A fast, large-scale learning method for protein sequence classification

Pavel Kuksa, Pai-Hsi Huang, Vladimir Pavlovic*
Department of Computer Science
Rutgers University
Piscataway, NJ 08854
{pkuksa;paihuang;vladimir}@cs.rutgers.edu

ABSTRACT

Motivation: Establishing structural and functional relationships between sequences in the presence of only the primary sequence information is a key task in biological sequence analysis. This ability can be critical for tasks such as making inferences of the structural class of unannotated proteins when no secondary or tertiary structure is available. Recent computational methods based on profile and mismatch neighborhood kernels have significantly improved one's ability to elucidate such relationships. However, the need for additional reduction in computational complexity and improvement in predictive accuracy hinders the widespread use of these powerful computational tools.

Results: We present a new general approach for sequence analysis based on a class of efficient string-based kernels, sparse spatial sample kernels (SSSK). The approach offers state-of-the-art accuracy for sequence classification, low computational cost, and scales well with the size of sequence databases, in both supervised and semi-supervised learning settings. Application of the proposed methods to a remote homology detection and a fold recognition problems yields performance equal to or better than existing state-of-the-art algorithms. We also demonstrate the benefit of the spatial information and multi-resolution sampling for achieving this accuracy and for discriminative sequence motif discovery. The proposed methods can be applied to very large partially-labeled databases of protein sequences because of low computational complexity and show substantial improvements in computing time over the existing methods.

Availability: Supplementary data and Matlab/C codes are available at http://seqam.rutgers.edu/spatial-kernels/

Contact: vladimir@cs.rutgers.edu

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Learning; I.5 [**Pattern Recognition**]: Applications; I.5.2. [**Pattern Recognition**]: Design Methodology

General Terms

Algorithms, Design, Measurement, Performance, Experimentation

Keywords

sequence classification, large-scale semi-supervised learning, string kernels

*corresponding author

BIOKDD '08 Las Vegas, Nevada, USA

1. INTRODUCTION

Classification of protein sequences into structural or functional classes is a fundamental problem in computational biology. With the advent of large-scale sequencing techniques, experimental elucidation of an unknown protein sequence function becomes an expensive and tedious task. Currently, there are more than 61 million DNA sequences in GenBank [3], and approximately 349,480 annotated and 5.3 million unannotated sequences in UNIPROT [2], making development of computational aids for sequence annotation a critical and timely task. In this work we focus on protein sequence classification problems using only the primary sequence information. While additional sources of information, such as the secondary or tertiary structure, may lessen the burden of establishing the homology, they may often be unavailable or difficult to acquire for new putative proteins.

Early approaches to computationally-aided homology detection, such as BLAST [1] and FASTA [22], rely on aligning the query sequence to a database of known sequences (pairwise alignment). Later methods, such as profiles [7] and profile hidden Markov models (profile HMM) [6], collect aggregate statistics from a group of sequences known to belong to the same family. Such generative approaches only make use of positive training examples, while the discriminative approaches attempt to capture the distinction between different classes by considering both positive and negative examples. In many sequence analysis tasks, the discriminative methods such as kernel-based [25] machine learning methods provide the most accurate results [4, 13, 17, 24]. Several types of kernels for protein homology detection have been proposed over the last decade. In [11], Jaakkola *et al.* proposed *SVMFisher*, derived from probabilistic models. Leslie *et al.* in [17] proposed a class of kernels that operate directly on strings and derive features from the sequence content. Both classes of kernels demonstrated improved discriminative power over methods that operate under generative settings.

Remote homology detection and fold recognition problems are typically characterized by few *positive training* sequences accompanied by a large number of negative training examples. Lack of positive training examples may lead to sub-optimal classifier performance, therefore making training set expansion necessary. However, enlarging the training set by experimentally labeling the sequences is costly leading to the need for leveraging *unlabeled data* to refine the decision boundary. The profile kernel [14] and the mismatch neighborhood kernel [26] both use large unlabeled datasets and show significant improvements over the sequence classifiers trained under the supervised setting. Nevertheless, the promising results can be offset by a significant increase in computational complexity, thus hindering use of such powerful computational tools on very large sequence data sets.

In this study, we present a general approach for efficient classification of biological sequences, based on a class of string kernels, the *sparse spatial sampling kernels (SSSK)*. The proposed method effectively models sequences under complex biological transformations such as multiple mutations, insertions, and deletions by multi-resolutional sampling. Under the SSSK, feature matching is independent of the size of the alphabet set, which ensures low computational cost. Such characteristics open the possibility of analyzing very large unlabeled datasets under the semi-supervised setting with modest computational resources. Compared to the existing string kernels, the SSSK provide a richer representation for sequences by explicitly encoding the information on spatial configuration of features within the sequences, leading to discovery of sequence motifs. The proposed methods perform better and run substantially faster than existing state-of-the-art kernel-based algorithms [14, 13, 26].

2. BACKGROUND

In this section, we briefly review previously published state-of-the-art methods for protein homology detection. We denote the alphabet set as Σ in the whole study. Given a sequence X the *spectrum-k* kernel [16] and the *mismatch(k,m)* kernel [17] induce the following $|\Sigma|^k$-dimensional representation for the sequence:

$$\Phi(X) = \left(\sum_{\alpha \in X} I(\alpha, \gamma) \right)_{\gamma \in \Sigma^k}, \quad (1)$$

where under the spectrum-k kernel, $I(\alpha, \gamma) = 1$ if $\alpha = \gamma$ and under the mismatch(k,m) kernel, $I(\alpha, \gamma) = 1$ if $\alpha \in N(\gamma, m)$, where $N(\gamma, m)$ denotes the *mutational neighborhood* induced by the k-mer γ for up to m mismatches.

Both the spectrum-k and the mismatch(k,m) kernel directly extract string features based on the observed sequence, X. On the other hand, the profile kernel, proposed by Kuang *et al.* in [13], builds a profile [7] P_X and uses a similar $|\Sigma|^k$-dimensional representation, derived from the profile:

$$\Phi^{profile(k,\sigma)}(X) = \left(\sum_{i=1 \cdots (T_{P_X} - k + 1)} I(P_X(i, \gamma) < \sigma) \right)_{\gamma \in \Sigma^k} \quad (2)$$

where $P_X(i, \gamma)$ denotes the cost of *locally* aligning the k-mer γ to the k-length segment starting at the ith position of P_X, σ a pre-defined threshold and T_{P_X} the length of the profile. Explicit inclusion of the amino acid substitution process allows both the mismatch and the profile kernels to significantly outperform the spectrum kernel and demonstrate state-of-the-art performance under both supervised and semi-supervised settings [26, 13] for the protein sequence classification tasks. However, such method of modeling substitution process induces a k-mer mutational neighborhood that is exponential in the size of the alphabet set during the matching step for kernel evaluation; for the mismatch(k,m) kernel, the size of the induced k-mer neighborhood is $k^m|\Sigma|^m$ and for the profile(k,σ) kernel, the maximum size of the mutational neighborhood is dependent on the threshold parameter σ and the shape of the profile. Increasing m or σ to model multiple mutations will incur high complexity for computing the kernel matrix hence hindering the use of such powerful tools.

The promising results of the profile kernel shown in [13] rely on the usage of a large unlabeled sequence database, such as the *non-redundant (NR)* data set, for estimation of profiles. On the other hand, for the mismatch string kernel, Weston *et al.* propose to use the *sequence neighborhood kernel* to leverage the unlabeled sequences in [26].

2.1 The sequence neighborhood kernel

The sequence neighborhood kernels take advantage of the unlabeled data using the process of neighborhood induced regularization. Let $\Phi^{orig}(X)$ be the original representation of sequence X. Also, let $N(X)$[1] denote the *sequence neighborhood* of X (a set of sequences neighboring X). Weston *et al.* proposed in [26] to re-represent X using:

$$\Phi^{new}(X) = \frac{1}{|N(X)|} \sum_{X' \in N(X)} \Phi^{orig}(X'). \quad (3)$$

Under the new representation, the kernel value between the two sequences X and Y becomes:

$$K^{nbhd}(X, Y) = \sum_{X' \in N(X), Y' \in N(Y)} \frac{K(X', Y')}{|N(X)||N(Y)|}. \quad (4)$$

Note that under such settings, all *training* and *testing* sequences will assume a new representation, whereas in a traditional semi-supervised setting, unlabeled data are used during the *training phase only*. The authors choose the mismatch representation for the sequences and show that the discriminative power of the classifiers improves significantly once information regarding the neighborhood of each sequence is available. Both the profile kernel and the mismatch neighborhood kernel show very promising results and demonstrate state-of-the-art performance in various protein sequence classification tasks. However, the exponential size of the incurred k-mer mutational neighborhood makes large-scale semi-supervised learning under the mismatch representation very computationally demanding.

3. THE SPARSE SPATIAL SAMPLE KERNELS

In this section, we present a new class of string kernels, the *sparse spatial sample kernels (SSSK)*, that effectively model complex biological transformations (such as highly diverse mutation, insertion and deletion processes) and can be efficiently computed. The SSSK family of kernels, parametrized

[1] We will discuss how to construct $N(X)$ in Section 4.1.

by three positive integers, assumes the following form:

$$K^{(t,k,d)}(X,Y) = \sum_{\substack{(a_1,d_1,\ldots,d_{t-1},a_t) \\ a_i \in \Sigma^k, 0 \le d_i < d}} \begin{array}{c} C(a_1,d_1,\cdots,a_{t-1},d_{t-1},a_t|X) \cdot \\ C(a_1,d_1,\cdots,a_{t-1},d_{t-1},a_t|Y) \end{array}, \quad (5)$$

where $C(a_1,d_1,\cdots,a_{t-1},d_{t-1},a_t|X)$ denotes the number of times we observe substring $a_1 \overset{d_1}{\leftrightarrow} a_2, \overset{d_2}{\leftrightarrow}, \cdots, \overset{d_{t-1}}{\longleftrightarrow} a_t$ (a_1 separated by d_1 characters from a_2, a_2 separated by d_2 characters from a_3, etc.) in the sequence X. This is illustrated in Figure 1.

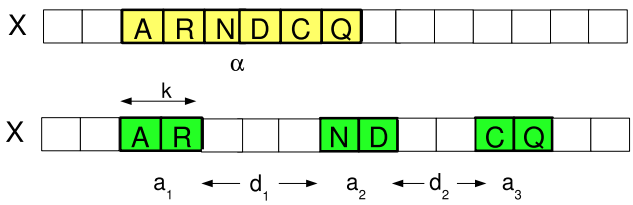


Figure 1: Contiguous k-mer feature α of a traditional spectrum/mismatch kernel (top) contrasted with the sparse spatial samples of the proposed kernel (bottom).

The new kernel implements the idea of sampling the sequences at different resolutions and comparing the resulting spectra; similar sequences will have similar spectrum at one or more resolutions. This takes into account possible mutations, as well as insertions/deletions[2]. Each sample consists of t spatially-constrained probes of size k, each of which lie no more than d positions away from its neighboring probes. In the proposed kernels, the parameter k controls the individual probe size, d controls the locality of the sample, and t controls the cardinality of the sampling neighborhood. In this work, we use short samples of size 1 (i.e., $k = 1$), and set t to 2 (i.e. features are pairs of monomers) or 3 (i.e. features are triples.)

The proposed sample string kernels not only take into account the feature counts (as in the family of spectrum kernels [16, 17] and gapped/subsequence kernels [15]), but also include spatial configuration information, *i.e.* how the features are positioned in the sequence. This is in contrast to the gapped or subsequence kernels where such information is not present. The spatial information can be critical in establishing similarity of sequences under complex transformations such as the evolutionary processes in protein sequences. The addition of the spatial information experimentally demonstrates very good performance, even with very short sequence features (*i.e. k=1*), as we will show in Section 4.

The use of short features can also lead to significantly lower computational complexity of the kernel evaluations. The dimensionality of the features induced by the proposed kernel is $|\Sigma|^t d^{t-1}$ for our choice of $k = 1$. As a result, for triple-(1,3) ($k = 1$, $t = 3$, $d = 3$) and double-(1,5) ($k = 1$,$t = 2$,$d = 5$) feature sets, the dimensionalities are $72,000$ and $2,000$, respectively, compared to $3,200,000$ for the spectrum-(k) [16], mismatch-(k,m) [17], and profile(k,σ) kernels with the common choice of $k = 5$. The low dimensionality of the feature sets ensures efficient computation. The proposed kernels can be efficiently computed using sorting and counting. To compute the kernel values, we first extract the features from the sequences and sort the extracted features in linear time using counting sort. Finally we count the number of distinct features and update the kernel matrix. For N sequences with the longest length n and u distinct features, computing the NxN kernel matrix takes linear $O(dnN + min(u, dn)N^2)$ time. Similar to the gapped kernels [15], the complexity for kernel evaluation is also independent of the size of the alphabet set. We provide a comprehensive comparison of the computational complexity and running times with other kernel methods in Section 5.

[2]We discuss how insertions and deletions are modeled in Section 5.

3.1 SSSK under Semi-supervised learning setting

The SSSK can also be extended to accommodate unlabeled data, similar to the approach presented by Weston *et al.* in [26]. Under the semi-supervised setting with unlabeled sequences, direct use of Equation 4 for computation of the refined kernel values between sequences X and Y requires $|N(X)| \times |N(Y)|$ kernel evaluations, i.e. quadratic running time in the size of the neighborhood. On the other hand, use of Equation 3 requires explicit representation of the sequences which can be problematic when the dimensionality of the feature space is high. As a result, performing such *smoothing* operation over the *mismatch kernel representation* is computationally intensive, as noted in [26, 14].

Equation 3 lends a useful insight into the complexity of the smoothing operation. For any explicit representation $\Phi(X)$, its smoothed version can be computed in time linear in the size of the neighborhood $|N(X)|$, therefore the smoothed kernel can also be evaluated in time linear in the neighborhood size. However, the smoothed representation in case of the mismatch kernel cannot be computed explicitly due to its exponential length. On the other hand, for the proposed kernels (doubles and triples) the smoothed representations can be computed explicitly, if desired.

In our experiments, we do not compute the explicit representation and instead use implicit computations over induced representations. For each neighborhood $N(X)$, a set of sequences neighboring X, we first sort the features (e.g. doubles of characters) and then obtain counts for distinct features to evaluate the kernel. This leads to a low space and time complexity for the kernel computations. The presence of mismatches, however, prevents one from applying the same approach under the mismatch representation.

4. EXPERIMENTAL RESULTS

We present experimental results for the remote homology detection under the supervised setting on the SCOP dataset in Section 4.2 and the results for large-scale semi-supervised homology detection in Section 4.3. In Section 4.4, we compare iterative (PSI-BLAST) and non-iterative (BLAST) methods for neighborhood construction. Finally, we present experimental results for remote fold recognition in Section 4.5.

4.1 Settings, parameters and performance measures

We evaluate all methods using the *Receiver Operating Characteristic* (ROC) and ROC-50 [8] scores. The ROC-50

score is the (normalized) area under the ROC curve computed for up to 50 false positives. With a small number of positive testing sequences and a large number of negative testing sequences, the ROC-50 score is typically more indicative of the prediction accuracy of a homology detection method than the ROC score.

In all experiments, we normalize kernel values $K(X,Y)$ using

$$K'(X,Y) = \frac{K(X,Y)}{\sqrt{K(X,X)K(Y,Y)}} \quad (6)$$

to remove the dependency between the kernel value and the sequence length. To perform our experiments, we use an existing SVM implementation from a standard machine learning package SPIDER[3] with the default parameters. In the semi-supervised experiments, we use kernel smoothing (Equation 4) as in [26]. For each sequence X, to construct the sequence neighborhood $N(X)$ we query the unlabeled dataset using 2 iterations of PSI-BLAST and recruit the sequences with e-values ≤ 0.05 as neighbors of X (*i.e.* $N(X) = \{X' : eValue(X,X') \leq 0.05\}$). To adhere to the true semi-supervised setting, we remove *all sequences in the unlabeled datasets that are identical to any test sequence.*

For all experiments, we compare with the state-of-the-art classifiers using the triple(1,3) ($k = 1$,$t = 3$,$d = 3$) and the double(1,5) ($k = 1$,$t = 2$,$d = 5$) feature sets.

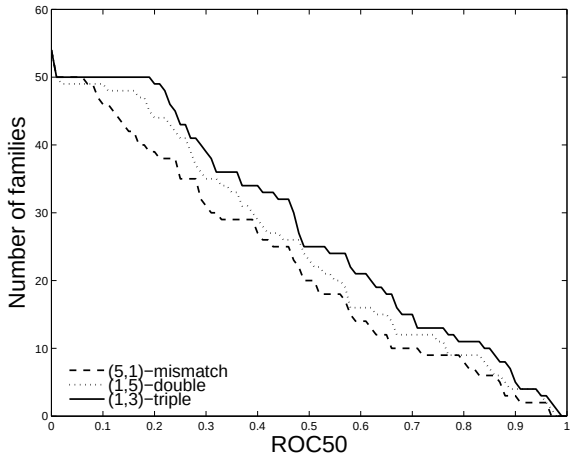
4.2 SCOP Dataset

We use the dataset published in [26] to perform our experiments. The dataset contains 54 target families from SCOP 1.59 [19] with $7,329$ isolated domains. Our experimental setup is the same as that of Jaakkola [11, 13]. In each of the 54 experiments, to simulate the remote homology problem one of the families is completely held out for testing (i.e. the classifiers are tested on the the sequences from unseen families). Different instances of this dataset have been used as a gold standard for protein remote homology detection in various studies [10, 18, 16, 17, 13].

We compare the performance of our proposed methods with previously published state-of-the-art methods [18, 17] under the supervised learning setting in Table 1. We also show the dimensionality of the induced features and the observed experimental running times, measured on a 2.8GHz CPU, for constructing the 7329x7329 kernel matrix[4]. It is clear from the table that the proposed kernels (doubles and triples) not only show significantly better performance than existing methods, but also require substantially less computational time. Also, as can be seen from the comparison with the gapped kernels, the addition of the spatial information substantially improves the classification performance. We also show the ROC-50 plot in Figure 2. In the plot, the horizontal axis corresponds to the ROC-50 scores and the vertical axis denotes the number of experiments, out of 54, with an equivalent or higher ROC-50 score. For clarity, we do not display the plot for every method. Our results clearly indicate that both double and triple kernels outperform all other methods.

[3]http://www.kyb.tuebingen.mpg.de/bs/people/spider

[4]The code used for evaluation of the competing methods has been highly optimized to perform on par or better than the published spectrum/mismatch code. We also used the code provided by the authors of the competing methods.



th

Figure 2: Comparison of the performance (ROC50) in the supervised setting. Spatial kernels (triples and doubles) outperform other supervised methods.

Table 1: Comparison of the performance on the SCOP 1.59 dataset under the supervised setting.

Method	ROC	ROC50	# dim.	Time (s)
(5, 1)-mismatch	0.8749	0.4167	3200000	938
SVM-pairwise†	0.8930	0.4340	-	-
gapped(6,2)[15]	0.8296	0.3316	400	55
gapped(7,3)	0.8540	0.3953	8000	297
(1,5) double	0.8901	0.4629	2000	54
(1,3) triple	**0.9148**	**0.5118**	72000	112

†: directly quoted from [18]

4.3 Large-Scale semi-supervised experiments

In this section, we perform the semi-supervised experiments on three unlabeled datasets:the *non-redundant* (NR) dataset, Swiss-Prot[5], and PDB[6]. Table 2 summarizes the main characteristics of the unlabeled datasets used in this study. The second column shows the size of the unlabeled datasets and the third column shows the mean, median and maximum number of neighbors per sequence recruited using PSI-BLAST with the corresponding unlabeled dataset.

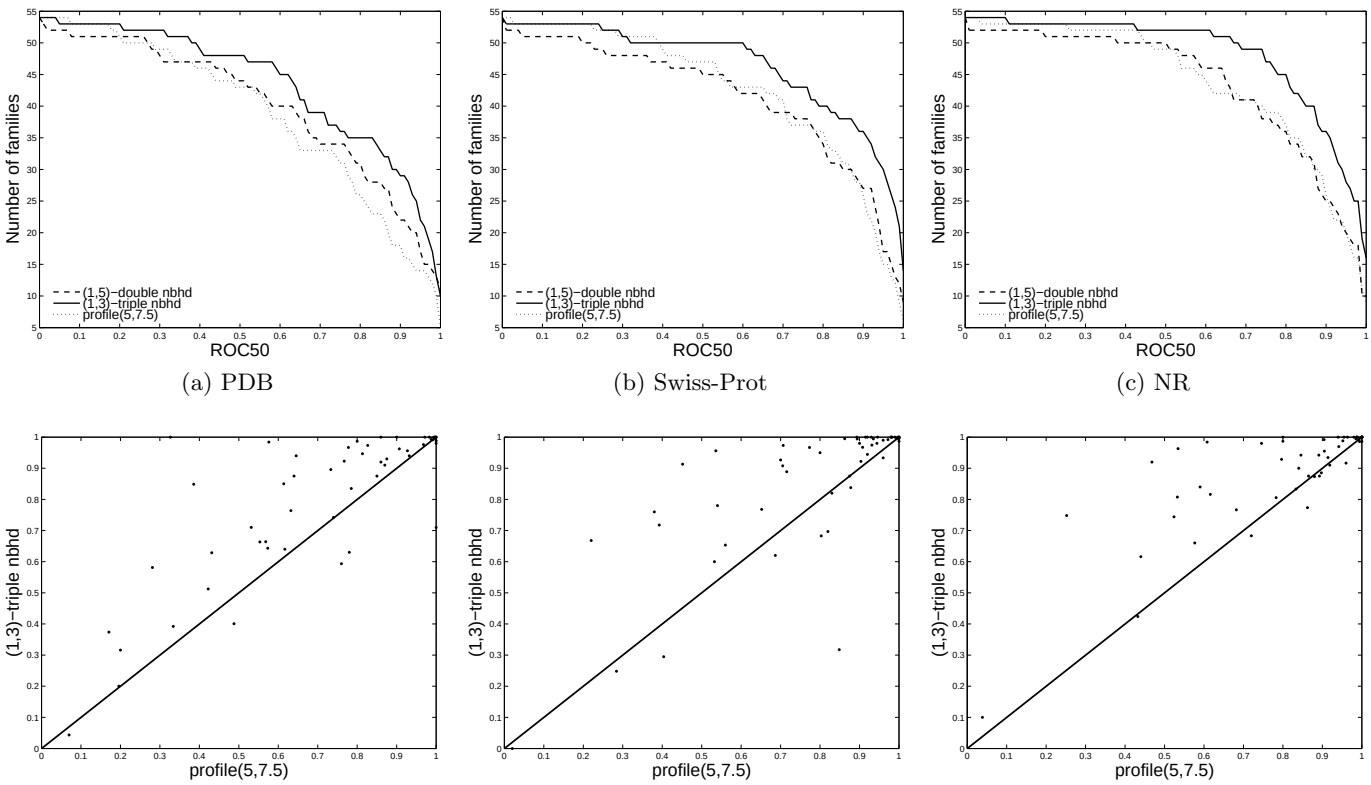
Table 2: Number of neighboring sequence recruited using PSI-BLAST with various unlabeled datasets(mean/median/max).

Dataset	# Seq	# Neighbors
Swiss-Prot	101602	56/28.5/385
PDB	116697	16/5/334
NR	534936	114/86/490

We perform all semi-supervised experiments on a 2.8GHz processor with 2GB of memory. Computation of the mismatch neighborhood kernels is computationally demanding and typically cannot be accomplished on a single machine for anything but relatively small unlabeled datasets. Therefore, the results for the mismatch neighborhood kernel can only

[5]We use the same version as the one employed in [26] for comparative analysis of performance.

[6]As of Dec. 2007.



(a) PDB (b) Swiss-Prot (c) NR

(d) PDB (e) Swiss-Prot (f) NR

Figure 3: In the upper panel, we show the ROC-50 plots of three different features using PDB, Swiss-Prot and NR databases as unlabeled datasets, respectively. In the lower panel, we show the scatter-plot of ROC-50 scores of the triple-(1,3) kernel (vertical) and the profile(5,7.5) kernel (horizontal). Any point above the diagonal line in the figures (d),(e),(f) indicates better performance for the triple-(1,3) kernel.

Table 3: Statistical significance (p-values of the Wilcoxon signed rank test) of the observed differences between pairs of methods (ROC-50 scores) on unlabeled datasets. Triple denotes the triple-(1,3) neighborhood kernel, double denotes the double-(1,5) neighborhood kernel, mismatch denotes the mismatch(5,1) neighborhood kernel, and profile denotes the profile(5,7.5) kernel.

PDB			
	double	triple	profile
double	-	1.017e-01	4.762e-02
triple	1.017e-01	-	7.666e-06
profile	4.762e-02	7.666e-06	-
Swiss-Prot			
	double	triple	profile
double	-	9.242e-05	4.992e-01
triple	9.242e-05	-	2.419e-04
profile	4.992e-01	2.419e-04	-
NR			
	double	triple	profile
double	-	8.782e-06	9.762e-01
triple	8.782e-06	-	7.017e-06
profile	9.762e-01	7.017e-06	-

be shown using the previously published summary statistics [26] on Swiss-Prot, a moderately populated sequence database. In the upper panel of Figure 3, we show the ROC-50 plots of the double-(1,5) neighborhood, triple-(1,3) neighborhood, and profile(5,7.5) kernels using PDB (first column), Swiss-Prot (second column), and NR (third column) sequence databases as the unlabeled datasets. The ROC-50 curves of the triple-(1,3) neighborhood kernel on all unlabeled datasets consistently outperform the other two kernels. Furthermore, the performance of the double-(1,5) neighborhood kernel is on par with that of the profile(5,7.5) kernel. In the lower panel, we show the scatterplots of the ROC-50 scores of the triple-(1,3) kernel and the profile(5,7.5) kernel. Any point falling above the diagonal line in the figures indicates better performance of the triple-(1,3) kernel over the profile(5,7.5) kernel. As can be seen from these plots, the triple kernel outperforms the profile kernel on all three datasets (43/37/34 wins and 4/5/10 ties on PDB, Swiss-Prot, and NR datasets, respectively).

We also show the statistical significance of the observed differences between pairs of methods on various unlabeled datasets in Table 3. All the entries in the table are the p-values of the Wilcoxon signed rank test using the ROC-50 scores. For each unlabeled dataset, we highlight the method that has the best overall performance. The triple-(1,3) kernel consistently outperforms all other kernels, with high sta-

Table 4: The overall prediction performance of all compared methods over various unlabeled datasets.

PDB	ROC	ROC50
double-(1,5) neighborhood	.9599	.7466
triple-(1,3) neighborhood	**.9717**	**.8240**
profile(5,7.5)	.9511	.7205
Swiss-Prot		
double-(1,5) neighborhood	.9582	.7701
triple-(1,3) neighborhood	**.9732**	**.8605**
profile(5,7.5)	.9709	.7914
mismatch nbhd†	.955	.810
NR		
double-(1,5) neighborhood	.9720	.8076
triple-(1,3) neighborhood	**.9861**	**.8944**
profile(5,7.5)-2 iterations	.9734	.8151
profile(5,7.5)-5 iterations‡	.984	.874
profile(5,7.5)-5 iter. with secondary structure‡	.989	.883

†:directly quoted from [26]
‡:directly quoted from [14]

tistical significance.

Finally, we show the overall prediction performance of all compared methods over various unlabeled datasets in Table 4. For each unlabeled dataset, we highlight the best ROC and ROC-50 scores; on all datasets, the triple-(1,3) neighborhood kernel achieves the best performance. Furthermore, we achieve such performance by only 2 PSI-BLAST iterations. For example, the triple-(1,3) neighborhood kernel with 2 PSI-BLAST iterations outperforms the profile(5,7.5) kernel with 5 PSI-BLAST iterations. We also note that the performance of our kernels is achieved using primary sequence information only. However, as shown in the table, the triple-(1,3) kernel still outperforms the profile(5,7.5) kernel with the added secondary structure information. Such higher order information (e.g. secondary structure), if available and desirable, can be easily included in our feature set.

4.4 Non-iterative neighborhood construction

Performing the iterative search using PSI-BLAST for neighborhood construction is computationally demanding and consumes large portion of the overall running time of the methods. In this section, we present the results obtained using BLAST search only. The use of BLAST only requires a single pass over the unlabeled sequence database and therefore requires substantially less computational time and resources compared to the iterative multi-pass PSI-BLAST search. We use the same threshold on e-value ($\leq .05$) to recruit the neighboring sequences to form the neighborhood sets $N(X)$, for each query sequence X. We compare the performance of the classifiers estimated using *iterative* (PSI-BLAST) and *non-iterative* (BLAST) sequence neighborhood construction in Table 5. First, we observe that, as the size of the unlabeled sequence database increases, the margins between the performance of the iterative and non-iterative sequence neighborhood construction procedures narrows. Second, we observe that, using only BLAST, the triple(1,3) neighborhood kernel already outperforms the profile(5,7.5) kernel, constructed with 2 PSI-BLAST iterations, and also shows comparable performance with the profile(5,7.5) kernel, constructed with 5 PSI-BLAST iterations on the non-redundant data set (Table 4). Finally, compared with the number of neighbors recruited using PSI-BLAST in Table 2, we observe a three-fold reduction when using non-iterative (BLAST) neighborhood construction procedure. Such reduction in neighborhood size enables faster training and classification as well as reduces storage requirements for the support vectors.

Table 5: Comparison of performance using iterative (PSI-BLAST) and non-iterative (BLAST) sequence neighborhood construction procedures. The performance is measured using the triple(1,3) feature set.

Data set	PSI-BLAST		BLAST		
	ROC	ROC50	ROC	ROC50	#neighbors with BLAST
PDB	0.9691	0.8240	0.9557	0.7535	6/3/95
Swiss-Prot	0.9732	0.8605	0.9640	0.8144	16/9/177
NR	0.9861	0.8944	0.9787	0.8647	40/23/232

4.5 Preliminary results for fold prediction

For the fold recognition task, we use a challenging dataset designed by Ding *et al.* [7] in [5], used as a benchmark in many studies. The data set contains sequences from 27 folds divided into two *independent* sets, such that the training and test sequences share less than 35% sequence identities and within the training set, no sequences share more than 40% sequence identities.

We compare the performance of our methods under supervised and semi-supervised settings with previously published methods on Ding and Dubchak benchmark data set in Table 6. As can be seen from the table, our spatial kernels achieve higher overall performance compared to the state-of-the-art classifiers.

5. DISCUSSION

We next compare our family of kernels with other kernel methods and discuss computational aspects of the methods. We also demonstrate how our method discovers discriminative short sequence motifs.

5.1 Complexity Comparison

We first compare computational complexity of the methods in Table 7 and show the observed running times. Running time measurements for our methods are done on a 2.8GHz CPU. For supervised experiments, we compute the full 7329x7329 kernel matrix for all methods. For the semi-supervised setting (neighborhood kernels), we report average running time on the datasets used (i.e. PDB, Swiss-Prot, and non-redundant (NR) databases.) Both the mismatch neighborhood and the profile kernels have higher complexity compared to the sample kernels due to the exponential neighborhood size. The cardinalities of the mismatch and profile neighborhoods are $O(k^m|\Sigma|^m)$, where $k \geq 5$, and $|\Sigma| = 20$, compared to a much smaller feature space size of $d^{t-1}|\Sigma|^t$ for the sample kernels, where t is 2 or 3, and d is 3 or 5, respectively. This complexity difference leads to order-of-magnitude improvements in the running times of the sample kernels over the mismatch and profile kernels. The difference is even more pronounced when kernel smoothing is used under a semi-supervised setting. The neighborhood mismatch

[7] http://ranger.uta.edu/∼chqding/bioinfo.html

Table 6: Comparison on Ding and Dubchak benchmark data set

Method	Error	Top 5 Error	Balanced Error	Top 5 Balanced Error	Recall	Top 5 Recall	Precision	Top 5 Precision	F1	Top5 F1
Supervised										
SVM(D&D)†	-	-	56.5	-	-	-	-	-	-	-
Mismatch(5,1)	51.17	22.72	53.22	28.86	46.78	71.14	**90.52**	**95.25**	61.68	81.45
Double(1,5)	44.13	23.50	46.19	23.92	53.81	76.18	61.90	79.85	57.57	77.97
Triple (1,3)	**41.51**	**18.54**	**44.99**	**21.09**	**55.01**	**78.91**	80.42	89.19	**65.33**	**83.74**
Semi-supervised (Non-redundant data set)										
Profile(5,7.5)	31.85	15.14	32.17	16.73	67.83	83.27	**89.49**	**94.9**	77.16	88.71
Double(1,5)	28.72	14.99	24.74	**11.6**	75.26	**88.4**	76.02	86.86	75.63	87.62
Triple(1,3)	**24.28**	**12.79**	**22.38**	11.79	**77.62**	88.21	84.02	91.45	**80.69**	**89.8**
Profile NR(Perceptron)‡	-	-	26.5	-	-	-	-	-	-	-

All measures are presented as percentages.
†: quoted from [5]; ‡: quoted from [21]

kernel becomes substantially more expensive to compute for large datasets as indicated in [14, 26] by Weston *et al.* .

Table 7: Complexity of computations.

Method	Time complexity	Running time (s)
Supervised setting		
Triple kernel	$O(d^2nN + d^2\|\Sigma\|^3N^2)$	112
Double kernel	$O(dnN + d\|\Sigma\|^2N^2)$	54
Mismatch	$O(k^{m+1}\|\Sigma\|^m nN + \|\Sigma\|^k N^2)$	948
Gapped kernel	$O(\binom{g}{k}knN + \|\Sigma\|^k N^2)$	176
Semi-supervised setting		
Triple kernel	$O(d^2HnN + d^2\|\Sigma\|^3N^2)$	327
Double kernel	$O(dHnN + d\|\Sigma\|^2N^2)$	67
Mismatch	$O(k^{m+1}\|\Sigma\|^m HnN + \|\Sigma^k\|N^2)$	-
Profile kernel	$O(kM_\sigma nN + \|\Sigma\|^k N^2)$	10 hours†

† the running time is quoted from [14]
Notations used in the table: N-number of sequences,
n-sequence length,
H is the sequence neighborhood size,
$|\Sigma|$ is the alphabet size
k, m are mismatch kernel parameters ($k = 5, 6$ and $m = 1, 2$ in most cases)
M_σ is the profile neighborhood size, $M_\sigma \leq |\Sigma^k|$

In previous studies [14, 26], to achieve good accuracy the number of the PSI-BLAST iterations needs to be at least 5, while our performance is achieved with only 2 iterations. We also note that the results reported in [23] are not directly comparable since an older SCOP 1.53 benchmark is used and the results are optimized on testing sequences; also, the obtained similarity measures in the corresponding study do not satisfy positive semi-definiteness condition (are not Mercer kernels).

5.2 Biological motivation

The feature sets induced by our kernels cover segments of variable length (e.g., $2-6$ residues in the case of the double-$(1,5)$ kernel). On the other hand, the mismatch and profile kernels cover segments of fixed length (e.g., 5 or 6 residues long) as illustrated in Figure 1. Sampling at different resolutions also allows one to capture similarity in the presence of more complex substitution, insertion, and deletion processes, whereas sampling at a fixed resolution, the approach used in mismatch and spectrum kernels, limits the sensitivity in the case of multiple insertions/deletions or substitutions. Increasing the parameter m (number of mismatches allowed) to accommodate the multiple substitutions, in the case of mismatch/spectrum kernels, leads to an exponential growth in the neighborhood size, and results in high computational complexity.

The proposed features also capture short-term dependencies and interactions between local sequence features by explicitly encoding the spatial information. In contrast, such information is not present in the gapped/subsequence kernels [15, 20]. In a weighted version of the subsequence kernel, where each instance (subsequence) of a particular k-mer is weighted inversely proportional to the length of the subsequence, the count for a particular k-mer is the sum of such weights. When sequences are matched under the weighted subsequence kernel, the final counts (the sum of weights) are compared and no distinction is made as to how the features were positioned in the sequences, i.e. the information on the spatial configuration of the features within the sequence is not retained.

We further illustrate differences between the proposed kernels and gapped/subsequence kernels for the case when the basic features (individual samples) of the spatial sample kernels are single characters in Equations 7 (spatial kernels) and 8 (gapped/subsequence kernels) below:

$$K(X,Y) = \sum_{\substack{(a_1,\ldots,a_t)\\ a_i\in\Sigma}} \sum_{\substack{(d_1,\ldots,d_{t-1})\\ 0\leq d_i<d}} c((a_1,d_1,\ldots,d_{t-1},a_t)|X)\cdot c((a_1,d_1,\ldots,d_{t-1},a_t)|Y) \tag{7}$$

$$K_g(X,Y) = \sum_{(a_1,a_2,\ldots,a_t)} \Big(\sum_{d_1,d_2,\ldots,d_{t-1}} c((a_1,d_1,\ldots,d_{t-1},a_t)|X)\Big)\cdot \Big(\sum_{d_1,d_2,\ldots,d_{t-1}} c((a_1,d_1,\ldots,d_{t-1},a_t)|Y)\Big) \tag{8}$$

where $c(\cdot|X)$ is the (weighted) count and $\sum_{i=1}^{t-1} d_i = g-t$ for the gapped (g,t) kernels. Note that the spatial configuration information is integrated out in the gapped/subsequence kernels, but still maintained in SSSK.

5.3 Discovering short sequence motifs with spatial information

Previous biological studies (e.g. [12]) suggested that the

spatial information such as distances between some conserved key positions can play a key role in capturing inherent characteristics of superfamilies. Our method indirectly identifies meaningful features in the protein data using the *Scorpion toxin-like* superfamily as an example. Several families in this superfamily are characterized by a number of disulphide bridges formed by conserved cysteine (C) residues. The relative positions (distances) of some neighboring key residues are also conserved as shown in Figure 4 (obtained from PROSITE [9]) for the *short-chain scorpion toxin* family. In the experiment, this family is held out for testing and all other families under the superfamily are used for training (16 positive training sequences). Among the positive sequences, 16 are selected as positive support vectors and 88 out of 1067 negative sequences are selected as negative support vectors. Under the double(1,5) representation, the pattern 'C__C' (3 residues between the two conserved cysteines residues) has the highest weight, consistent with the schematic representation shown in Figure 4. This feature is present in all positive support vectors, with the average count of 1.81 and in the negative support vectors with the average count of 0.43. The corresponding feature 'CC' under the gapped(2,4) representation has been suppressed (ranked 38 out of 400 features) due to over-representation of such feature in the negative support vectors: 39 out of 43 negative support vectors contain the feature, compared to 25 out of 88 negative support vectors with the similar feature using the double kernel. Integrating out the spatial information suppresses such feature due to its presence in the negative sequences (the average counts in the positive and negative support vectors are very close: 8.33 and 7.61). Lack of spatial information also leads to lower performance for the gapped kernel: the ROC50 score for the gapped kernel is 28.35, compared to 76.61 for the double kernel.

```
        +---------------------+
        |                     |
        |                     |
xxxxxxxCxxxxxCxxxCxxxxxxxxxxxCxxxxCxCxxx
              |   |               | |
              |   +---------------+ |
              +---------------------+
0         1         2         3
1         1         1         1
```

Figure 4: The schematic representation of the *short-chain scorpion toxins* family (obtained from PROSITE)

6. CONCLUSION

We present a computationally efficient approach for protein sequence analysis that scales well with very large sequence databases and shows state-of-the-art performance on two difficult tasks in protein sequence classification: remote homology detection and remote fold recognition. The key component of the method is the spatially-constrained sample kernel for efficient sequence comparison, which, when combined with kernel smoothing using unlabeled sequence databases, leads to rapid and accurate semi-supervised remote homology detection and fold recognition. The proposed methodology can be readily applied to other challenging problems in biological sequence analysis such as motif elucidation, ranking and clustering.

7. REFERENCES

[1] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic Local Alignment Search Tool. *Journal of Molecular Biology*, pages 403–410, 1990.

[2] A. Bairoch, R. Apweiler, C. H. Wu, W. C. Barker, B. Boeckmann, S. Ferro, E. Gasteiger, H. Huang, R. Lopez, M. Magrane, M. J. Martin, D. A. Natale, C. O'Donovan, N. Redaschi, and L.-S. L. Yeh. The Universal Protein Resource (UniProt). *Nucl. Acids Res.*, 33(suppl-1):D154–159, 2005.

[3] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler. Genbank. *Nucl. Acids Res.*, 33(suppl-1):D34–38, 2005.

[4] J. Cheng and P. Baldi. A machine learning information retrieval approach to protein fold recognition. *Bioinformatics*, 22(12):1456–1463, June 2006.

[5] C. H. Ding and I. Dubchak. Multi-class protein fold recognition using support vector machines an d neural networks . *Bioinformatics*, 17(4):349–358, 2001.

[6] S. Eddy. Profile hidden Markov models. *Bioinformatics*, 14(9):755–763, 1998.

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

[8] M. Gribskov and N. L. Robinson. Use of receiver operating characteristic (roc) analysis to evaluate sequence matching. *Computers & Chemistry*, 20(1):25–33, 1996.

[9] N. Hulo, A. Bairoch, V. Bulliard, L. Cerutti, E. De Castro, P. S. Langendijk-Genevaux, M. Pagni, and C. J. A. Sigrist. The PROSITE database. *Nucl. Acids Res.*, 34:D227–230, 2006.

[10] T. Jaakkola, M. Diekhans, and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 149–158. AAAI Press, 1999.

[11] T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. In *Journal of Computational Biology*, volume 7, pages 95–114, 2000.

[12] A. E. Kister, A. V. Finkelstein, and I. M. Gelfand. Common features in structures and sequences of sandwich-like proteins. *PNAS*, 99(22):14137–14141, 2002.

[13] 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. In *CSB '04: Proceedings of the 2004 IEEE Computational Systems Bioinformatics Conference (CSB'04)*, pages 152–160, August 2004.

[14] 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. *J Bioinform Comput Biol*, 3(3):527–550, June 2005.

[15] C. Leslie and R. Kuang. Fast string kernels using inexact matching for protein sequences. *J. Mach. Learn. Res.*, 5:1435–1455, 2004.

[16] C. S. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for svm protein classification. In *Pacific Symposium on Biocomputing*, pages 566–575, 2002.

[17] C. S. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for svm protein classification. In *NIPS*, pages 1417–1424, 2002.

[18] L. Liao and W. S. Noble. Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In *RECOMB*, pages 225–232, 2002.

[19] L. Lo Conte, B. Ailey, T. Hubbard, S. Brenner, A. Murzin, and C. Chothia. SCOP: a structural classification of proteins database. *Nucleic Acids Res.*, 28:257–259, 2000.

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

[21] I. Melvin, E. Ie, J. Weston, W. S. Noble, and C. Leslie. Multi-class protein classification using adaptive codes. *J. Mach. Learn. Res.*, 8:1557–1581, 2007.

[22] W. Pearson and D. Lipman. Improved tools for biological sequence comparison. *PNAS*, 85:2444–2448, 1988.

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

[24] S. Sonnenburg, G. Rätsch, and B. Schölkopf. Large scale genomic sequence svm classifiers. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 848–855, New York, NY, USA, 2005.

[25] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[26] J. Weston, C. Leslie, E. Ie, D. Zhou, A. Elisseeff, and W. S. Noble. Semi-supervised protein classification using cluster kernels. *Bioinformatics*, 21(15):3241–3247, 2005.