Remote Homology Detection of Protein Sequences

Matteo Comin   Davide Verzotto*

August 24, 2010

Abstract

The automatic classification of protein sequences into families is of great help for the functional prediction and annotation of new proteins. In the paper we present a method called Irredundant Class that address the remote homology detection problem. The best performing methods that solve this problem are string kernels, that compute a similarity function between pairs of proteins based on their subsequence composition. We provide evidence that almost all string kernels are based on patterns that are not independent, and therefore the associated similarity scores are obtained using a set of redundant features, overestimating the similarity between the proteins. To specifically address this issue, we introduce the class of irredundant common patterns. Loosely speaking the set of irredundant common patterns is the smallest class of independent patterns that can describe all common patterns in a pair of sequences. We present a classification method based on the statistics of these patterns, named Irredundant Class. Results on benchmark data show that the Irredundant Class outperforms most of the string kernels previously proposed, and it achieves results as good as the current state-of-the-art method Local Alignment, but using the same pairwise information only once.

*Department of Information Engineering, University of Padova, Italy. {ciompin,verzotto}@dei.unipd.it

Dagstuhl Seminar Proceedings 10231
Structure Discovery in Biology: Motifs, Networks & Phylogenies
http://drops.dagstuhl.de/opus/volltexte/2010/2741

1 Introduction

The increasing availability of biological sequences, from protein sequences to entire genomes, poses the need for the development of automatic classification tools. In the paper we address the classification of proteins based on their primary structure, that is the amino acid sequence. This classification problem can also be treated as a binary string classification problem.

A number of methods have been proposed for the protein sequence classification. Historically this problem has been studied, for quite some times, in the field of text documents classification. Unfortunately most of the proposed approaches, developed for a different kind of strings, fail when applied to biological sequences. The main reasons of this failure are the different nature of biological sequences, particularly rich of regularities known as patterns that are difficult to digest for a general purpose application.

The main distinction is between generative methods against discriminative methods. The former class includes methods such as protein family profiles (Gribskov et al., 1987), hidden Markov models (HMMs) (Krogh et al., 1994; Baldi et al., 1994; Karplus et al., 1998), and PSI-BLAST (Altschul et al., 1997). These methods tend to derive a model for a set of proteins and then check whether a candidate protein fits the model or not. Unlike generative methods, discriminative methods such as (Jaakkola et al., 1999; Liao and Noble, 2003; Leslie et al., 2002, 2004; Saigo et al., 2004; Hou et al., 2003; Rangwala and Karypis, 2005; Kuang et al., 2005) focus on finding which sequences can describe a set of proteins despite of another set.

Recent results (Liao and Noble, 2003; Leslie et al., 2004) suggest that the best-performing methods are discriminative string kernels. These methods use kernel functions based on common patterns of pairs of protein sequences to train a support vector machine (SVM) (Vapnik, 1998; Cristianini and Shawe-Taylor, 2000). The string kernel extracts information from sequences and computes either a feature vector for each sequence or directly a kernel matrix with scores between pairs of sequences.

The first string kernel, called Fisher's kernel (Jaakkola et al., 1999), uses an HMM to provide the necessary means of converting proteins into fixed-length vectors. The vector summarizes how different the given sequence is from a typical member of the given protein family. In the Pairwise kernel (Liao and Noble, 2003) the feature vector corresponding to a protein sequence is formed by all E-values, given by the Smith-Waterman algorithm (Smith and Waterman, 1981), computed on the sequence analyzed and each of the training sequences of a particular experiment.

The Spectrum and the Mismatch kernels (Leslie et al., 2004, 2002) use as protein features the set of all possible substrings of amino acids of fixed length (k-mers). If two sequences contain many of the same k-length contiguous subsequences, their inner product under the k-Spectrum kernel will be large. Equivalently, the Mismatch kernel computes a large inner product between two sequences if these sequences contain many k-length contiguous subsequences that differ by at most e mismatches.

The Local Alignment method (Saigo et al., 2004) mimics the behavior of the Smith-Waterman algorithm to build a family of valid kernels. Following the work of (Haussler, 1999) they defined a kernel that mimics the detection of all local alignments between two sequences by convolving simpler kernels. In the Word Correlation Matrices method (Lingner and Meinicke, 2008), the kernel is defined by average pairwise word similarity between two sequences. The consequent analysis of discriminative words allows also for the identification of characteristic regions in biological sequences.

Other methods, such as the I-Sites (Hou et al., 2003), encode into feature vectors information related both to three-dimensional structure properties and sequence similarities of proteins. Or, like the eMOTIF-database method (Ben-Hur and Brutlag, 2003), they define a kernel function in terms of occurrences of sequence motifs previously stored in databases, and tipically extracted using popular algorithms on reference sequences. In particular,the Profile-based Mismatch methods (Kuang et al., 2005) use probabilistic profiles, such as those produced by the PSI-BLAST algorithm, to define kernels based on position-dependent mutation neighborhoods of k-mers with mismatches (in a similar way to the original Mismatch kernel). While the Profile-based Direct methods (Rangwala and Karypis, 2005) build kernel functions combining sequence profiles obtained with different approaches for determining the similarity between pairs of protein sequences. Note that the latter methods make an extensive use of hyperparameters, increasing the risk of overfitting.

We selected for comparison with our method some of the algorithms presented above, in particular those with state-of-the-art performance on the classification of proteins and which seem somewhat more reliable: Fisher, Pairwise, Spectrum, Mismatch, Local Alignment (version “eig”), and Word Correlation Matrices. In general, all pattern-based methods operate two distinct steps: first extract and process common patterns from pair of sequences, then use this set of patterns as features to build an automatic classification tool based on SVMs; and so does the method proposed here. As we will show in the next sections, almost all string kernels are based on patterns that are not independent, namely patterns with occurrences that are related each other. Any score built using a set of related patterns is in practice based on redundant features, and therefore it can potentially overestimate the similarity score.

In the paper we want to stress the idea that if the learning process has to deal with a set of *redundant* features, this might mislead the classification. The goal is somehow similar to the feature selection problem, but, in the case of pattern-based methods for classification contexts, the class of *irredundant* common patterns is specifically designed to address the issue of a repeated information. Our conjecture is that a set of *irredundant* common patterns, and consequently a set of independent features, can facilitate the automatic learning and classification of sequences.

2 Methods

2.1 The Irredundant Class

The method is based on the extraction and filtering of patterns that are common to two sequences, using the mathematical notion of *irredundancy*. This notion was first studied by (Gao et al., 1997; Parida, 1998; Parida et al., 2000) for the case of repeated patterns on a single sequence, and thereafter by (Apostolico and Parida, 2004; Pisanti et al., 2005; Apostolico and Tagliacollo, 2008). In (Comin and Verzotto, 2010) we extended the notion of irredundancy to the case of two sequences, to avoid the overcount of common patterns that "cover" the same region of a sequence. Indeed, one can easily show that there are lots of protein sequences that share an unusually large number of common patterns without conveying extra information about the input (see Table 4). To keep the paper self-contained, here we summarize the basic facts already proved in (Comin and Verzotto, 2010).

Let s_1 and s_2 be two sequences, respectively, of m and n characters over an alphabet Σ. A character from Σ is called a *solid* character, while a *don't care* character '$\cdot$' *equals* and represents any character on Σ. Let $s_1[i,j]$ be the subsequence given by the $j-i+1$ consecutive characters of s_1 starting from position i. We call a *suffix* of s_1 any subsequence of the type $s_1[i,m]$, with $1 \leq i \leq m$.

A *pattern* p is a string over $\Sigma\ (\Sigma \cup \{\cdot\})^*\ \Sigma$, thus having at least two solid characters: the first and the last character. A location $[i, j]$ of s_1 is an *occurrence* of p if $s_1[i, j] = p$. A *common pattern* is a pattern that occurs in both s_1 and s_2.

Definition 1. *(Coverage) For characters σ_1 and σ_2 we write that $\sigma_1 \preceq \sigma_2$ if σ_1 is a don't care or $\sigma_1 = \sigma_2$. Given two different patterns p_1 and p_2, respectively, of q and $r \geq q$ characters, we say that the occurrence $[i, q+i-1]$ of p_1 on s_1 is covered by p_2 if (1) $p_1 \preceq p_2[j, q+j-1]$ for such an offset $j \geq 0$, considering the characters corresponding to the same positions, and (2) $s_1[i-j, r+i-j-1] = p_2$.*

In other words, property (1) of Definition 1 says that p_2 has to extend p_1 in composition (that is, p_2 would be more specific than p_1) and/or in length, and (2) indicates that p_2 occurs in the same region of p_1. We give an example of coverage in Figure 1. In this case the occurrences of the common pattern `ABA` at $[7, 9]$ on s_1 and $[1, 3]$ on s_2 are covered, respectively, by the common patterns `ABAA·C` and `A···ABA`. Using Definition 1 we can now define an *irredundant common pattern* as a common pattern with an occurrence that cannot be deduced from the other common patterns without knowing the input sequences:

Definition 2. *(Irredundant Common Pattern) A common pattern p is* irredundant *if at least an occurrence of p on s_1 or s_2 is not covered by other common patterns.*

Clearly, a pattern that is not irredundant is called *redundant.*

Definition 3. *(Meet) The* meet *of two subsequences of s_1 and s_2 is obtained by matching the characters corresponding to the same positions, inserting a don't care in case of mismatch, and, thereafter, deleting all leading and trailing don't cares. In this case every meet is also a common pattern if it has at least two solid characters.*

As a result of Lemma 1 presented in (Comin and Verzotto, 2010) we have that:

Theorem 1. *Every irredundant common pattern of s_1 and s_2 is the meet of a sequence with a suffix of the other one.*

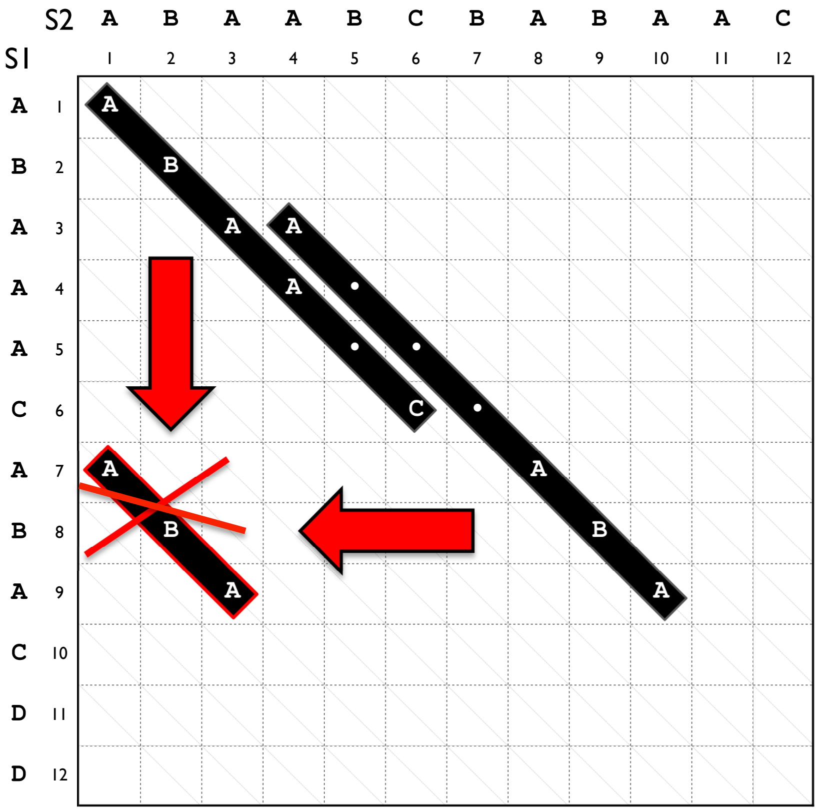

Figure 1: Example of pattern occurrence coverage on the sequences s_1 = ABAAACABACDD and s_2 = ABAABCBABAAC of length 12. The occurrences of the meet ABA are covered, respectively, by the meets ABAA·C and A· · ·ABA.

Proof. In Lemma 1 of (Comin and Verzotto, 2010) we showed that a common pattern must appear at least in the meet of a sequence with a suffix of the other one to be irredundant, and this proves the theorem. □

For example, in Figure 1 the common pattern `ABA` is the meet of s_2 with the suffix $s_1[7, 12]$ of s_1. However `ABA` turns out to be in any case a redundant pattern, and thus we need a more sophisticated algorithm to discover the whole class of irredundant common patterns, or *Irredundant Class*. In this regard, we refer the reader to the Algorithm presented in Section Methods of (Comin and Verzotto, 2010).

As an immediate consequence of Theorem 1, we obtain that the number of irredundant common patterns of s_1 and s_2 is linear in the number of characters (or length) of the sequences:

Theorem 2. *The number of irredundant common patterns of* s_1 *and* s_2 *is at most* $m + n - 3$.

Proof. As of Theorem 1, the maximum number of meets between a sequence with the suffixes of the other one is limited in number by the length of s_1 and s_2. These common patterns, necessarily of length greater than 1, are at most $m + n - 3$. □

Finally, we note that if s_1 and s_2 are identical, there is only one irredundant common pattern: the sequence itself. Moreover, we can extract the Irredundant Class in time $O(z^2 \log z \log |\Sigma|)$, where $z = m + n$, making use of the FFT in the step of searching for occurrences of the $m + n - 3$ meets described above.[1]

2.2 Scoring the Irredundant Class

Once acquired the Irredundant Class $\mathcal{I}_{s_1,s_2}$ of s_1 and s_2, we score this set of patterns using their frequencies and some properties of the amino acid composition. Here we report the general form of the scoring function:

$$Score(s_1, s_2) = \ln \left(\sum_{p \in \mathcal{I}_{s_1,s_2}} \frac{F_p}{E[F_p]} \right),$$

[1] This step, that is the most expensive in our procedure presented in (Comin and Verzotto, 2010), is described in detail in (Fischer and Paterson, 1974).

where F_p is defined as the number of occurrences of the pattern p in s_1 and s_2, and $E[F_p]$ is the expected value of F_p. The value $E[F_p]$ is computed combining the probability of each character a of p, extracted from the BLOSUM62 substitution matrix (Henikoff and Henikoff, 1992), with the length of the sequences:

$$E[F_p] = (m + n - 2(k-1)) \times \prod_{a \in p} P(a),$$

where k is the length of p.

Unfortunately the Irredundant Class, the name as we will call the general method in the rest of the paper, seems to lack the positive-definiteness property, and therefore it must be treated as an indefinite kernel. In particular, following the work of (Eichhorn, 2007) for indefinite kernels applied to SVMs, we have that the Irredundant Class is in the case of weak non-positivity, and thus we need only to force the SVM optimizer to stop after a maximum number of iterations.

3 Why Resort to Irredundant Common Patterns?

The exhaustive detection of homologies in protein families and superfamilies leads to prohibitive computational methods, but on the other hand a low-complexity detection, for example using k-mers, would only consider a low-significant set of possible homologies, often overcounting the same information. These issues can be solved using an alternative method based on irredundant common patterns. Moreover, the automatic filter given by the notion of "non-redundancy", or *irredundancy*, ensures us to select just those informative patterns that characterize the homologies of a pair of sequences.

We selected five algorithms of pairwise string similarity detection, used within the state-of-the-art protein classification methods, for the comparison with our method: Spectrum, Mismatch, Word Correlation (the core of Word Correlation Matrices), Local Alignment (namely the distance function given by all local alignments), and Smith-Waterman (the core of Pairwise).

3.1 Description of State-of-the-Art Pairwise String Algorithms

In the following we briefly explain the meaning of the selected algorithms on a pair of sequences s_1 and s_2, and then in the next subsection we estimate the redundancy, or *information overcount*, for each algorithm. Note that every algorithm computes a specific score for each extracted pattern, and then a global score is assigned to a pair of sequences using these pattern-specific scores.

In *Spectrum (k)* we count the number of occurrences for all the shared substrings (or consecutive subsequences) of length k on Σ in s_1 and s_2. In *Mismatch (k, e)* we count the number of occurrences for all the shared strings of length k on Σ in s_1 and s_2, and then we add each value to the other k-mers of which the *meet* has at most e don't cares. In *Word Correlation (k)* we compute a similarity score between all the k-mers of s_1 versus all the k-mers of s_2, and this is like to consider all the meets on $\Sigma \cup \{.\}$ of k-length substrings of s_1 with k-length substrings of s_2. In *Local Alignment* we consider the global alignments between all pairs of substrings of s_1 and s_2 (given a scheme of scores for matches, substitutions, insertions, and deletions), that are all possible local alignments. Similar to Local Alignment, in *Smith-Waterman* we take the best global alignment between all pairs of substrings of s_1 and s_2. In *Irredundant Class* we consider all possible shared patterns on $\Sigma \cup \{.\}$ in s_1 and s_2, and then we avoid the contribution to be "overcounted" using the mathematical notion of *irredundancy* and selecting up to $m+n-3$ patterns among the meets between all suffixes of s_1 and s_2.

3.2 Information Overcount: From a Theoretical Perspective

For each method we can now identify two characteristic phases: *(1) pattern extraction* and *(2) pattern processing.* We can think the output of phase (2) as a vector of pattern-specific scores, where each column represents just the score related to a single pattern.

For example, for Mismatch (1) is the process of finding k-mers in the two sequences s_1 and s_2, while (2) is the multiplication of the respective number of occurrences of these k-mers in s_1 and s_2, where the number of occurrences of each k-mer is the number of times it appears with up to e errors. In this case the output of phase (2) will be the vector of values resulting from the multiplications, and each column will represent a single k-mer. For Spectrum (1) is the same as for Mismatch, but (2) is only the multiplication of the shared k-mer occurrences without any other preliminary process, and thus without error parameters. For Word Correlation, in (1) we individually find the k-mers of s_1 and s_2, and in (2) we compute a similarity score between all the possible pairs of these k-mers (one k-mer of s_1 versus one of s_2). For Local Alignment (1) is represented by the extraction of all the substrings of the two sequences, while (2) is the global alignment of all the possible pairs of these substrings. For Smith-Waterman (1) and (2) are the same as for Local Alignment, but in (2) we have also a *max* operation between all the computed values on the possible global alignments. For Irredundant Class (1) is the extraction of all suffixes of s_1 and s_2, while (2) is the set of operations in which we compute the meets between a sequence and a suffix of the other one, and then we filter out the redundant ones.

Here we consider the information overcount as the number of outputs from phase (2) obtained taking into account the same pair of characters of s_1 and s_2 more than once:

Definition 4. *The* information overcount *is the number of vector components output of phase (2) in which the same pair of characters, one from s_1 and one from s_2, contributes more than once.*

Each output from phase (2), or component of the resulting vector, is intended as the score obtained comparing some pairs of single characters. For instance, after phase (2) of Spectrum we have a vector of values where each column represents the multiplication of the numbers of occurrences of a specific k-mer found in s_1 and s_2. These k-mers are overlapped in the two sequences by construction, and each component of the resulting vector represents at least k positions of each sequence. Therefore we use an information about the comparison of a shared position between s_1 and s_2 in more than one k-mer, and thus we store this information in more than one column of the final vector, resulting in an information overcount. We call the model that considers the information overcount as the *Information Overcount Model.*

Table 1 shows a comparison of the algorithms based on the Information Overcount Model, where we fixed a priori $m \geq n$. The computation of these values is quite simple. For Irredundant Class, the meets between a sequence with all suffixes of the other sequence can be computed in a $m \times n$ grid, where each item represents the comparison of two different characters and each meet is a different diagonal of items from the top-left to the bottom-right part of the grid. Therefore we have *no* information overcount. For Smith-Waterman we have again *no* information overcount, because after phase (2) we consider only the best local alignment pattern that is comparing different characters in each position. For Spectrum we could have at most $n-k+1$ shared k-mers between s_1 and s_2. Thus, in this case, in s_2 we have at most a coverage of k times (given by k-mers) for each of the $n-2(k-1)$ central positions, and at most a coverage of $2\sum_{i=1}^{k-1} i$ times for all leading $k-1$ and all trailing $k-1$ characters. Given that a coverage without repetitions considers each shared position only once, we have a total information overcount of $(k-1)(n-2(k-1)+2\sum_{i=1}^{k-2} i) = (k-1)(n-k)$, hence $O(kn)$. For Word Correlation we have the same maximum value of information overcount as for Spectrum, because in the evaluation of pairwise similarity between the k-mers of s_1 and s_2 we consider the comparison of a k-mer with another k-mer only once. Thus the output repetitions are based on the overlaps between the shared k-mers. In Mismatch we have the information overcount of Spectrum plus an additional redundancy due to the spreading of the number of occurrences of a k-mer to the other k-mers within e errors. The last part of the summation can be estimated in $k(n-k+1)\sum_{i=1}^{e}\binom{k}{i}(|\Sigma|-1)^i$, where the factor k is the number of positions covered by each k-mer, $n-k+1$ is the maximum number of shared k-mers, and the last factor is the number of k-mers within e errors from a fixed k-mer. Then, the resulting information overcount would be $(k-1)(n-k)+k(n-k+1)\sum_{i=1}^{e}\binom{k}{i}(|\Sigma|-1)^i = O(k^{e+1}|\Sigma|^e n)$. Finally, in Local Alignment we compute a global alignment for each pair of substrings of s_1 and s_2. Thus the information overcount will be based only on the overlaps of these substrings in s_2, resulting in $n(n+1)(n+2)/6$ repeated outputs, that is $O(n^3)$.

In Table 2 we present a comparison of pairwise computational complexity for the six algorithms described above, to give an idea of trade-off between information overcount (see Table 1), computational complexity, and effectiveness in the classification of protein sequences (see Table 3). These values were taken from the original papers.

4 Experimental Results

We assess the effectiveness of the Irredundant Class method on the classification of protein families into superfamilies. This problem refers to the detection of sequence homologies in evolutionarily related proteins with low-

Table 1: Comparison of Information Overcount for State-of-the-Art Methods[a]

Algorithm	Information Overcount
Irredundant Class	*none*
Smith-Waterman	*none*
Spectrum (k)	$O(kn)$
Word Correlation (k)	$O(kn)$
Mismatch (k, e)	$O(k^{e+1}\|\Sigma\|^e n)$
Local Alignment	$O(n^3)$

[a] Comparison of algorithms using the Information Overcount Model, with $m \geq n$. Rows are listed from the best to the worst.

Table 2: Comparison of Pairwise Complexity for State-of-the-Art Methods[b]

Algorithm	Pairwise Complexity
Spectrum (k)	$O(kz)$
Word Correlation (k)	$O(k^2\|\Sigma\|^2 z)$
Mismatch (k, e)	$O(k^{e+1}\|\Sigma\|^e z)$
Local Alignment	$O(z^2)$
Smith-Waterman	$O(z^2)$
Irredundant Class	$O(z^2 \log z \log \|\Sigma\|)$

[b] Comparison of algorithms based on their pairwise computational complexity, where $z = m + n$. Rows are listed from the best to the worst.

sequence similarity, and is called *remote homology detection*.

Tests are based on the dataset described in (Liao and Noble, 2003),[2] which uses the Structural Classification Of Proteins (SCOP)[3] of (Murzin et al., 1995), version 1.53. The data consist of 4,352 sequences grouped into 54 families and 23 superfamilies selected by Liao and Noble. For each family, proteins within the family are considered positive test examples, and proteins within the superfamily but outside the family are considered positive training examples; negative examples are chosen outside the fold, and were randomly split into training and test sets in the same ratio respect to the positive examples. Therefore this assessment consists of 54 experiments, each corresponding to a target family and having at least 10 positive training examples (taken from its respective superfamily) and at least 5 positive test examples (taken directly from the family), and no sequence homologies known a priori. In these experiments there is usually a much larger number of negative examples than of positive examples.

We compared the Irredundant Class with the state-of-the-art methods using as metric the receiver operating characteristic (ROC) score. For each experiment, given a ranking of test example scores in output from the SVM, the ROC score is the normalized area under the curve that plots the number of true positive examples found as a function of the number of false positive examples detected (Gribskov and Robinson, 1996). This is like to plot the number of true and false positives found on a two-dimensional histogram (in abscissa the false positives, and in ordinate the true positives) for each different possible classification threshold based on SVM scores.

All methods compared are of discriminative nature, so we used a popular SVM software: the Gist SVM[4] described in (Noble and Pavlidis, 2002), version 2.3. Experimental results of the other methods were taken from (Saigo et al., 2004; Lingner and Meinicke, 2008).[5]

Table 3 shows the mean ROC scores, that is the average of ROC scores over all experiments, for the Irredundant Class and the other methods. These scores indicate that our method outperforms most methods in literature, and it is comparable to the state-of-the-art Local Alignment. For a more detailed

[2] The dataset is available at `http://noble.gs.washington.edu/proj/svm-pairwise`.

[3] SCOP, a protein classification constructed manually by visual inspection and comparison of structures, is available at `http://scop.mrc-lmb.cam.ac.uk/scop`.

[4] Gist SVM is available at `http://www.bioinformatics.ubc.ca/gist`.

[5] Details on the state-of-the-art results are available at `http://sunflower.kuicr.kyoto-u.ac.jp/~hiroto/project/homology.html`.

Table 3: Comparison of Experimental Results for State-of-the-Art Methods[c]

Method	Mean ROC
Irredundant Class	0.929
Local Alignment (version "eig")	0.925
Word Correlation Matrices (6)	0.904
Pairwise	0.896
Mismatch (5,1)	0.872
Spectrum (3)	0.824
Fisher	0.773

[c] Comparison of algorithms based on their mean ROC score over all experiments. Rows are listed from the best to the worst.

view, the ROC scores distribution is illustrated for some methods in Figure 2. The Local Alignment in green color (triangles) seems to perform slightly better than the Irredundant Class in blue (squares), but the minimum ROC score of the Local Alignment is much smaller. In particular, the smallest ROC score of our method was obtained in experiment 15 of (Liao and Noble, 2003) with a value of 0.614, while all other methods got their lowest peaks in experiment 13 with very small values, for example 0.22 for the Local Alignment. To confirm this fact, Figure 3 reports the ROC scores distribution showing in detail the trend for all experiments, and evidencing that the Irredundant Class exhibits, in general, a more robust behavior than the other methods.

Finally, Table 4 reports the number of irredundant common patterns against a less restrictive notion of patterns, the maximal common patterns introduced in (Comin and Verzotto, 2010), for 10 pairs of protein sequences taken from experiments. Results indicate that the number of irredundant common patterns $\mathcal{I}_{s_1,s_2}$ tends to be an order of magnitude smaller than the number of maximals $\mathcal{M}_{s_1,s_2}$, except for very short sequences (see pair numbers 9 and 10 of Table 4). Furthermore, Table 4 evidences that slightly relaxing the notion of irredundancy (for example considering maximal common patterns, that are in relation $\mathcal{I}_{s_1,s_2} \subseteq \mathcal{M}_{s_1,s_2}$) we tipically have a number of patterns that grows exponentially with the length of the sequences, while the irredundants are in every case less than $m + n - 3$, thus avoiding the

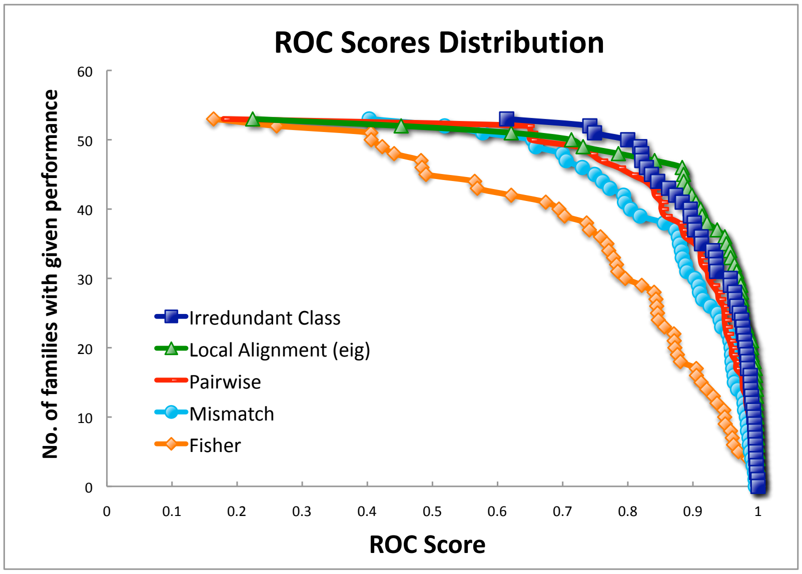

Figure 2: ROC scores distribution for the Irredundant Class and the state-of-the-art methods.

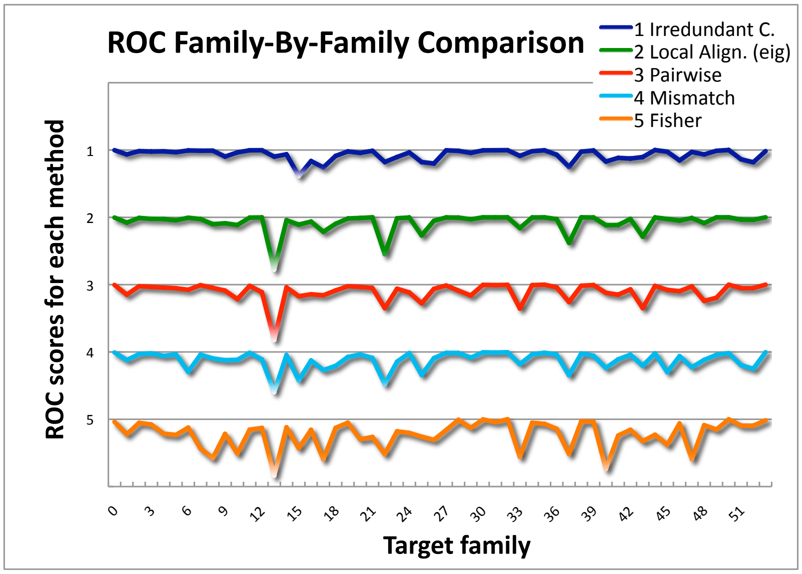

Figure 3: ROC scores across experiments.

Table 4: Counting the Number of Irredundant Common Patterns[d]

No.	s_1	s_2	m	n	$m+n$	$\lvert\mathcal{M}_{s_1,s_2}\rvert$	$\lvert\mathcal{I}_{s_1,s_2}\rvert$	% $\mathcal{I}_{s_1,s_2}$
1.	1alo_4	1bjt__	597	760	1,357	16,697	1,256	7.5
2.	1qaxa2	1cxp.1c	316	466	782	8,397	682	8.1
3.	1gai__	1nmta2	472	227	699	7,037	612	8.7
4.	1cvua1	1lgr_2	511	368	879	9,014	787	8.7
5.	1gpea1	1yrga_	392	343	735	6,853	653	9.5
6.	1qqja_	3pccm_	415	236	651	5,090	566	11.1
7.	1bxka_	1ofga1	352	220	572	3,549	489	13.8
8.	1ebfa1	2naca1	169	188	357	1,126	277	24.6
9.	1a03a_	1mho__	90	88	178	257	108	42.0
10.	1gpt__	1ayj__	47	50	97	64	45	70.3

[d] Number of irredundant ($\mathcal{I}_{s_1,s_2}$) against maximal ($\mathcal{M}_{s_1,s_2}$) common patterns over 10 pairs of protein sequences taken from experiments. Rows are listed according to the percentage of irredundants over the number of maximals, where $\mathcal{I}_{s_1,s_2} \subseteq \mathcal{M}_{s_1,s_2}$.

comparison of the same pair of characters of s_1 and s_2 a multiple number of times (see Section 3).

5 Conclusion

In the paper we study how the notion of irredundant common patterns can solve an issue that is rising in the field of string kernels, the remote homology detection. Almost all string kernels are based on patterns that are not independent, and therefore the associated scores are obtained using a set of redundant features. We specifically address this issue considering a particular class of patterns called irredundant common patterns. The method is based on the statistics of these patterns, and is called Irredundant Class. Results on benchmark data show that the Irredundant Class outperforms most of the string kernels previously proposed, and it achieves results as good as the current state-of-the-art method Local Alignment.

Despite its information properties, the Irredundant Class has a computational complexity that is much more than linear in the length of the sequences, and, in addiction, it processes the same characters of a single sequence a multiple number of times, due to pattern overlaps. Therefore we plan to study a new notion of common patterns to manage these issues and then to better fit the problem of remote homology detection of protein sequences.

References

Altschul, S. F., Madden, T. L., Schffer, A. A., Schaffer, R. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25:3389–3402.

Apostolico, A. and Parida, L. (2004). Incremental paradigms of motif discovery. *Journal of Computational Biology*, 11:15–25.

Apostolico, A. and Tagliacollo, C. (2008). Incremental discovery of the irredundant motif bases for all suffixes of a string in $O(n^2 \log n)$ time. *Theoretical Computer Science*, 408(2-3):106–115.

Baldi, P., Chauvin, Y., Hunkapiller, T., and McClure, M. (1994). Hidden Markov models of biological primary sequence information. *Proceedings of the National Academy of Sciences USA*, 91:1053–1063.

Ben-Hur, A. and Brutlag, D. L. (2003). Remote homology detection: a motif based approach. *Bioinformatics*, 19(Suppl. 1).

Comin, M. and Verzotto, D. (2010). Classification of protein sequences by means of irredundant patterns. *BMC Bioinformatics*, 11(Suppl. 1):S16.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press.

Eichhorn, J. (2007). *Applications of Kernel Machines to Structured Data.* PhD thesis, Technische Universität Berlin.

Fischer, M. J. and Paterson, M. S. (1974). String-matching and other products. Technical report, Massachusetts Institute of Technology.

Gao, Y., Yang, M., Wan, X., Mathee, K., and Narasimhan, G. (1997). Detection of HTH motifs via data mining. In *International Conference on Bioinformatics*.

Gribskov, M., McLachlan, A. D., and Eisenberg, D. (1987). Profile analysis: detection of distantly related proteins. *Proceedings of the National Academy of Sciences USA*, 84(13):4355–4358.

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

Haussler, D. (1999). Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California at Santa Cruz.

Henikoff, S. and Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences USA*, 89(22):10915–10919.

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

Jaakkola, T., Diekhans, M., and Haussler, D. (1999). Using the Fisher kernel method to detect remote protein homologies. In Lengauer, T., Schneider, R., Bork, P., Brutlag, D. L., Glasgow, J. I., Mewes, H.-W., and Zimmer, R., editors, *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology (ISMB 1999)*, pages 149–158. AAAI.

Karplus, K., Barrett, C., and Hughey, R. (1998). Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, 14:846–856.

Krogh, A., Brown, M., Mian, I. S., Sjölander, K., and Haussler, D. (1994). Hidden Markov models in computational biology: applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531.

Kuang, R., Ie, E., Wang, K., Wang, K., Siddiqi, M., Freund, Y., and Leslie, C. (2005). Profile-based string kernels for remote homology detection and motif extraction. *Journal of Bioinformatics and Computational Biology*, 3(3):527–550.

Leslie, C. S., Eskin, E., Cohen, A., Weston, J., and Noble, W. S. (2004). Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4):467–476.

Leslie, C. S., Eskin, E., and Noble, W. S. (2002). The spectrum kernel: a string kernel for SVM protein classification. In Altman, R. B., Dunker, A. K., Hunter, L., and Klein, T. E., editors, *Proceedings of the 7th Pacific Symposium on Biocomputing (PSB 2002)*, pages 564–575.

Liao, L. and Noble, W. S. (2003). Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. *Journal of Computational Biology*, 10:857–868.

Lingner, T. and Meinicke, P. (2008). Word correlation matrices for protein sequence analysis and remote homology detection. *BMC Bioinformatics*, 9:259.

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

Noble, W. S. and Pavlidis, P. (2002). Gist support vector machine and kernel principal components analysis software toolkit. Columbia University.

Parida, L. (1998). *Algorithmic Techniques in Computational Genomics.* PhD thesis, Courant Institute of Mathematical Sciences, New York University.

Parida, L., Rigoutsos, I., Floratos, A., Platt, D., and Gao, Y. (2000). Pattern discovery on character sets and real-valued data: linear bound on irredundant motifs and an efficient polynomial time algorithm. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*, pages 297–308. SIAM.

Pisanti, N., Crochemore, M., Grossi, R., and Sagot, M.-F. (2005). Bases of motifs for generating repeated patterns with wild cards. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(1):40–50.

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

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

Smith, T. F. and Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.