAL REPORT

TR no. 05–028, 2005
Pages 1–7

Better Kernels and Coding Schemes Lead to Improvements in SVM-based Secondary Structure Prediction

George Karypis

Department of Computer Science & Engineering, University of Minnesota, Minneapolis, MN 55455

ABSTRACT

Motivation: The accurate prediction of a protein's secondary structure plays an increasingly critical role in predicting its function and tertiary structure, as it is utilized by many of the current state-of-the-art methods for remote homology, fold recognition, and *ab initio* structure prediction.

Methods: We developed a new secondary structure prediction algorithm called YASSPP that uses a pair of cascaded models constructed from two sets of binary SVM-based models. YASSPP uses an input coding scheme that combines both position-specific and non-position specific information, utilizes a kernel function designed to capture the sequence conservation signals around the local window of each residue, and constructs a second-level model by incorporating both the three-state predictions produced by the first-level model and information about the original sequence.

Results: Experiments on three standard datasets (RS126, PB513, and EVA common subset 4) show that YASSPP is capable of producing the highest Q_3 and SOV scores than that achieved by existing widely used schemes such as PSIPRED, SSPro 4.0, SAM-T99sec, as well as previously developed SVM-based schemes. On the EVA dataset it achieves a Q_3 and SOV score of 79.34% and 78.65%, which are considerably higher than the best reported scores of 77.64% and 76.05%, respectively.

Availability: The YASSPP prediction server is available at http://bioinfo.cs.umn.edu/yasspp.

Contact: karypis@cs.umn.edu.

1 INTRODUCTION

Breakthroughs in large-scale sequencing have led to a surge in the available protein sequence information that has far out-stripped our ability to experimentally characterize their functions and tertiary structures. As a result, researchers are increasingly relying on computational techniques to classify these sequences into functional and structural families and to predict their three dimensional structure. Algorithms for protein secondary structure prediction play an essential role in many of these techniques [16]. This was evident in the most recent CASP6 competition, in which predicted secondary structure information was an integral part of the best performing schemes for the comparative modeling, fold-recognition, and new fold prediction tasks.

A large number of secondary structure prediction algorithms have been developed, and over the years, their prediction accuracy has been continuously improved. Many algorithms can nowadays 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 [26], 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 [27, 5, 11] predict the secondary structure by identifying a set of similar sequence-fragments with known secondary structure; the model-based approaches [24, 10, 21, 19], employ sophisticated machine learning techniques to learn a predictive model trained on sequences of known structure; whereas the meta-predictor-based approaches [4, 18] predict the structure by combining the predictions produced by different neighbor and/or model-based techniques. The near real-time evaluation of many of these methods performed by the EVA server [23] shows that the model-based approaches tend to produce statistically better results than the neighbor-based schemes, which is further improved by some of the more recently developed meta-predictor-based approaches [18].

Historically, the most successful model-based approaches such as PHD [24], PSIPRED [10], and SSPro [19], 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 (SVM) [30]. Even though the initial performance of these schemes was not competitive with that achieved by the best NN-based schemes [8], recent advances have lead to the development of algorithms [14, 31, 6] whose performance is comparable and sometimes better than that achieved by NN-based schemes.

In this paper we present a secondary structure prediction algorithm called YASSPP that further improves the performance achieved by SVM-based methods. YASSPP employs the common framework for secondary structure prediction that is based on a pair of cascaded models. The first-level model, often referred to as *sequence-to-structure* model, computes a three-state prediction for each position by taking into account the sequence information around that position, whereas the second-level model, often referred to as *structure-to-structure* model, computes the final secondary structure assignment by taking into account the predictions computed by the first model. Each of these models is constructed using three sets of binary SVM classifiers employing a one-vs-rest learning approach.

YASSPP improves prediction performance through the incorporation of a number of new ideas. It uses an exponential kernel function derived by combining a normalized second order kernel in which the contribution of each position is inversely proportional to its distance from the central residue. It constructs the second-level model by incorporating both the predicted secondary structure as well as information from the original input sequence; thus, using what can be considered as a *sequence+structure-to-structure* model. It uses a coding scheme for the input sequence that in addition to position specific information obtained using PSI-BLAST [1], also incorporates non-position specific information obtained using the BLOSUM62 [7] scoring matrix. Finally, YASSPP uses a loss function that assigns different misclassification costs to each secondary structure state based on its relative size in the training set, which accounts for the unbalanced class-size distribution.

Experiments on the widely used RS126 and PB513 benchmark datasets and on a dataset obtained from the EVA server (common subset #4) show that YASSPP is consistently more accurate than existing state-of-the-art SVM- and NN-based secondary structure prediction algorithms. On PB513 YASSPP achieves Q_3 and SOV scores of 77.78% and 74.99%, respectively, whereas on the EVA dataset its Q_3 and SOV scores are 79.34% and 78.65%, respectively. These latter results represent an improvement of 2.2% and 3.4%, respectively, over that achieved by the next best performing algorithm.

2 METHODS AND ALGORITHMS

2.1 Secondary Structure Definition

The secondary structure information for each residue was obtained using the DSSP [12], which assigns each residue to one of eight 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). We 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 eight-to-three state reduction scheme is used by most secondary structure prediction methods [10, 23] and allows us to compare YASSPP's results with those produced by other schemes.

2.2 PSSM Representation & Generation

The position specific score matrix of a sequence X of length n is represented by a $n \times 20$ matrix. The rows of this matrix correspond to the various positions in X and the columns correspond to the 20 distinct amino acids. The position specific score matrices used by YASSPP were generated using the latest version of the PSI-BLAST algorithm [1] (available in NCBI's blast release 2.2.10), and were derived from the multiple sequence alignment constructed after five iterations using an e value of 10^{-2} for initial and subsequent sequence inclusions (i.e., we used `blastpgp -j 5 -e 0.01 -h 0.01`). The PSI-BLAST was performed against NCBI's nr database that was downloaded in November of 2004 and contained 2,171,938 sequences.

2.3 Algorithm

The overall structure of YASSPP is similar to that used by many existing secondary structure prediction algorithms like PHD and PSIPRED. 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 the these binary classifiers (i.e., $M_1^{C/\bar{C}}$, $M_1^{E/\bar{E}}$, and $M_1^{H/\bar{H}}$), whereas the predicted secondary state of the L_2 model corresponds to the state whose corresponding binary classifier achieves the maximum value. That is,

$$\text{Predicted state} = \underset{x \in \{C,E,H\}}{\operatorname{argmax}} (M_2^{x/\bar{x}}). \quad (1)$$

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 of X centered at position i. The parameter w determines the length of the local environment around the ith sequence position to be used while building the model, and its proper value is determined experimentally. YASSPP uses the same value of w for all binary classifiers used by the L_1 and L_2 models. We will refer to these subsequences as $w$mers. During secondary structure prediction, a similar approach is used to construct a $wmer$ around each position i of a sequence X with unknown secondary structure (we will refer to such sequence as a *query* sequence).

2.4 Input Sequence Coding

We used two different schemes to code the $w$mers for the L_1 model and two different schemes for the L_2 model.

L_1's first coding scheme represents each $wmer$ 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. We will refer to these as the P and the PB coding schemes, respectively.

The primary motivation behind the second coding scheme is to improve the classification accuracy (in conjunction with the kernel function described later) in cases in which the query sequence does not have a sufficiently large number of homologous sequences in nr, and/or PSI-BLAST failed to compute a correct alignment for some segments of the sequence. By augmenting the $wmer$ 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 matrices P_x, B_x, and S_x. We will refer to these as the PS and the PBS coding schemes, respectively. Note that the information captured by these two coding schemes are different than those used by existing secondary structure prediction algorithms, as the latter consist only of S_x and ignore any information about the original sequence.

For each coding scheme the rows of the matrices that correspond to $w\text{mer}$ positions extending past the beginning and end of the input sequence are set to zero.

Even though each coding scheme of L_1 can be combined with either of the two coding schemes for L_2, in YASSPP we investigated only two combinations: P with PS, and PB with PBS, which will be denoted as $P + PS$ and $PB + PBS$, respectively.

2.5 Kernel Functions

In developing YASSPP, a considerable effort was spent in designing and evaluating various kernel functions for use by the binary SVM classifiers of the L_1 and L_2 models. This effort led us to construct 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 that we used is given by

$$\mathcal{K}(x, y) = \exp\left(1.0 + \frac{\mathcal{K}_1(x, y)}{\sqrt{\mathcal{K}_1(x, x)\,\mathcal{K}_1(y, y)}}\right), \qquad (2)$$

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

$$\mathcal{K}_1(x, y) = \mathcal{K}_2^{cs}(x, y) + (\mathcal{K}_2^{cs}(x, y))^2, \qquad (3)$$

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:

$$\begin{aligned}
\mathcal{K}_2^{P}(x, y) &= \sum_{j=-w}^{j=w} \frac{P_x(j,:)P_y^t(j,:)}{1+|j|}, & (4)\\
\mathcal{K}_2^{PB}(x, y) &= \mathcal{K}_2^{P}(x, y) + \sum_{j=-w}^{j=w} \frac{B_x(j,:)B_y^t(j,:)}{1+|j|}, & (5)\\
\mathcal{K}_2^{PS}(x, y) &= \mathcal{K}_2^{P}(x, y) + 50\sum_{j=-w}^{j=w} \frac{S_x(j,:)S_y^t(j,:)}{1+|j|}, & (6)\\
\mathcal{K}_2^{PBS}(x, y) &= \mathcal{K}_2^{PB}(x, y) + 50\sum_{j=-w}^{j=w} \frac{S_x(j,:)S_y^t(j,:)}{1+|j|}. & (7)
\end{aligned}$$

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 $+w$).

A number of observations can be made by analyzing the various kernel functions involved in the above definitions. First, by linearizing matrices P, B, and S, we can see that $\mathcal{K}_2^{cs}(x, y)$ is a linear function corresponding to the dot-product of the linearized representation of x and y. Depending on the choice of the coding scheme, these dot-products involve $20(2w + 1)$, $40(2w + 1)$, $23(2w + 1)$, or $43(2w + 1)$ dimension vectors. Second, the contribution of each $w\text{mer}$ position in $\mathcal{K}_2^{cs}(x, y)$ decreases linearly with respect to its distance from the central residue (i.e., the residue that defines the class or whose class needs to be predicted). This was motivated by the fact that the secondary structure state of a residue is in general more dependent on the nearby sequence positions than the positions that are further away [3]. Third, the contribution of the S matrix in the kernels used for the L_2 model (i.e., PS and PBS coding schemes) is weighted higher than the corresponding contributions of the P and B matrices. This is done by scaling its weight by 50. This value was determined experimentally by testing a number of scaling factors in the set $\{1, 5, 25, 50, 75, 100\}$. Note that a similar optimization can be performed for assigning different weights to the contributions of the P and B matrices. However, we did not perform such an optimization. Fourth, since $\mathcal{K}_2^{cs}(x, y)$ is a linear function, the $\mathcal{K}_1(x, y)$ is a kernel corresponding to a second-order polynomial. This allows the kernel function to capture pairwise dependencies among the residues used at various positions within each $w\text{mer}$, and we found that this leads to better results over the linear function. This observation is also supported by other research as well [31]. Fifth, the exponential structure of $\mathcal{K}(x, y)$ allow us to capture highly non-linear relations.

2.6 Unbalanced Classes

In the absence of well-separable classes, SVM learns a model that minimizes the number of examples that get misclassified (i.e., number of errors). In cases in which there is a large difference in the sizes of the positive and negative classes, this minimization can potentially be achieved by learning a model that is biased towards the largest class. When the outputs of such binary SVM classifiers are used to build a multi-class classifier, as it is the case for the three-state secondary structure prediction problem, such biases may decrease the overall classification performance. Unfortunately, in the context of secondary structure prediction, due to the higher frequency of the coil state over the strand and helix states, such unbalanced class scenarios do occur.

One way of overcoming this problem is to convert the raw functional outputs of the binary SVM classifiers into probability values. A popular method used for achieving this is to fit the output of the SVM to a sigmoid function, and use this fit to compute probabilities [29]. Our experimentation with this approach did not improve the overall results and for this reason we adapted an alternate scheme that associates different misclassification costs to the examples of the three classes; thus, trying to prevent the SVM from introducing a class-size bias in the first place.

The misclassification cost assigned to each class is computed as follows. Let n_i^o, be the (observed) number of residues at state i in the training set, where $i \in \{C, E, H\}$, and let N be the total number of resides over the three states. The effective number of residues n_i^e at state i is defined to be

$$n_i^e = n_i^o + \frac{N}{3}. \qquad (8)$$

This definition includes both the observed number of residues as well as the expected number of residues $N/3$, under the assumption that all three states occur with the same probability. Then the misclassification cost mc_i associated with state i is given by solving

$$n_i^e\,\text{mc}_i = \sum_{j\neq i} n_j^e \;\;\Rightarrow\;\; \text{mc}_i = \frac{1}{n_i^e}\sum_{j\neq i} n_j^e. \qquad (9)$$

This ensures that the overall cost of the positive class (i.e., number of instances multiplied by the misclassification cost for that class) is equal to the overall cost of the negative class.

3 EXPERIMENTAL DESIGN

3.1 Dataset Description

The performance of YASSPP was evaluated on three different datasets. The first is the RS126 dataset, originally developed by Rost and Sander [24], which contains 126 sequences. The second is the CB513 dataset, originally developed by Cuff and Borton [4], which contains 513 non-homologous sequences [1]. The third is a dataset obtained from the EVA server [23], which compares a number of prediction servers using the sequences deposited in the PDB every week. In particular, we used the set labeled "common4" (http://cubic.bioc.columbia.edu/eva/sec/set_com4.html), which contains 165 sequences, most of which have been tested against a number of different secondary structure prediction methods. We will refer to this dataset as EVAc4.

These three datasets were used to experimentally evaluate the secondary structure prediction performance of YASSPP as follows. First, the RS126 and CB513 datasets were used to study the impact of its various input coding schemes, kernel/learning choices, and optimize its parameters. Second, the EVAc4 dataset was used to assess YASSPP's performance on an independent dataset and compare it against that achieved by other popular algorithms.

3.2 Prediction Accuracy Assessment

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). These measures are among the most widely used performance assessment measures for secondary structure prediction, and because they are also reported by the EVA server we can make direct comparisons with existing schemes.

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 [24]. The SOV is a segment-level measure of the overall prediction accuracy. This measure is initially introduced in [25] and subsequently refined in [28]. The SOV values produced by these two definitions are different and cannot be directly compared. For our assessment purposes, we use the most recent definition of the SOV measure (also referred to as SOV99), as it allows us to perform comparisons with recent schemes and with the results reported by the EVA server. Matthews correlation coefficients [15] provide a per-state measure of prediction performance and for a particular state $i \in \{C, E, H\}$ it is given by

$$C_i = \frac{p_i n_i - u_i o_i}{\sqrt{(p+i+u_i)(p_i+o_i)(n_i+u_i)(n_i+o_i)}}, \qquad (10)$$

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 [24] 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.

[1] Both the RS126 and CB513 datasets can be obtained from http://www.compbio.dundee.ac.uk/~www-jpred/data/pred_res/.

3.3 SVM Training & Testing

We use the publicly available support vector machine tool SVM^{light} [9] which implements an efficient soft margin optimization algorithm. We use the default parameters for solving the quadratic programming problem, and we use a regularization parameter of $C = 1/e^2 = 0.1353$, which is the default value used by SVM^{light} and computed as the average of $1/(||x||^2)$.

We used two different approaches to predict these three datasets. In the case of RS126 and CB513, we followed a seven-fold cross-validation framework, in which each one of the seven folds was predicted using a model that was built on the remaining six folds. This approach allowed us to directly compare our results against those obtained by earlier methods [8, 14] that used a similar seven-fold cross-validation approach.

In the case of the EVAc4, we used a model that was trained on a set of proteins derived from SCOP 1.67 [17] as follows. We used Astral [2] to obtain a set of protein domains whose pairwise sequence identity was less than 25%. This resulted in a set of 4,993 domains that belong to 3,971 proteins. This set was further pruned by removing all proteins that were identified to have greater than a 25% identity with at least one of the sequences in EVAc4. This pruning step left 3,223 proteins, which was used to train YASSPP.

4 RESULTS

4.1 RS126 and CB513 Datasets

We investigate the impact of YASSPP's parameters by performing a number of experiments in which we (i) vary the length of the $wmer$, (ii) disable certain aspects of the kernel functions, (iii) eliminate the class-size sensitive misclassification costs, and (iv) use different input coding schemes. The key results of these studies are summarized in the subsequent sections.

4.1.1 Window Length. Table 4.1.1 shows the performance achieved by YASSPP for different length $w$mers ranging from nine to nineteen residues long (w =4–9). These results show that the best performance is achieved for $w$mers that are 13 or 15 residues long, which is in agreement with the results reported in previous studies. The results also illustrate that as the length of the window increases, the performance of YASSPP^{-pw} reduces faster than that of YASSPP, verifying the initial motivation behind YASSPP's distance-sensitive position weighting scheme.

4.1.2 Kernel & Learning Parameters. Table 4.1.2 shows the impact to YASSPP's performance by disabling certain elements of its kernel function and by eliminating the class-size sensitive misclassification costs. These results show that each one of these parameters lead to an improvement in the overall prediction accuracy across the two datasets. Among them, the gains achieved by using a coding scheme for the L_2 model that incorporates the amino acid composition of each $wmer$ are the highest, whereas the gains achieved by the distance-sensitive position weighting schemes are the lowest.

4.1.3 Input Sequence Coding. Table 4.1.3 shows the effect of the different input coding schemes to YASSPP's overall performance. In

Table 1. Effect of the window length on the performance of YASSPP for the RS126 dataset.

	YASSPP					YASSPP^{-pw}				
w	Q_3	SOV	C_C	C_E	C_H	Q_3	SOV	C_C	C_E	C_H
4	76.67	70.79	0.54	0.62	0.70	76.65	70.56	0.54	0.63	0.70
5	76.94	70.97	0.55	0.63	0.71	76.76	70.78	0.54	0.63	0.70
6	77.08	71.20	0.55	0.62	0.71	76.70	70.66	0.54	0.63	0.70
7	77.08	71.27	0.55	0.62	0.71	76.54	70.47	0.54	0.63	0.70
8	76.98	71.14	0.55	0.63	0.71	76.29	70.15	0.53	0.62	0.70
9	76.89	71.01	0.54	0.63	0.71	76.07	69.73	0.53	0.62	0.70

The results labeled YASSPP are obtained using the kernel functions as described in Section 2.5, whereas the YASSPP^{-pw} were obtained by weighting each position of the $wmer$ equally in Equations 4–7 (i.e., no distance-sensitive decrease of each position's contribution). The reported values correspond to the averages over the 126 sequences obtained using both the $P+PS$ and $PB+PBS$ input coding schemes.

Table 2. Effect of various kernel and learning parameters on the performance of YASSPP.

RS126 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
YASSPP	77.58	72.04	0.370	0.560	0.628	0.713
YASSPP^{-pw}	77.24	71.63	0.360	0.552	0.626	0.710
YASSPP^{-cw}	77.00	70.82	0.362	0.551	0.622	0.706
YASSPP$^{-P/PB}$	76.64	71.09	0.354	0.539	0.626	0.704
CB513 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
YASSPP	77.65	74.62	0.393	0.575	0.638	0.703
YASSPP^{-pw}	77.48	74.35	0.388	0.572	0.630	0.699
YASSPP^{-cw}	77.58	74.54	0.390	0.578	0.631	0.696
YASSPP$^{-P/PB}$	77.07	73.95	0.385	0.566	0.628	0.694

YASSPP^{-pw}, YASSPP^{-cw}, and YASSPP$^{-P/PB}$ are derived from YASSPP by disabling some of its features as follows. YASSPP^{-pw} does not use distance-sensitive position weighting; YASSPP^{-cw} does not use class-size sensitive misclassification costs (i.e., the misclassification costs for all binary classifiers was one); and YASSPP$^{-P/PB}$ uses only the S matrix when constructing the binary classifiers for the L_2 model and does not use either the PSSM-based coding or the BLOSUM62-based coding. For YASSPP^{-pw} and YASSPP^{-cw} the reported values correspond to the averages obtained using both the $P+PS$ and $PB+PBS$ input coding schemes and w ranging from four to nine. For YASSPP $^{-P/PB}$ the reported values correspond to the averages obtained using both the $P+S$ and $PB+S$ input coding schemes and w ranging from four to nine.

general, by augmenting the traditional input coding schemes to also include non-position specific information, we are able to achieve an improvement in the overall classification performance. However, this improvement is not uniform across the two datasets and performance assessment measures, as the $P+PS$ coding scheme achieves better SOV, C_C, and C_E values for the RS126 dataset and better C_E values for the CB513 dataset over the $PB+PBS$ coding scheme.

4.1.4 Comparison with Other Methods. Table 4.1.4 compares the performance achieved by YASSPP with that achieved by SVM-freq [8], SVMpsi [14], and PMSVM [6], three recently developed SVM-based secondary structure prediction methods.

From these results we can see that both YASSPP$_{P+PS}$ and YASSPP$_{PB+PBS}$ achieve better results than any of the other three schemes. In terms of Q_3 and SOV, these improvements are also statistically significant across the different methods and datasets. Among these methods, PMSVM is more similar to YASSPP$_{P+PS}$ as it uses a pair of cascaded models, utilizes PSSMs, employs an input coding scheme for the L_1 model that is similar to P. Thus, the improvement achieved by YASSPP$_{P+PS}$ over PMSVM can be attributed to the different kernel function (PMSVM uses an rbf kernel function), the class-size sensitive misclassification cost, and the coding used for the L_2 model.

Table 3. Effect of the feature space on the performance of YASSPP.

RS126 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
YASSPP$_{P+PS}$	77.03	71.32	0.360	0.548	0.632	0.712
YASSPP$_{PB+PBS}$	76.85	70.81	0.359	0.546	0.617	0.701
CB513 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
YASSPP$_{P+PS}$	77.54	74.32	0.390	0.571	0.642	0.697
YASSPP$_{PB+PBS}$	77.72	74.98	0.395	0.578	0.633	0.707

YASSPP$_{P+PS}$ uses the $P+PS$ input coding and the YASSPP$_{PB+PBS}$ uses the $PB+PBS$ input coding. The reported values correspond to the averages obtained over different values of w ranging from four to nine.

Table 4. Comparative performance of YASSPP against other methods.

RS126 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
SVMfreq	71.20	—	—	0.510	0.520	0.620
SVMpsi	76.10	72.00	—	—	—	—
YASSPP$_{P+PS}$	77.63	72.25	0.371	0.559	0.637	0.721
ErrSig	0.82	1.34	0.015	0.015	0.022	0.020
YASSPP$_{PB+PBS}$	77.68	72.04	0.373	0.562	0.617	0.708
ErrSig	0.84	1.34	0.015	0.015	0.023	0.021
CB513 Dataset						
Scheme	Q_3	SOV	Info	C_C	C_E	C_H
SVMfreq	73.50	—	—	0.540	0.530	0.650
SVMpsi	76.60	73.50	—	0.560	0.600	0.680
PMSVM	75.20	—	—	0.610	0.610	0.710
YASSPP$_{P+PS}$	77.53	74.25	0.389	0.571	0.642	0.696
ErrSig	0.41	0.63	0.007	0.007	0.011	0.010
YASSPP$_{PB+PBS}$	77.78	74.99	0.396	0.580	0.634	0.710
ErrSig	0.42	0.63	0.007	0.007	0.011	0.010

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 SVMpsi, SVMfreq, and PMSVM were obtained using a similar seven-fold cross validation approach and are directly comparable with YASSPP's results. Entries marked with '—' indicate results that could not be obtained from the publications of the respective methods.
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 ($\sigma/\sqrt{N}$).

Table 5. 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.

4.2 EVAc4 Dataset

Table 4.2 compares the performance achieved by YASSPP against that achieved by PHDpsi [21], PSIPRED [10], SAM-T99sec [13], PROFsec [22], SCRATCH [19], SSPro4 [19], and SABLE2 [20]. 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 4.2 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 2.2% better than the second best-performing scheme in terms of Q_3 (SAM-T99sec), and an SOV score of 78.65%, which is 3.4% better than the second best performing scheme in terms of SOV (PSIPRED).

Comparing the two different versions of YASSPP, we can see that unlike the results reported earlier for RS126 and PB513, YASSPP$_{PB+PBS}$ performs considerably better than YASSPP$_{P+PS}$. On the entire dataset, its prediction performance is better by one percentage point in terms of Q_3, and better by 1.45 percentage points in terms of SOV. To better understand the source of this improvement, we analyzed the two sets of predictions and compared the positions that both schemes predicted correctly with those that were predicted correctly by only one of the two schemes. This comparison was performed by analyzing the amount of information that is captured at each position of the profile, which provides a quantitative measure of each position's sequence conservation among the homologous sequences used to construct the PSSM. For this purpose we used the "information per position" measure that is computed by PSI-BLAST itself and is stored at the generated PSSM file.

Table 6. Analysis of the correct predictions computed by YASSPP $_{P+PS}$ and YASSPP$_{PB+PBS}$ on the EVAc4 dataset.

	P & PB			P & $\neg PB$			$\neg P$ & PB		
w	C	E	H	C	E	H	C	E	H
0	0.71	0.75	0.67	0.72	0.73	0.75	0.62	0.67	0.50
1	0.70	0.75	0.66	0.68	0.79	0.74	0.65	0.68	0.51
2	0.69	0.75	0.66	0.67	0.78	0.76	0.67	0.68	0.51
3	0.69	0.75	0.67	0.68	0.76	0.76	0.67	0.67	0.51

The average information per position of different length $w$mers centered at each residue that was correctly predicted by both methods (P & PB), correctly predicted only by YASSPP$_{P+PS}$ (P & $\neg PB$), and correctly predicted only by YASSPP $_{PB+PBS}$ ($\neg P$ & PB). The results are presented based on the secondary structure state of the central residue. The $w = 0$ results correspond to the $wmer$ consisting of just the position itself. The average information for longer $w$mers was computed by first computing the average information for each $wmer$ and then reporting the average of these averages.

The results of this analysis are summarized in Table 4.2, which shows the average information for positions that were correctly predicted by both schemes, positions that were correctly predicted only by YASSPP$_{P+PS}$, and positions that were correctly predicted only by YASSPP$_{PB+PBS}$. From these results we can see that the positions that are predicted correctly only by YASSPP $_{PB+PBS}$ have considerably less information than those predicted correctly by either YASSPP$_{P+PS}$ alone or by both schemes. This is true across all three secondary structure states, and it is more pronounced for helices and for coils. Even though there are many reasons why such low information positions can occur in the PSSM, one reason is the lack of a sufficient number of strong homologous sequences. This is indeed the case for the 165 sequences of the EVAc4 dataset, for which PSI-BLAST was unable to find more than 20 homologous sequences for each one of 51 query sequences, and could find at least 100 homologous sequences for only 68 query sequences. Thus, by augmenting the input coding of each $wmer$ with the BLOSUM62 information of their residues, YASSPP$_{PB+PBS}$ is able to correctly predict a larger number of such low information positions, and to some degree overcome the information loss due to insufficient number of homologous sequences.

5 DISCUSSION AND CONCLUSION

This paper presented and experimentally evaluated a new protein secondary structure prediction algorithm YASSPP that uses a pair of cascaded SVM-based models to compute a three-state prediction

(C, E, H). The experimental evaluation using three standard benchmark datasets showed that YASSPP is capable of producing superior prediction performance, measured both in terms of the three-state prediction accuracy (Q_3) and the segment overlap score (SOV), than that achieved by existing widely used schemes such as PSIPRED, SSPro, SAM-T99sec, as well as previously developed SVM-based schemes such as SVMfreq and SVMpsi.

These improvement gains can be attributed to three different factors. First, YASSPP uses a kernel function that is designed to capture the sequence conservation signals around the local window of each residue. This kernel function captures position information, interdependencies between positions, and a distance-based position weighting scheme, all of which have been shown to have some correlation with secondary structure [3]. Even though each of these elements have been used in the past in various secondary structure prediction algorithms, to the best of our knowledge, YASSPP is the first scheme that explicitly couples all of them together.

Second, YASSPP's L_2 model in addition to the three-state predictions produced by the L_1 model also combines information about the original sequence as captured by its PSSM-based (and BLOSUM62-based) coding. This additional information allows SVM to explicitly capture dependencies between amino acid composition and predicted secondary structure of different positions. These dependencies are captured by the second order (Equation 3) and the exponential kernel (Equation 2). The results reported in Table 4.1.2 show that by doing so, YASSPP is able to achieve measurable prediction improvements.

Third, YASSPP$_{PB+PBS}$ uses an input coding scheme that combines both position-specific and non-position specific information for each sequence. In doing so, it can learn a model that depends on information being derived from these two sources as well as their interdependencies. The latter is achieved via YASSPP's kernel function. The experiments with the EVAc4 dataset and their analysis suggest that this combined input coding scheme can lead to accuracy gains for sequence positions with low information per position. This often occurs when there is not a sufficiently large number of strong homologous sequences covering this position and/or the profile generation algorithm failed to produce correct alignments for them.

ACKNOWLEDGMENT

This work was supported by NSF EIA-9986042, ACI-9982274, ACI-0133464, ACI-0312828, IIS-0431135, 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, 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.

[2] J. M. Chandonia, G. Hon, N. S. Walker, L. Lo Conte, P. Koehl, M. Levitt, and S. E. Brenner. The astral compendium in 2004. *Nucleic Acids Research*, 32:D189–D192, 2004.

[3] G. E. Crooks, J. Wolfe, and S. E. Brenner. Measurements of protein sequence-structure correlations. *PROTEINS: Structure, Function, and Genetics*, 57:804–810, 2004.

[4] 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.

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

[6] J. Guo, H. Chen, Z. Sun, and Y. Lin. A novel method for protein secondary structure prediction using dual-layer svm and profiles. *PROTEINS: Structure, Function, and Genetics*, 54:738–743, 2004.

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

[8] S. Hua and Z. Sun. A novel method of protein secondary structure prediction with high segment overlap measure: Support vector machine approach. *J. Mol. Biol.*, 308:397–407, 2001.

[9] T. Joachims. Making large-scale svm learning practical. *Advances in Kernel Methods - Support Vector Learning, B. Schlkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999*, 1999.

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

[11] 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.

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

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

[14] H. Kim and H. Park. Protein secondary structure prediction based on an improved support vector machine approach. *Protein Engineering*, 16(8):553–560, 2003.

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

[16] L. J. McGuffin and D. T. Jones. Benchmarking secondary structure prediction for fold recognition. *PROTEINS: Structure, Function, and Genetics*, 52:166–175, 2003.

[17] 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.

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

[19] 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.

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

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

[22] B. Rost. unpublished.

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

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

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

[26] Burkhard Rost. Review: Protein secondary structure prediction continues to rise. *Journal of Structural Biology*, 2001.

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

[28] 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.

[29] A. J. Smola, P. Bartlett, B. Scholkopf, and D. Shuurmans, editors. *Probabilistic outputs for support vector machines and comparison of regularized likelihood methods*, chapter 5, pages 61–74. Advances in Large Margin Classifiers. MIT Press, 2000.

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

[31] J. J. Ward, L. J. McGuffin, B. F. Buxton, and D. T. Jones. Secondary structure prediction with support vector machines. *Bioinformatics*, 19:1650–1655, 2003.