Rutgers Computer Science Technical Report RU-DCS-TR634
May 2008

Fast and accurate semi-supervised protein homology detection with large uncurated sequence databases

by

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

ABSTRACT

Establishing structural and functional relationship between sequences in the presence of only the primary sequence information is a key task in biological sequence analysis. This ability is critical for tasks such as inferring the superfamily membership of unannotated proteins (remote homology detection) when no secondary or tertiary structure is available. Recent methods such as profile kernels and mismatch neighborhood kernels have shown promising results by leveraging unlabeled data and explicit modeling mutations using *mutational neighborhood*. However, the size of such neighborhood exhibit exponential dependency on the cardinality of the alphabet set which incurs expensive cost for kernel evaluation and hence hinders the use of such powerful tools. Moreover, another missing component in previous studies for large-scale semi-supervised protein homology detection is a systematic and biologically motivated approach for leveraging the unlabeled data set.

In this study, we propose a systematic and biologically motivated approach for extracting relevant information from unlabeled sequence databases. We also propose a method to remove the bias caused by overly represented sequences which are commonly seen in the unlabeled sequence databases. Combining these approaches with a class of kernels (*sparse spatial sampling kernels, SSSK*) that effectively model mutation, insertion, and deletion, we achieve fast and accurate semi-supervised protein homology detection on three large unlabeled databases. The resulting classifiers based on our proposed methods significantly outperform previously published state-of-the-art methods in performance accuracy and exhibit order-of-magnitude differences in experimental running time.

1 Introduction

Protein homology detection 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 [4], and approximately 349,480 annotated and 5.3 million unannotated sequences in UNIPROT [3], making development of computational aids for sequence annotation a critical and timely task. In this work we focus on the problem of predicting protein remote homology 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 and even when present, such information is only available on a very small group of protein sequences and absent on larger uncurated sequence databases.

Early approaches to computationally-aided homology detection, for example BLAST [2] and FASTA [20], rely on aligning the query sequence to a database of known sequences (pairwise alignment). Later methods, such as profiles [9] and profile hidden Markov models (profile HMM) [8], 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 [22] machine learning methods provide the most accurate results [7, 12, 17, 21]. 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 problem is typically characterized by few *positive training* sequences accompanied by a large number of negative training examples. Experimentally labeling the sequences is costly leading to the need to leverage *unlabeled data* to refine the decision boundary. The profile kernel [13] and the mismatch neighborhood kernel [23] both use unlabeled data sets and show significant improvements over the sequence classifiers trained under the supervised setting. We believe the major contributions for their great success come from first, leveraging unlabeled data and second, the use of *mutational neighborhood* to model amino acid substitution process. However, kernel evaluation based on the induced mutational neighborhood incurs exponential complexity in the size of the alphabet set hence hindering the use of such powerful tools.

Another missing component in previous studies for large-scale semi-supervised protein homology detection is a systematic and biologically motivated approach for leveraging the unlabeled data set. In this study, we address both issues. First, we employ a class of previously established kernels, the Sparse Spatial Sample Kernels (SSSK) [15]. This class of biologically motivated kernels model mutation, insertion and deletion effectively and induce low-dimensional feature space; moreover, the computational complexity of kernel evaluation based on feature matching is independent of the size of the alphabet set and such key characteristics opens the door for rapid

large-scale semi-supervised learning. Second, we propose a biologically meaningful way of extracting relevant information from the unlabeled database for semi-supervised learning. Third, we also propose a method to remove the bias caused by overly represented or duplicated unlabeled sequences which are commonly seen in uncurated sequence databases. Our experimental results show that the combination of these approaches yields state-of-the-art performance that are significantly better than previously published methods and also exhibit order-of-magnitude differences in experimental running time.

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:

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

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 $\alpha \in N(\gamma, m)$ and $N(\gamma, m)$ denotes the set of k-mer *mutational neighborhood* induced by the k-mer γ for up to m mismatches or substitutions.

Both spectrum-k and mismatch(k,m) kernel directly extract string features based on the observed sequence. Under the mismatch representation, all substitutions are treated as equally likely, which may not be deemed practical due to the physical and chemical properties of amino acids. The profile kernel [12] takes such constraints into consideration: given a sequence X and its corresponding profile [9] P_X, Kuang et al. [12, 13] define the $|\Sigma|^k$-dimensional profile(k,σ) representation of X as:

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

where σ is a pre-defined threshold, T_{P_X} denotes the length of the profile and $P_X(i, \gamma)$ the cost of *locally* aligning the k-mer γ to the k-length segment starting at the i^{th} position of P_X. Explicit inclusion of the amino acid substitution process allows both the mismatch and profile kernels to significantly outperform the spectrum kernel and demonstrate state-of-the-art performance under both supervised and semi-supervised settings [23, 12]. 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 size of the neighborhood is bounded below by $k^m|\Sigma|^m$, above by $|\Sigma|^k$, and is dependent on the threshold parameter σ. Increasing m or σ to incorporate more mismatches will incur higher complexity for computing the kernel matrix hence hindering the use of such powerful tools.

Finally, to construct the sequence profiles required for computation of the profile kernel, we need to leverage the unlabeled sequences to avoid overfitting of the profile. For the mismatch string kernel, Weston et al. propose to use the *sequence neighborhood kernel* to leverage the unlabeled sequences in [23].

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 $N(X)$ denote the *sequence neighborhood* of X [1]. Weston et al. proposed in [23] to re-represent X using:

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

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

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

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 improve significantly once information regarding the neighborhood of each sequence is available. However, the exponential size of the incurred k-mer mutational neighborhood makes large-scale semi-supervised learning under the mismatch representation very computationally demanding and cannot be performed using only moderate computational resources.

3 Proposed methods

In this section, we first discuss the *sparse spatial sample kernels* (SSSK) for protein homology detection. Such kernels effectively model the insertion, deletion and substitution processes and the complexity of the string matching step for kernel evaluation is independent of the size of the alphabet set. The kernels show very promising results under the supervised setting and also under the semi-supervised setting with a small unlabeled sequence data set [15]. Next, we discuss a systematic and biologically motivated way to extract only relevant information from the unlabeled database. Finally we also discuss how to remove the bias caused by duplicated or overly represented sequences which are commonly found in large uncurated sequence databases. The combination of the proposed methods enables fast and accurate semi-supervised learning for protein homology detection.

[1] We will discuss how to define $N(X)$ in later sections.

3.1 The sparse spatial sample kernel

The class of *sparse spatial sample kernels*, proposed by Kuksa et al. [15] have the following form:

$$K(X,Y|k,t,d) = \sum_{\substack{(a_1,d_1,...,d_{t-1},a_t) \\ a_i \in \Sigma^k, 0 \le d_i < d}} C(a_1,d_1,a_2,d_2,...,d_{t-1},a_t|X)C(a_1,d_1,a_2,d_2,...,d_{t-1},a_t|Y), \quad (5)$$

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. The 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 and deletions. Each sample consists of t spatially-constrained probes of size k, each of which lie less than d positions away from its neighboring probes. 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 of monomers). The spatial sample kernels, unlike the family of spectrum kernels [16, 17], not only take into account the feature counts, but also include spatial configuration information, *i.e.* how the features are positioned in the sequences. 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.

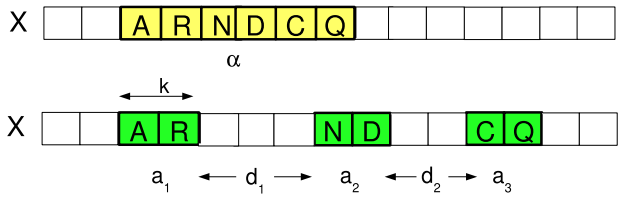


Figure 1: Contiguous k-mer feature α of a traditional spectrum/mismatch kernel (top) contrasted with the sparse spatial samples of the proposed kernel (bottom).

The use of short features can also lead to significantly lower computational complexity of the kernel evaluations. The dimensionality of the features induced by the spatial sample kernels is $|\Sigma|^t d^{t-1}$ for the 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) [12] kernels with the common choice of $k = 5$. In Figure 2 we show the differences between the spatial (double(1,5)) and the spectrum (mismatch(5,1)) features on two slightly diverged sequences, S and S'. In the mismatch features, each symbol 'X' represent an arbitrary symbol in the alphabet set, Σ. As a result, each feature basis corresponds to $|\Sigma|$ features. Such way of modeling substitution

induces a k-mer mutational neighborhood in $O(k^m|\Sigma|^m)$ size. In contrast, the spatial features sample the sequences at different resolutions and therefore performing string matching does not require neighborhood expansion; matching on a position with substitution is achieved by extending the current spectrum. Such way of modeling substitution opens the door for a matching algorithm with low complexity *i.e.* independent of the size of the alphabet, which in turns opens the door for fast large-scale semi-supervised learning, as we will see in Section 4. In the figure, we represent all common features between the original and the mutated strings with bold fonts and red (light) color.

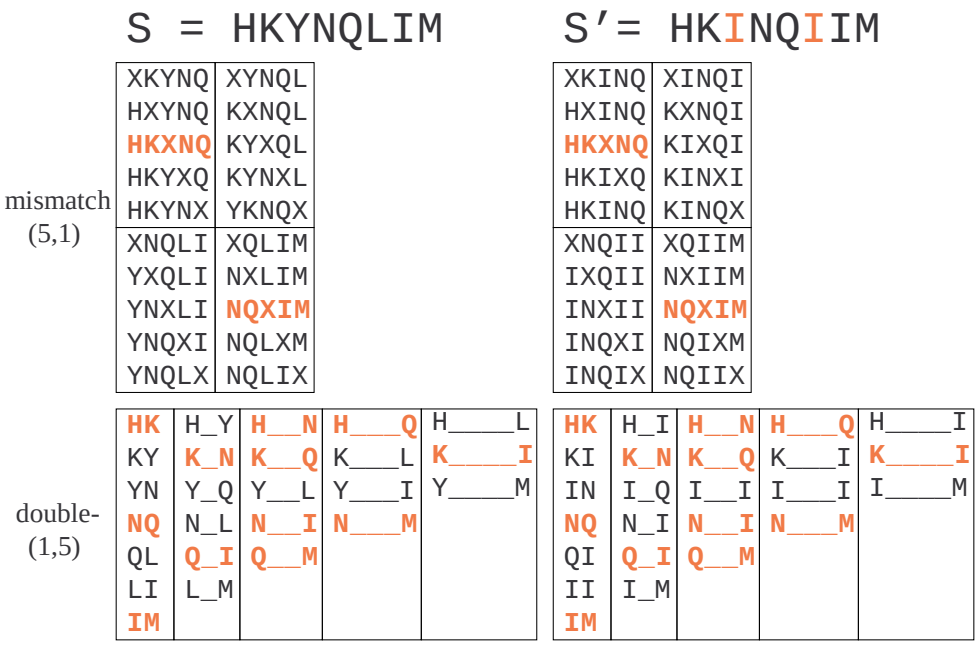


Figure 2: Differences in handling substitutions by the mismatch and spatial features. We represent all common features between the original and the mutated strings, S and S', with bold fonts and red (light) color. Each symbol 'X' under the mismatch representation represent an arbitrary symbol in the alphabet set Σ. As a result, each feature basis corresponds to $|\Sigma|$ features.

To compute the kernel values under the supervised setting, we first extract the features and sort the extracted features in linear time using counting sort. Finally we count the number of distinct features and for each observed feature, we update the kernel matrix. For N sequences with the longest length n and u distinct features, computing the N-by-N kernel matrix takes linear $O(dnN + min(u, dn)N^2)$ time.

Under the semi-supervised setting, on the one hand, 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 sequence 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(5,1) representation is computationally intensive for both methods due to first, the exponential length of the induced k-mer mutational neighborhood and second, the quadratic

running time induced by equation 4.

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. As mentioned before, the smoothed representation under the mismatch features cannot be efficiently computed because of the exponential size of the induced k-mer neighborhood; however, for the double and triple feature sets 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 set $N(X)$, we first sort the features and then obtain counts for distinct features to evaluate the kernel. The low-dimensional feature space and efficient feature matching induced by the kernels ensure low complexity for kernel evaluation. Kuksa et al. provides a more detailed description of algorithm for spatial kernel evaluation under both supervised and semi-supervised settings in [14].

3.2 Extracting relevant information from the unlabeled sequence database

Remote homology detection problem is typically characterized by *few positive sequences* accompanied by a large number of *negative examples*. Experimentally labeling the sequences is costly, leading to the need to leverage *unlabeled data* to refine the decision boundary. In [23], Weston et al. leverage the unlabeled sequences to construct a *sequence neighborhood kernel* under the mismatch representation to refine the decision boundary. However, in most sequence databases, we have multi-domain protein sequences in abundance and thus, such multi-domain sequences can be similar to several unrelated single-domain sequence, as noted in [23]. Direct use of such long sequences may falsely establish similarities among unrelated sequences. Under semi-supervised learning setting, our goal is to recruit *neighbors*, or *homologues* of training and testing sequences and use these intermediate neighbors to establish similarity between the remotely homologous proteins, which bear little to no similarity on the primary sequence level. As a result, the quality of the intermediate neighboring sequences is crucial for inferring labels of remote homologues. Sequences that are too long will contribute excessive features, while sequences that are too short often have missing features and hence induce very sparse representation, which in turn bias the averaged neighborhood representation. As a result, the performance of the classifiers will be compromised with direct use of these sequences. Weston et al. in [23] proposed to only capture neighboring sequences with maximal length of 250 as a remedy. However, such practice may not offer a direct and meaningful biological interpretation and may discard valuable information. In this study, we propose to extract only *statistically significant sequence regions*, reported by PSI-BLAST, from the unlabeled neighboring sequences. We summarize all competing methods in below:

- *unfiltered*: all neighboring sequences are recruited. This is to show how much excessive or missing features in neighboring sequences that are too long or too short compromise the performance of the classifiers.

- *extracting the most significant region*: for each recruited neighboring sequence, we extract only the most *statistically significant sequence region* reported by PSI-BLAST; such sub-

sequence is more likely to be biologically relevant to the query sequence.

- *filter out sequences that are too long or too short*: for each query sequence X, we remove any neighboring sequences Y if $T_Y > 2T_X$ or $T_Y < \frac{T_X}{2}$, where T_X is the length of sequence X. This method will alleviate the effect of the excessive and missing features induced by the unfiltered method.

- *maximal length of 250*: this is the method proposed by Weston et al. in their study.

To recruit neighbors of a sequence X, we query the unlabeled database using PSI-BLAST [1] with two iterations. We recruit all sequences with e-values less than or equal to 0.05 as the neighboring sequences of X. To obtain only relevant information from a neighboring sequence, we extract from the unlabeled neighboring sequence the most significant region (lowest e-value) reported by PSI-BLAST. We illustrate the procedure in Figure 3. In the figure, given the query sequence, PSI-BLAST reports sequences (hits) containing substrings that exhibit statistically significant similarity with the query sequence. For each reported hit with e-value less than or equal to 0.05, we extract the most significant region and recruit the extracted sub-sequence to the neighboring set of the query sequence.

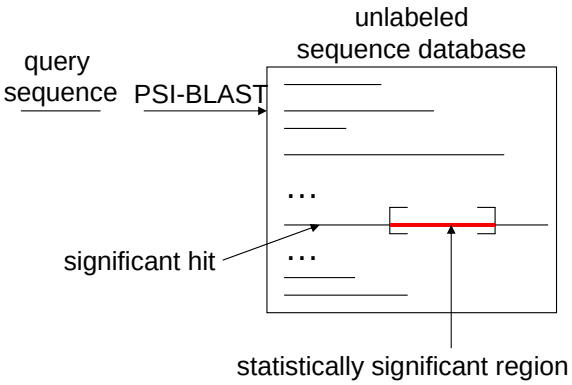


Figure 3: Extracting only statistically significant regions (red/light color, bold line) from the significant hit reported by PSI-BLAST

3.3 Clustering the neighboring sets

The smoothing operation in Equation 3 is susceptible to overly represented neighbors in the unlabeled data set since if we we append many replicated copies of a neighbor to the set, the computed average will be biased towards such sequence. In large uncurated sequence databases, duplicated sequences are common. For example, some sequences in Swiss-Prot have the so-called *secondary accession numbers*. Such sequences can be easily identified and removed. However, there are two other types of duplication that are harder to find: sequences that are nearly identical and sequences that contain substrings that have high sequence similarity and are significant hits to the query sequence. Existence of such examples will bias the estimate of the averaged representation, hence

compromising the performance of the classifiers. Pre-processing the data is necessary to remove such bias. In this study we propose to cluster the neighboring sets as a remedy. Conducting clustering analysis typically incurs quadratic complexity in the number of sequences to be clustered. As a result, though clustering the union of all neighbor sets is more desirable, to minimize the experimental running time we propose to cluster each reported neighbor set *one at a time*; for example, the union of all neighbor sets (e-value less than or equal to 0.05) induced by the NR unlabeled database is $129,646$, while the average size of the neighbor sets is 115 (reported in later sections). Clustering each reported neighbor set individually will lead to tremendous saving in experimental running time.

We use the program *CDHit* [18] for clustering analysis. The program employs a heuristic (incremental clustering algorithm) to avoid all-by-all comparisons. First, the sequences are sorted in decreasing length with the longest one representing the clustering center. Next, each remaining sequences is compared to each existing clustering center and will be assigned to the first cluster in which the similarity between the cluster representative and the query sequence exceeds a predefined threshold. If no such cluster exists, the sequence will form a new cluster. In this study we perform clustering at 70% sequence identity level.

4 Experiments

We present experimental results for protein remote homology detection under the semi-supervised setting on the SCOP 1.59 [19] data set, published in [23]. The data set contains 54 target families with $7,329$ isolated domains. Only $2,862$ domains out of $7,329$ are labeled, which allows to perform experiments in both supervised (labeled sequences only) and semi-supervised (labeled and unlabeled sequences) settings. Different instances of this data set have been used as a gold standard for protein remote homology detection in various studies.

In [15], Kuksa et al. show that the class of spatial sample kernels achieve the state-of-the-art performance under the supervised setting and semi-supervised setting, where in the semi-supervised setting, the unlabeled data set comes from the SCOP 1.59 [19] sequence database itself. Note that sequences in the SCOP database are represented in single domains and therefore, use of such unlabeled data set does not raise any concern over extracting relevant information from a multi-domain sequence. In this study, we use three larger unlabeled sequence databases, some of which contains abundant multi-domain protein sequences as well as duplicated or overly represented sequences. The three databases are PDB [5][2] (116,697 sequences), Swiss-Prot [6][3] (101,602 sequences), and the *non-redundant* (NR) sequence database (534,936 sequences). To adhere to the true semi-supervised setting, *all sequences in the unlabeled data sets that are identical to any test sequences are removed*.

We evaluate all methods using the *Receiver Operating Characteristic* (ROC) and ROC50 [10] scores. The ROC50 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

[2] as of Dec. 2007

[3] We use the same version as the one used in [23] for comparative analysis of performance

		double(1,5)			triple(1,3)		
	#neighbors	ROC	ROC50	p-value	ROC	ROC50	p-value
PDB							
unfiltered	14/5/311	.9333	.7324	.3498	.9393	.7444	3.46e-04
region	11/5/311	.9533	.7352	-	.9666	.8074	-
no tails	11/3/286	.9255	.6926	.0197	.9433	.7456	3.50e-03
by length	11/2/300	.9254	.6848	6.02e-02	.9418	.7127	4.53e-05
Swiss-Prot							
unfiltered	56/28/385	.9145	.6360	6.55e-04	.9245	.6908	2.46e-04
region	56/28/385	.9593	.7635	-	.9752	.8556	-
no tails	27/4/385	.9160	.6318	2.12e-04	.9361	.6938	1.55e-06
by length	21/3/385	.9070	.5652	2.03e-05	.9300	.6514	7.33e-07
NR							
unfiltered	115/86/490	.9319	.6758	1.40e-03	.9419	.7328	1.07e-05
region	115/86/490	.9715	.7932	-	.9824	.8861	-
no tails	55/13/399	.9463	.6775	4.40e-03	.9575	.7438	9.47e-06
by length	38/10/426	.9275	.6656	7.32e-04	.9513	.7401	2.66e-06

*p-value: signed-rank test on ROC50 scores against region
#neighbors: mean/median/max

Table 1: The overall prediction performance of all compared methods over various unlabeled data sets.

testing sequences, the ROC50 score is typically more indicative of the prediction accuracy of a homology detection method than the ROC score.

In all experiments, all kernel values $K(X, Y)$ are normalized using

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

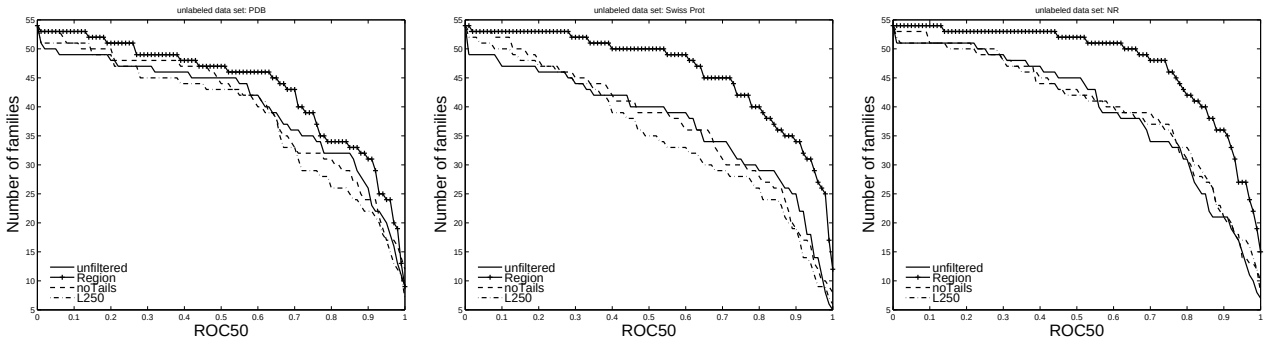
to remove the dependency between the kernel value and the sequence length. We use the sequence neighborhood kernel in Equation 4, as in [23], under the spatial sample representation. To perform our experiments, we use an existing SVM implementation from a standard machine learning package SPIDER[4] with default parameters.

4.1 Experimental results without clustering

In Table 1, we show the performance in ROC and ROC50 scores for the four competing methods on the double(1,5) and triple(1,3) feature sets using 3 different unlabeled sequence data sets. We denote the method of filtering out sequences that exhibit a two-fold difference in length with the query sequence as *no tails* and the method of filtering out sequences whose length is greater than

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

250 as *by length*. In all but one case, extracting only relevant regions from the unlabeled sequence leads to significant improvement in the ROC and ROC50 scores compared to the unfiltered method. In the second column, we note the number of recruited neighbors (mean, median, and max). We also calculate the p-values of each competing method **against** the *region* method using Wilcoxon signed-rank test. In all cases except one, we observe statistically significant improvement in classification performance. Extracting significant regions for neighborhood smoothing improves the ROC and ROC50 scores on average by 0.0373 and 0.1048, respectively, when compared to the *unfiltered* method. We show the ROC50 plots of the four competing methods using the triple(1,3) feature set in Figure 4. In the figures, the horizontal axis corresponds to an ROC50 score and the vertical axis denotes the number of experiments, out of 54, with the corresponding or higher ROC50 score. In all cases, we observe that the ROC50 curves for region extraction show strong dominance over all other competing methods. Based on the table and figures, we also observe that filtering out neighboring sequences based on the length degrades the performance of the classifiers on the PDB (Figure 4(a)) and Swiss-Prot (Figure 4(b)) unlabeled sequence databases while in the case of using the NR data set (Figure 4(a)), the classifier shows slight improvement. Although filtering out sequences based on the length removes the unnecessary and noisy features from irrelevant regions within the sequences, at the same time, longer unlabeled sequences that carry critical information for inferring the class labels of the test sequences are also discarded. In a larger unlabeled data set (NR), such problem is alleviated since larger databases are more likely to contain short sequences carrying such critical information.



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

Figure 4: The ROC50 plots of four competing methods using the triple-(1,3) feature set with PDB, Swiss-Prot and NR databases as unlabeled data sets, respectively. The ROC50 curves of the method that only extracts relevant regions from the neighboring sequences consistently show strong dominance over all competing methods.

4.2 Experimental results with clustering

In Table 2, we present the performance in ROC and ROC50 scores for the four competing methods on the double(1,5) and triple(1,3) feature sets using 3 different unlabeled data sets. All smoothed

		double(1,5)			triple(1,3)		
	#neighbors	ROC	ROC50	p-value	ROC	ROC50	p-value
PDB							
unfiltered	11/4/116	.9369	.7142	6.74e-02	.9439	.7585	4.70e-03
region	11/4/120	.9599	.7466	-	.9717	.8240	-
no tails	9/3/102	.9291	.6902	4.8e-03	.9490	.7545	2.30e-03
by length	7/2/104	.9229	.6589	1.10e-03	.9490	.7211	2.66e-05
Swiss-Prot							
unfiltered	30/17/223	.9526	.6397	3.76e-04	.9464	.7474	1.50e-03
region	27/15/210	.9582	.7701	-	.9732	.8605	-
no tails	15/3/192	.9214	.6446	1.95e-04	.9395	.7160	2.30e-06
by length	10/2/107	.9100	.5841	1.21e-05	.9348	.6817	7.33e-07
NR							
unfiltered	77/55/344	.9403	.6874	5.62e-04	.9556	.7566	2.20e-05
region	67/47/339	.9734	.8048	-	.9861	.8944	-
no tails	37/10/310	.9452	.6815	2.90e-04	.9602	.7486	2.06e-07
by length	24/8/263	.9313	.6686	1.00e-03	.9528	.7595	2.56e-07

*p-value: signed-rank test on ROC50 scores against region
#neighbors: mean/median/max

Table 2: The overall prediction performance of all compared methods over various unlabeled data sets **with clustering** the neighbor sets. All neighbor sets are clustered on a 70% sequence identity level and representatives of each cluster are chosen to form a reduced neighbor set.

representations are induced by the reduced neighbor sets. In contrast to Table 1, extracting relevant regions from neighboring sequences and performing clustering on the neighbor sets significantly improve performance on **all** unlabeled data sets. With clustering, extracting regions improves the ROC and ROC50 scores on average by 0.0245 and 0.0994, respectively, when compared to the *unfiltered* method. We again observe performance degradation when filtering out neighboring sequences based on their lengths. In the second column, we show the number of neighbors (mean, median, and maximum) after clustering. In most cases, we observe a 2-3 fold reduction in the number of neighbors contrasting to the neighborhood size reported in Table 1. We note that the reduction in the neighborhood size is critical for faster training and classification.

Finally we show the experimental running time in Table 3 under various settings, performed on a 3.6GHz CPU, based on the $2,862$ labeled sequences in the SCOP 1.59 data set. The average reduction in running time for kernel evaluation is 10.32% for the double(1,5) kernel and 10.66% for the triple(1,3) kernel. Clustering takes very little CPU time; for example, clustering the neighbor sets induced by the NR sequence database on all $2,862$ labeled sequences takes 126.24 seconds in total on the region-based method. We want to note the large fold change in running time by adding one spatial sample ($t = 3$ for triple in contrast to $t = 2$ in double). Increasing the number of spatial samples by 1 implies multiplying the complexity for string matching by d, the maximum number of distance allowed between two samples. After clustering, the reduction in experimental

	double(1,5)		triple(1,3)	
	without clustering	with clustering	without clustering	with clustering
PDB				
unfiltered	10.70	10.19	170.45	161.01
region	10.22	9.98	99.57	95.05
no tails	10.17	9.94	103.39	104.49
by length	9.97	9.85	73.85	73.36
Swiss-Prot				
unfiltered	16.14	12.84	802.27	719.62
region	12.74	11.17	289.03	240.15
no tails	11.61	10.52	186.06	160.84
by length	10.64	10.01	107.62	94.28
NR				
unfiltered	23.26	18.60	1451.52	1345.89
region	15.69	13.54	630.95	531.07
no tails	13.69	12.32	383.80	348.52
by length	11.66	10.74	245.23	213.02

Table 3: The experimental running time (seconds) for constructing the (2862-by-2862) kernel matrices on each unlabeled data set under different settings. The experiments are performed on a 3.6GHz CPU.

running time will be significant for other tasks that require more spatial samples (increasing t), larger distance between each spatial samples (increasing d), larger sequence database (increasing N) or longer sequences (increasing n) since the complexity for feature matching exhibit multiplicative dependency on these parameters ; performing feature matching incurs $O(d^{t-1}HNn)$ and $O(k^{m+1}|\Sigma|^m HnN)$ complexity for spatial and mismatch(k,m) kernels, respectively, where H denotes the size of the sequence neighborhood. Performing clustering reduces the neighborhood size by two fold on average, which in turn implies less computational resources for storage: under the discriminative kernel learning setting, we need to save the support vectors along with their corresponding neighbor sets. The savings in experimental running time for kernel evaluation will be even more pronounced if the previously described parameters are increased simultaneously. For a detailed analysis of computational complexity, please refer to [14].

4.3 Comparison with other state-of-the-art methods

We compare the performance of our proposed method with previously published state-of-the-art methods over various unlabeled sequence databases and present the overall prediction performance of all compared methods in Table 4. For spatial kernels, all reported scores are based on extracting the most significant region and performing clustering on the neighbor sets. We perform all experiments on a $3.6GHz$ machine with 2GB of memory. Computation of the mismatch neighborhood kernels is computationally demanding and typically cannot be accomplished on a single

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 [23] ;‡:directly quoted from [13]

Table 4: The overall prediction performance of all compared methods over various unlabeled data sets. For spatial kernels, all reported scores are based on extracting the most significant region and performing clustering on the neighbor sets.

machine for anything but relatively small unlabeled data sets. Therefore, the results for the mismatch neighborhood kernel can only be shown using the previously published summary statistics [23] on Swiss-prot, a moderately populated sequence database. For each unlabeled data set, we highlight the best ROC and ROC50 scores; on all unlabeled data sets, 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. Moreover, the triple(1,3) neighborhood kernel with 2 PSI-BLAST iterations on the PDB unlabeled data set already outperforms the profile(5,7.5) kernel with 2 PSI-BLAST iterations on the NR unlabeled data set. 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 added secondary structure information. Such higher order information (*e.g.* secondary structure), if available and desirable, can be easily included in the feature set. In this study, we do not pursue such direction.

We also show the statistical significance of the observed differences between pairs of methods on various unlabeled data sets in Table 5. All the entries in the table are the p-values of the Wilcoxon signed-rank test using the ROC50 scores. For each unlabeled data set, we highlight the method that has the best overall performance. The triple(1,3) kernel consistently outperform all other kernels, with high statistical significance.

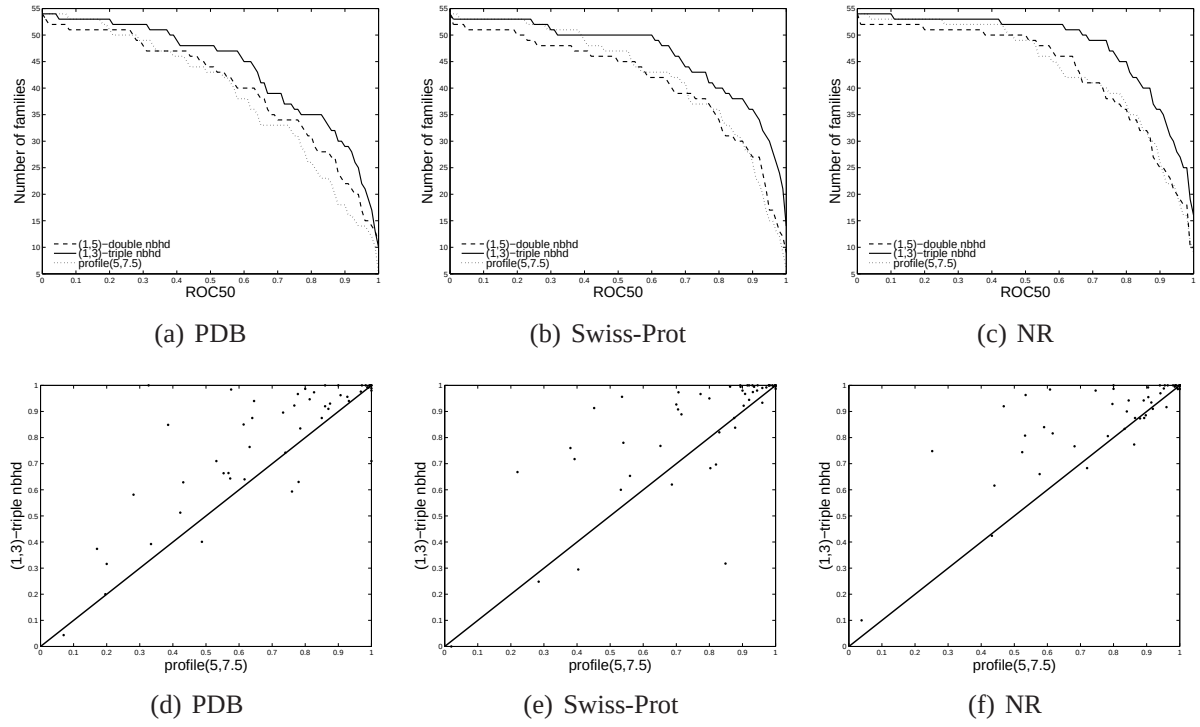
Next, in the upper panel of Figure 5, we show the ROC50 plots of the double(1,5) neighborhood, triple(1,3) neighborhood and profile(5,7.5) kernels using PDB (first column), Swiss-Prot

SCOP 1.59				
	mismatch	profile	double	triple
mismatch	-	2.245e-03	1.804e-02	3.570e-06
profile	2.245e-03	-	2.874e-01	9.615e-09
double	1.804e-02	2.874e-01	-	6.712e-06
triple	3.570e-06	9.615e-09	6.712e-06	-
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	-	

Table 5: Statistical significance (p-values of the Wilcoxon signed-rank test) of the observed differences between pairs of methods (ROC50 scores) on unlabeled data sets. 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.

(second column) and NR (third column) sequence databases as the unlabeled data sets. The ROC50 curves of the triple(1,3) neighborhood kernel on all unlabeled data sets consistently show strong dominance over those of 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 scatter plots of the ROC50 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 data sets (43/37/34 wins and 4/5/10 ties, out of 54 experiments, on PDB, Swiss-Prot, and NR data sets, respectively).

Finally, in Table 6, we show the experimental running time for constructing the kernel matrix, based on all available sequences in the SCOP 1.59 data set. The size of the kernel matrix is 7329-by-7329. For the semi-supervised setting (neighborhood kernels), we report average running time on the data sets used (*i.e.* PDB, Swiss-Prot, and non-redundant (NR) sequence databases). As mentioned in previous sections, both mismatch and profile kernels require higher complexity



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

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

Figure 5: In the upper panel, we show the ROC50 plots of three different features using PDB, Swiss-Prot and NR databases as unlabeled data sets, respectively. In the lower panel, we show the scatter-plot of ROC50 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.

to perform feature matching due to the exponential size of the mutational neighborhood, which in turns depend on the size of the alphabet set, whereas the complexity of performing feature matching for the spatial features is independent of the alphabet set size. This complexity difference leads to order-of-magnitude improvements in the running times of the spatial 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 kernel becomes substantially more expensive to compute for large unlabeled data sets as indicated in [13, 23] by the authors.

5 Discussion

We first illustrate the benefit of extracting only statistically significant regions from the neighboring sequences from a machine learning perspective and then we discuss the biological motivation of the spatial feature sets. The spatial features allow alphabet-free matching and model substitution, insertion and deletion effectively. The combination of both methods leads to fast and accurate semi-supervised protein remote homology detection.

Method	Running time (s)
supervised methods	
Triple(1,3) kernel	112
Double(1,5) kernel	54
Mismatch(5,1) kernel	948
semi-supervised methods	
Triple(1,3) neighborhood kernel	327
Double(1,5) neighborhood kernel	67
Mismatch(5,1) neighborhood kernel	-
Profile(5,7.5) kernel	10 hours[†]

[†]: the running time reported in [13]

Table 6: Experimental running time of all methods based on all sequences in the SCOP 1.59 data set. The size of the kernel is 7329-by-7329. For triple and double kernels, under the semi-supervised setting, the reported running time are based on extracting relevant regions and performing clustering on neighboring sets.

5.1 Motivation for extracting relevant regions

To illustrate the benefit of extracting only statistically significant regions from an unlabeled sequence, consider the example in Figure 6. In the figure, colors indicate membership: yellow (shaded) corresponds to the positive class and green (pattern) corresponds to the negative class. Sequences that demonstrate statistically significant similarity are more likely to be evolutionarily related and therefore to belong to the same superfamily. The goal is to infer membership of the test (unshaded) sequences via the unlabeled sequence (in the middle). In the figure, arcs indicate (possibly weak) similarity induced by shared features, denoted by the black boxes, and absence of arcs indicates no similarity. As can be seen from the figure, the positive training and test sequences share no features and therefore have no similarity; however, the unlabeled sequence shares some features with both sequences in the reported region, which are very likely to be biologically relevant to both positive training and test sequences and therefore establishes the similarity between them. On the other hand, if the whole unlabeled sequence is recruited as a neighbor without discarding irrelevant regions, the similarity between the positive training and negative testing sequences will be incorrectly established, hence compromising the performance of the classifiers.

5.2 Biological Motivation of the spatial feature sets

Compared to mismatch/profile kernels, the feature sets induced by our kernels cover segments of variable length (*e.g.* 2-6 and 3-7 residues in the case of the double(1,5) kernel and the triple(1,3) kernels, respectively), whereas 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 to capture similarity in the presence of more complex substitution, insertion and deletion processes, while 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. We illustrate the benefit

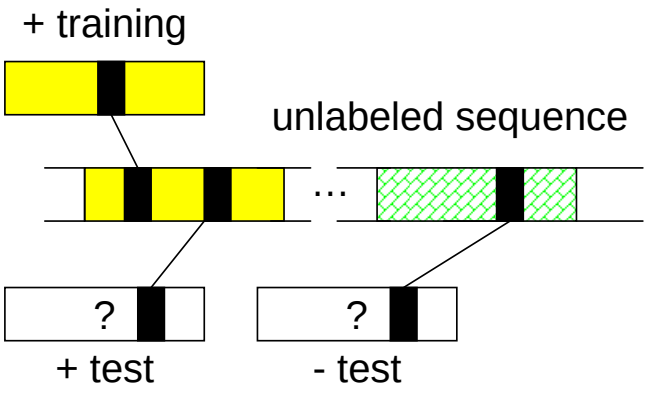


Figure 6: The importance of only extracting relevant region from neighboring sequences (in the middle): in the figure, the colors indicate the membership: yellow (shaded) indicates membership of the positive class and green (pattern) indicates membership of the negative class. The goal is to infer the label of the test (unshaded) sequences via the intermediate neighboring sequences. The arcs in the figure indicate (possibly weak) similarity and absence of arcs indicates no similarity. The black boxes in the sequence correspond to the shared features.

of multi-resolution sampling in Figure 7. In the figure, we show six slightly diverged sequences with the presence of both mutation and insertion. We also show the double(1,5) and mismatch(5,1) kernel matrices. We observe that the spatial kernel still captures substantial amount of similarities whereas the mismatch kernel, which performs fixed-resolution sampling captures little similarities among the related sequences. Both images are shown on the same scales. Increasing the parameter m (number of mismatches allowed) to accommodate multiple substitutions, in the case of mismatch kernels, leads to an exponential growth in the size of the k-mer mutational neighborhood, and results in high computational complexity. On the other hand, increasing the threshold σ in the profile kernel also incurs an exponential growth in the size of mutational neighborhood since in a highly diverged region the profile may be flat.

6 Conclusion

In this study, we propose a systematic and biologically motivated approach for extracting relevant information from unlabeled sequence database under the semi-supervised learning setting. We also propose to perform clustering on each neighbor sets to remove the bias caused by duplicated or overly represented neighboring sequences which are commonly found in large uncurated sequence databases. Combing these approaches with the sparse spatial sample kernels we achieve fast and accurate semi-supervised protein homology detection on three large unlabeled sequence databases. The spatial kernels induce low-dimensional feature space, effectively model mutation, insertion, and deletion with multi-resolution sampling and incur low computational complexity for kernel evaluation; its running time on string matching is independent of the size of the alphabet set, making rapid kernel evaluation possible on large sequence databases. The resulting classifiers based on our proposed methods significantly outperform previously published state-of-the-art methods

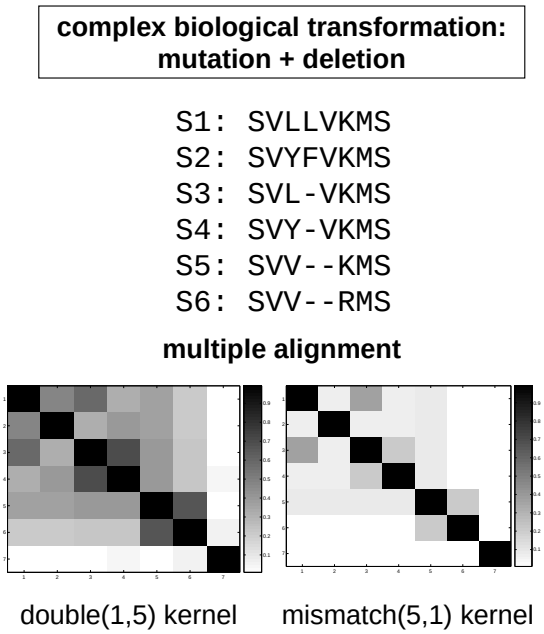


Figure 7: The benefit of multi-resolution sampling: in the presence of both mutations and insertions, the spatial kernel still captures substantial amount of similarities in such moderately conserved region; on the other hand, the mismatch kernel, which performs fixed-resolution sampling captures little similarity among related sequences.

in performance accuracy and exhibit order-of-magnitude differences in experimental running time.

References

[1] S. Altschul et al. Gapped Blast and PSI-Blast: A new generation of protein database search programs. *NAR*, 25:3389–3402, 1997. 7

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

[3] 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. 1

[4] Dennis A. Benson, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and David L. Wheeler. Genbank. *Nucl. Acids Res.*, 33(suppl-1):D34–38, 2005. 1

[5] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28:235–242, 2000. 8

[6] B. Boeckmann, A. Bairoch, R. Apweiler, M.C. Blatter, A. Estreicher, E. Gasteiger, M.J. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Res*, 31:365–370, 2003. 8

[7] Jianlin Cheng and Pierre Baldi. A machine learning information retrieval approach to protein fold recognition. *Bioinformatics*, 22(12):1456–1463, June 2006. 1

[8] SR Eddy. Profile hidden Markov models. *Bioinformatics*, 14(9):755–763, 1998. 1

[9] M. Gribskov, A.D. McLachlan, and D. Eisenberg. Profile analysis: detection of distantly related proteins. *PNAS*, 84:4355–4358, 1987. 1, 2

[10] M. Gribskov and N. Robinson. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching, 1996. 8

[11] Tommi Jaakkola, Mark Diekhans, and David Haussler. A discriminative framework for detecting remote protein homologies. In *Journal of Computational Biology*, volume 7, pages 95–114, 2000. 1

[12] Rui Kuang, Eugene Ie, Ke Wang, Kai Wang, Mahira Siddiqi, Yoav Freund, and Christina 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. 1, 2, 4

[13] Rui Kuang, Eugene Ie, Ke Wang, Kai Wang, Mahira Siddiqi, Yoav Freund, and Christina Leslie. Profile-based string kernels for remote homology detection and motif extraction. *J Bioinform Comput Biol*, 3(3):527–550, June 2005. 1, 2, 13, 15, 16

[14] Pavel Kuksa, Pai-Hsi Huang, and Vladimir Pavlovic. Kemel methods and algorithms for general sequence analysis. Technical Report RU-DCS-TR630, Department of Computer Sciences, Rutgers University, 2008. 6, 12

[15] Pavel Kuksa, Pai-Hsi Huang, and Vladimir Pavlovic. Spatially-constrained sample kernel for sequence classification. In *The Learning Workshop (SNOWBIRD)*, 2008. 1, 3, 4, 8

[16] Christina S. Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: A string kernel for svm protein classification. In *Pacific Symposium on Biocomputing*, pages 566–575, 2002. 2, 4

[17] Christina S. Leslie, Eleazar Eskin, Jason Weston, and William S. Noble. Mismatch string kernels for svm protein classification. In *NIPS*, pages 1417–1424, 2002. 1, 2, 4

[18] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006. 8

[19] L. Lo Conte, B. Ailey, T.J. Hubbard, S.E. Brenner, A.G. Murzin, and C. Chothia. SCOP: a structural classification of proteins database. *Nucleic Acids Res.*, 28:257–259, 2000. 8

[20] W.R. Pearson and D.J. Lipman. Improved tools for biological sequence comparison. *PNAS*, 85:2444–2448, 1988. 1

[21] Sören Sonnenburg, Gunnar Rätsch, and Bernhard 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. 1

[22] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998. 1

[23] Jason Weston, Christina Leslie, Eugene Ie, Dengyong Zhou, Andre Elisseeff, and William Stafford Noble. Semi-supervised protein classification using cluster kernels. *Bioinformatics*, 21(15):3241–3247, 2005. 1, 2, 3, 6, 8, 9, 13, 15