BIOINFORMATICS

Vol. 19 Suppl. 1 2003, pages i26–i33
DOI: 10.1093/bioinformatics/btg1002

Remote homology detection: a motif based approach

Asa Ben-Hur and Douglas Brutlag*

Department of Biochemistry, B400 Beckman Center, Stanford University, CA 94305-5307, USA

Received on January 6, 2003; accepted on February 20, 2003

ABSTRACT

Motivation: Remote homology detection is the problem of detecting homology in cases of low sequence similarity. It is a hard computational problem with no approach that works well in all cases.

Results: We present a method for detecting remote homology that is based on the presence of discrete sequence motifs. The motif content of a pair of sequences is used to define a similarity that is used as a kernel for a Support Vector Machine (SVM) classifier. We test the method on two remote homology detection tasks: prediction of a previously unseen SCOP family and prediction of an enzyme class given other enzymes that have a similar function on other substrates. We find that it performs significantly better than an SVM method that uses BLAST or Smith-Waterman similarity scores as features.

Availability: The software is available from the authors upon request.

Contact: asa.benhur@stanford.edu

Keywords: remote homology, discrete sequence motifs, sequence similarity, Support Vector Machines, kernel methods

INTRODUCTION

Protein homology detection is one of the most important problems in computational biology. Homology is generally established by sequence similarity. Many methods for measuring similarity have been studied in the past two decades. The two most established methods are the Smith-Waterman algorithm (Smith and Waterman, 1981) and its heuristic, faster counterpart, BLAST (Altschul *et al.*, 1997). Protein sequence motifs are an alternative way of detecting sequence similarity. Motifs are usually constructed from multiple sequence alignments of related sequences. As a preliminary step, one extracts from an alignment 'blocks' which are ungapped regions of high sequence similarity. The blocks are then described either by Position Specific Scoring Matrices (PSSMs), which indicate the relative abundance of each amino acid at each position in the block, or by discrete sequence motifs, which indicate the possible amino acids at each position.

By focusing on limited, highly conserved regions of proteins, motifs can often reveal important clues to a protein's role even if it is not globally similar to any known protein (Nevill-Manning *et al.*, 1998). The motifs for most catalytic sites and binding sites are conserved over much wider taxonomic distances and evolutionary time than are the sequences of the proteins themselves. Thus, motifs often represent functionally important regions such as catalytic sites, binding sites, protein-protein interaction sites, and structural motifs.

The first database of protein motifs was the PROSITE database (Falquet *et al.*, 2002), whose discrete motifs are manually constructed. Other databases derived from multiple sequence alignments of protein families include BLOCKs+, PRINTs, pFAM, ProDom, DOMO, and InterPro; see Henikoff *et al.* (1999) and references therein. The BLOCKs+ database combines these databases (Henikoff *et al.*, 1999); the eMOTIF database contains discrete sequence motifs constructed from the blocks of BLOCKS+ (Huang and Brutlag, 2001). Unlike the PROSITE and PRINTS databases, eMOTIFs are constructed automatically in a principled approach that has been shown to perform well (Nevill-Manning *et al.*, 1998). More recently, the eBLOCKS database of blocks was constructed (Su *et al.*, 2003). It presents a systematic approach for creating blocks using groups of proteins constructed using clustered PSI-BLAST results. It is more comprehensive than BLOCKS+, and motifs constructed from it have better coverage than BLOCKS+ motifs. This paper uses discrete sequence motifs extracted from the eBLOCKs database using the eMOTIF method.

In this paper we introduce a sequence similarity measure based on the motif content of a pair of sequences. A simple way to use a sequence similarity to annotate a novel protein is by a nearest neighbor type of approach: decide on the annotation according to the annotations of the nearest neighbor(s). However, when the novel protein has no highly similar protein in the training data the

*To whom correspondence should be addressed.

problem becomes more difficult, and this straightforward approach will often fail, requiring a better discriminative approach (Liao and Noble, 2002). Such an approach for remote homology detection of structural domains was used by Jaakkola *et al.* (1999). Their method, called the Fisher kernel method, is based on Support Vector Machines (SVMs) trained on features extracted from Hidden Markov Models (HMMs). An SVM method that uses Smith-Waterman similarity scores was shown in (Liao and Noble, 2002) to perform better than several methods for homology detection including the Fisher kernel method, the HMM based SAM T-98 and PSI-BLAST.

When a sequence similarity can be shown to be a dot product in some space, it is called a *kernel*. This is important since many successful machine learning methods such as SVMs are defined in terms of a kernel (Schölkopf and Smola, 2002; Cristianini and Shawe-Taylor, 2000). Kernels appropriate for sequence objects were suggested in (Leslie *et al.*, 2002; Vishwanathan and Smola, 2002); the mismatch kernel (Leslie *et al.*, 2002) was shown to match the performance of the Fisher kernel in remote homology detection.

In this paper we use protein motifs to construct a kernel that can be computed efficiently; we show that this kernel performs significantly better than a kernel based on BLAST or Smith-Waterman scores. We tested the methods on two tasks: prediction of a SCOP family when trained on other families in that family's fold and prediction of the function of an enzyme when the training set contains enzymes that have the same general function, but have different substrates. We find that good performance is obtained when using a combination of a good similarity measure (kernel), and a state of the art classifier: the performance motif-based kernel is significantly improved when coupled to an SVM classifier rather than a nearest neighbor classifier.

REMOTE HOMOLOGY DETECTION

We will use two datasets to simulate the problem of remote homology detection. The first, following (Jaakkola *et al.*, 1999) is composed of sequences of domains from the Structural Classification Of Proteins (SCOP) database (Murzin *et al.*, 1995). The objective is to detect homology at the SCOP superfamily level: recognizing a SCOP family when the training set contains other families in the family's superfamily. This specifies the positive examples in the test set and the training set. The negative examples are taken from families outside of the family's fold. The division of the negative examples into training set and test set follows the same idea as in the construction of the positive examples: a random family is chosen to belong to the negative test set, and the rest of the families in its superfamily are added to the negative training set. This setup is illustrated in Figure 1.

In the second dataset we use the classification of enzymes to simulate remote homology. The function of an enzyme is specified by a name given to it by the Enzyme Commission (EC) (NC-IUBMB, 1992). The name corresponds to an *EC number*, which is of the form: `n1.n2.n3.n4`, e.g. `1.1.3.13` for alcohol oxidase. The first number is between 1 and 6, and indicates the type of chemical reaction catalyzed by the enzyme; enzymes fall into the main categories of oxidoreductases, transferases, hydrolases, lyases, isomerases and ligases. The remaining numbers have meanings that are particular to each class. For example, in the oxidoreductase class of enzymes (EC number starting with 1), which involve reactions in which hydrogen or oxygen atoms or electrons are transferred between molecules, `n2` specifies the chemical group of the (electron) donor molecule, `n3` specifies the (electron) acceptor, and `n4` specifies the substrate. In this paper we concentrate on oxidoreductases. We will train a classifier to predict oxidoreductases with a certain function (`n2` and `n3` specified); the classifier will be tested on oxidoreductases with a different substrate (`n4`) than those it was trained on. For example, the training set may include as positive examples the EC classes `1.14.13.8` and `1.14.13.11`, while the positive examples of the test set will include the EC class `1.14.13.39`, which has the same general function, but on a different substrate. Enzymes that do not share a common substrate will have lower sequence similarity than enzymes that do, so the sequence similarity between the positive training set and positive test set may not be very high. The negative training and test set are defined analogously. This task tests the ability to find sequence characteristics that are independent of the specific substrate.

METHODS

Most classification methods require a measure of similarity. When the similarity is a dot product, it is called a *kernel*. Kernel methods are a growing field in machine learning, and include Support Vector Machines (SVM) (Boser *et al.*, 1992; Schölkopf and Smola, 2002; Cristianini and Shawe-Taylor, 2000).

The motif kernel

The kernel defined here is a dot product between sequences over the alphabet of amino acids; the kernel will be defined in terms of the sequence motifs that appear in a pair of sequences. We use discrete sequence motifs extracted using the eMOTIF method (Nevill-Manning *et al.*, 1998; Huang and Brutlag, 2001), which is described here briefly. Each position in the motif represents the variability in a column in a block from a

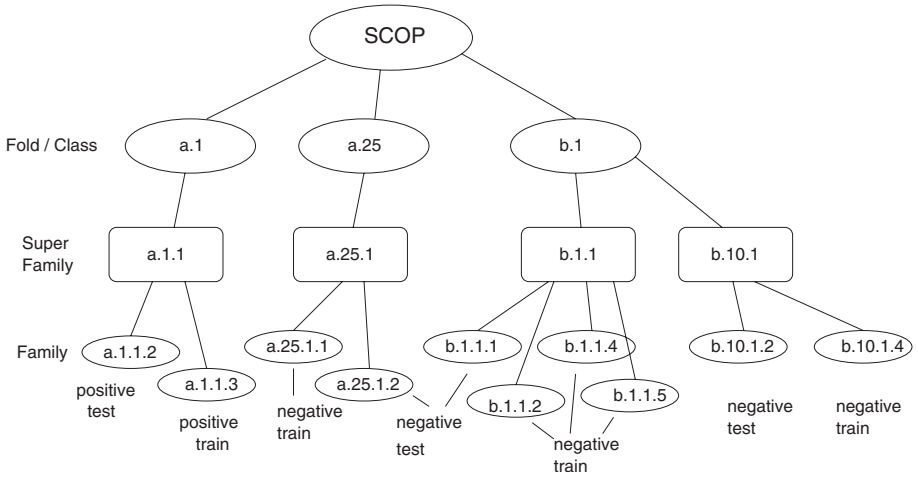

Fig. 1. The organization of the SCOP database into folds, super-families and families, and the division of families into positive/negative train/test sets, where the objective is to predict the family `a.1.1.2`; the training set includes the family `a.1.1.3` as the positive training sequences. In the negative set of sequences, a randomly chosen family out of a superfamily is added to the negative test set, while the rest of the families in the superfamily are added to the negative training set.

multiple sequence alignment. Take for example the motif `[as].dkf[filmv]..[filmv]...l[ast]`. A substitution group such as `[filmv]` denotes the appearance of several amino acids in a particular column in a block. eMOTIFS contain only a limited number of substitution groups that reflect chemical and physical properties of amino acids and their tendency to co-occur in multiple sequence alignments. If the pattern of amino acids that appear in a column of a block does not match any substitution group, then the motif contains the wildcard symbol, '.'. A sequence will match (or contain) the above motif if it has either an `a` or an `s` in some position, then any character, then `d`, `k`, `f` and so on, matching until the end of the motif. A formal definition is as follows:

DEFINITION 1. Let $\mathcal{A}$ be an alphabet. A *substitution group* $\mathcal{S} = \{s_1, \ldots, s_k\}$ is a subset of $\mathcal{A}$. It is written as $[s_1 \ldots s_k]$. Let $\bar{\mathcal{S}}$ be a set of substitution groups, and let '.' denote the wildcard character.
A *motif* m is a sequence over $\mathcal{A} \cup \bar{\mathcal{S}} \cup \{.\}$.
A sequence $x = x_1 x_2 \ldots x_{|x|} \in \mathcal{A}^*$ is said to *contain* a motif m at position i if for $j = 1, \ldots, |m|$, if $m_j \in \mathcal{A}$ then $x_{i+j-1} = m_j$; if m_j is a substitution group $\mathcal{S}$ then $x_{i+j-1} \in \mathcal{S}$; if m_j is the wildcard character, then x_{i+j-1} can be any character. A sequence x contains a motif m, if x contains m at some position.

A sequence x can be represented in a vector space indexed by a set of motifs $\mathcal{M}$:

$$\Phi(x) = (\phi_m(x))_{m \in \mathcal{M}}, \tag{1}$$

where $\phi_m(x)$ is the number of occurrences of the motif m in x. Now define the *motif kernel* as:

$$K(x, x') = \Phi(x) \cdot \Phi(x'). \tag{2}$$

Since in most cases a motif appears only once in a sequence, this kernel essentially counts the number of motifs that are common to both sequences.

The motif vector space is high dimensional: the eBLOCKs database that is used in this study contains close to 500 000 motifs. Thus, the motif representation is very sparse, since a sequence typically contains a few tens of motifs.

There are several motif databases that can be used to define a motif kernel, including PROSITE and the eMOTIF database that is based on the BLOCKS+ set of blocks. However, the usage of these databases presents a problem in the evaluation of the performance of the kernel: the BLOCKS+ blocks are constructed in a supervised way from known protein families, i.e. proteins are grouped into blocks according to information about the function or structure of a protein. Thus the method cannot be tested on sequences that were used in the development of the database. The eBLOCKS database on the other hand is generated in an unsupervised way. For this reason, and because of the increased coverage of the eBLOCKS set of blocks, we chose to use motifs that were generated from the eBLOCKS set of blocks. Briefly, the eBLOCKS database is constructed as follows: Each protein sequence in SwissProt is used as a PSI-BLAST query to search against the entire SwissProt database. The PSI-BLAST results are clustered to build protein groups with varying

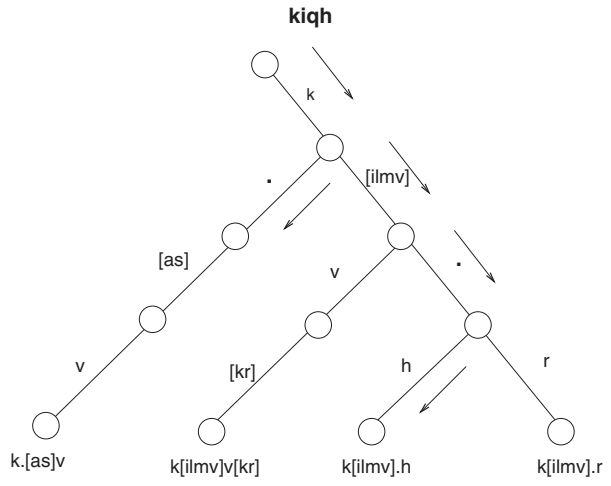

Fig. 2. Motifs are stored in the leaves of a TRIE. The figure shows a TRIE storing the motifs `k.[as]v`, `k[ilmv]v[kr]`, `k[ilmv].h` and `k[ilmv].r`. To find the motif content, the tree is traversed, matching at each position a letter from the sequence with the same letter, a substitution group containing the letter, or the wildcard symbol. Traversing the tree shows that the sequence `kiqh` contains the motif `k[ilmv].h`.

levels of similarity. Each group of sequences are aligned and trimmed into blocks, from which motifs are made using the eMOTIF method; see Su *et al.* (2003) for details.

Computing the motif kernel

The motif representation of a sequence is high-dimensional: its dimensionality equals the number of motifs in the database, 483 521 for the eBLOCKS-based motifs. Although the number of motifs is large, a given sequence contains a relatively small number of motifs. Our approach is to compute the motif content of each sequence; the subsequent computation of the kernel is simply a dot product between sparse vectors. To facilitate efficient computation of the motif content of a sequence, the motif database is stored in a TRIE (Knuth, 1998), which is defined as follows. Let m be a motif over the alphabet $\mathcal{A} \cup \bar{\mathcal{S}} \cup \{.\}$. Every prefix of m has a node; let m_1 and m_2 be prefixes of m; there is an edge from m_1 to m_2 if $|m_2| = |m_1| + 1$. The leaf nodes of the TRIE store the motifs. To find all motifs that are contained in a sequence x at a certain position, traverse the TRIE using DFS and record all the leaf nodes encountered during the traversal (see Fig. 2 for an illustration). To find all motifs that are contained in x at any position, this search is started at each position of x. Thus the computation time of the motif content of a sequence is linear in its length.

The BLAST kernel

BLAST scores (either bit-scores or E-values) are perhaps the most widely accepted measure of sequence similarity in the field of computational biology (Altschul *et al.*, 1997). Following Liao and Noble (2002) we represent a query sequence by its BLAST scores against the training set. This representation, in conjunction with SVMs, was used successfully by Liao and Noble (2002) to address the problem of remote homology detection. It was shown there to be superior to several methods, including the Fisher kernel method.

Classification methods

In this paper we report results using two classification methods: SVMs and k-Nearest-Neighbors (kNN). SVMs are two-class methods: a linear SVM has a decision function of the form

$$f(x) = w \cdot x + b, \tag{3}$$

where w is a weight vector, and b is a constant bias, and a query is classified according to the sign of f. The vector w and the bias, b, are chosen to maximize the margin between the decision surface (hyperplane) and the positive examples on one side, and negative examples on the other side, in the case of linearly separable data. In the case of non-separable data some slack is introduced (Schölkopf and Smola, 2002; Cristianini and Shawe-Taylor, 2000). As a consequence of the optimization process, the weight vector can be expressed as a weighted sum of the Support Vectors (SV):

$$w = \sum_{i \in SV} \beta_i x_i. \tag{4}$$

The decision function is now written as:

$$f(x) = \sum_{i \in SV} \beta_i x_i \cdot x + b. \tag{5}$$

To extend the usefulness of SVMs to include nonlinear decision functions and non-vector data, one proceeds as follows (Schölkopf and Smola, 2002; Cristianini and Shawe-Taylor, 2000): map the data into a feature space, typically high dimensional, using a map Φ, and then consider the dot product $\Phi(x) \cdot \Phi(x')$. This is possible, since the SVM optimization problem can be expressed in terms of dot products. This approach is practical if the so called *kernel function*, $K(x, x') = \Phi(x) \cdot \Phi(x')$, can be computed efficiently. In terms of the kernel function, the decision function is expressed as:

$$f(x) = \sum_{i \in SV} \beta_i K(x_i, x) + b. \tag{6}$$

A kNN classifier works by classifying a query pattern according to the class label of the most similar patterns

from the training set. We use a kNN classifier with a continuous valued decision function. A score for class j is defined by

$$f_j(x) = \sum_{i \in \mathrm{kNN}_j(x)} K(x_i, x), \qquad (7)$$

where $\mathrm{kNN}_j(x)$ is the set of k nearest neighbors of x in class j; a query x is classified to the highest scoring class.

In unbalanced learning tasks where one class is much larger than the other, error rates are not a good measure of the performance of a classifier. Thus we consider two metrics for assessing the performance of a classifier: the area under the Receiver Operator Characteristic (ROC) curve (Egan, 1975), and the median Rate of False Positive (RFP) (Jaakkola *et al.*, 1999). Since all the classifiers used here produce a ranking of the test examples according to the value of the decision function, these statistics can be used. The ROC curve describes the trade-off between sensitivity and specificity; it is a plot of the true positive rate as a function of the false positive rate for varying classification thresholds (Egan, 1975). The area under the ROC curve is commonly used to summarize the ROC curve. More specifically, we use the ROC50 curve, which counts true positives only up to the first 50 false positives. A classifier that correctly classifies all the data has an ROC50 score equal to 1, while if the top 50 values of the decision function belong to false positives the score is 0. The RFP score of a positive test sequence x is the fraction of negative test sequences that have a value of the decision function that is at least as high as the value of the decision function of x (Jaakkola *et al.*, 1999).

RESULTS

We used the ASTRAL database (Brenner *et al.*, 2000) to obtain a non-redundant set of protein domain sequences from version 1.59 of the SCOP database with less than 95% identity. We kept only superfamilies that have at least two families with at least 10 members in each family. This yielded a dataset with 1639 domains in 23 superfamilies and 56 families.

Protein sequences annotated with EC numbers were extracted from the SwissProt database Release 40.0 (O'Donovan *et al.*, 2002). EC numbers were extracted from the description lines; we removed sequence fragments, and sequences where the assigned EC number was designated as 'putative' or assigned by homology. Sequences with an incompletely specified EC number were removed as well. Enzyme classes with a small number of representatives (less than 10) were not considered. The extracted dataset has 2187 enzymes in 65 classes.

To generate the BLAST kernel we ran an all-vs-all BLAST on these two datasets using the default parameters and an E-value cutoff of 0.1. The motif kernel for these datasets was computed with eBLOCKS sequence motifs using the TRIE method described earlier. The current version of eBLOCKs is based on the protein sequences contained in SwissProt release 37.

We ran our classification experiments using a python machine learning package that provides several machine learning methods including a wrapper for the LIBSVM package (Chang and Lin, 2002). The class imbalance in the dataset was taken into account by using a soft margin constant that is inversely proportional to the class probabilities.

The results reported for the BLAST kernel are for a linear kernel in the vector space of BLAST scores (-log of the E-value). The results were not improved by the use of other types of kernels. A family by family comparison of classification performance of the motif-SVM and BLAST-SVM methods is provided in Figure 3. On the SCOP task, the motif-SVM method performs significantly better than the BLAST-SVM method, with a p-value of $3 \cdot 10^{-9}$ in a Wilcoxon signed rank test for the ROC50 score, and a lower value for the ROC score. In the enzyme classification task there is no significant difference in the ROC50 scores. The ROC scores show a difference in favor of the motif-SVM method, with a p-value of 0.001; it remains significant under adjustment for multiple comparisons. Similar behavior is observed in the median RFP and its analogous RFP50 statistic. The results were very similar when the Smith-Waterman algorithm was used instead of BLAST, also with lower value BLOSUM matrices used to better detect remote homology.

In Figure 4 we show a visualization of the motif kernel matrix and the BLAST kernel matrix for one superfamily of the SCOP database. The motif kernel shows similarity between families in the superfamily, whereas none is detected by the BLAST kernel, even at the very low E-value of 0.1 that was used. This is consistent with the observation that motifs are often able to detect important sequence similarity across highly divergent sequences (Nevill-Manning *et al.*, 1998). This increased sensitivity allows the motif kernel to perform well in many cases where the BLAST kernel performs poorly (Fig. 3).

In Figure 5 we show the comparison of the SVM-based methods to ones that use kNN as a classifier; as a reference we also include results using a method that assigns the class according to the class of the training sequence with which a query has the maximum BLAST score, rather than using the pattern of BLAST scores as in the BLAST kernel. In both the motif and BLAST kernels the SVM-based classifier performs significantly better than the corresponding kNN classifier (significance levels were less than 10^{-8} using a Wilcoxon signed rank test).

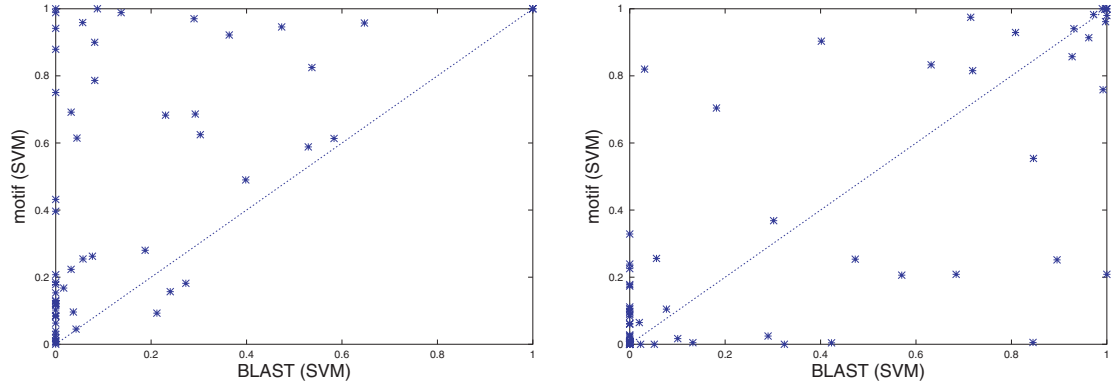

Fig. 3. Comparison of the ROC scores of the motif kernel and blast kernels with an SVM classifier. The axes are ROC scores achieved by these methods on the SCOP classification task (left) and the enzyme classification task (right). Each datapoint represents a SCOP family or EC class. The performance of the motif kernel on the SCOP classification task is significantly better than the BLAST kernel (p-value less than 10^{-8} in a Wilcoxon signed rank test).

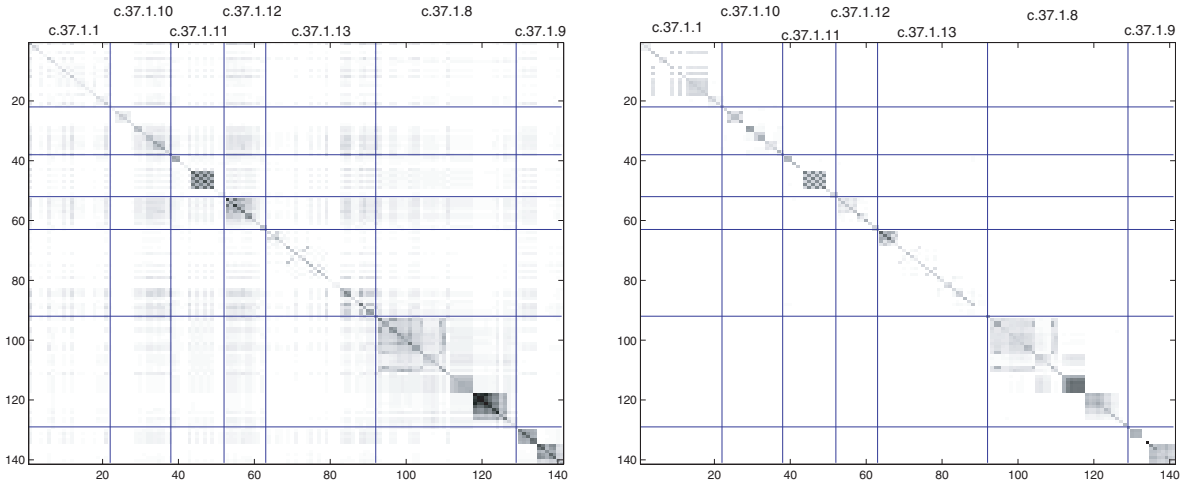

Fig. 4. Visualization of the motif kernel matrix (left) and the BLAST kernel (right) for one SCOP superfamily that has 140 domains. The division into families is indicated by horizontal and vertical lines. Grey level denotes the level of similarity. The BLAST kernel detects no similarity between families despite our use of a low E-value threshold of 0.1; the motif kernel shows inter-family similarity, but it is lower than the intra-family level of similarity. This difference between the two kernels is reflected in the ROC50 scores for these families: for the motif-SVM method they are 0.68, 0.98, 1.0, 0.69, 0.98, 0.94; for the SVM-BLAST method they are 0.08, 0.29, 0.13, 0.08, 0.03. The SCOP family shown here is in the class of alpha and beta proteins (a/b); more specifically, P-loop containing nucleotide triphosphate hydrolases, where the division into families is based on beta-sheet topologies.

DISCUSSION

In this paper we compared two approaches for remote homology detection: one based on the motif content of a sequence, and another, based on BLAST/Smith–Waterman scores against the training set. We defined a kernel function that reflects the similarity in the motif content of two sequences, and showed that an SVM classifier that uses the motif kernel performs significantly better than an SVM that uses a BLAST/Smith–Waterman kernel on a remote homology detection problem derived from the SCOP database. The two methods had very similar performance on a task of enzyme classification. In a related paper, an SVM-based method that uses sequence similarity scores computed by the Smith–Waterman algorithm was shown to perform better than state-of-the-art methods such as the Fisher kernel (Liao and Noble, 2002). It is interesting to note that the BLAST methods performed as well as the Smith–Waterman methods, even

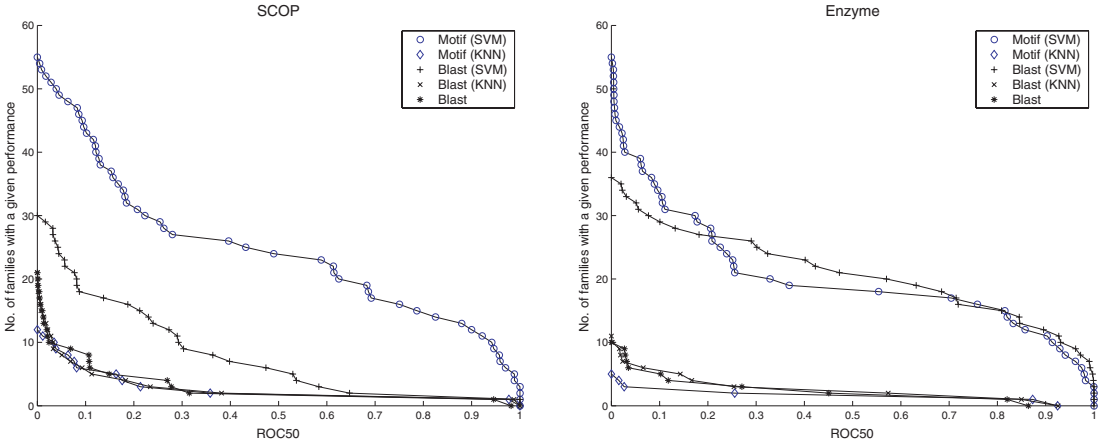

Fig. 5. Comparison of the performance of the five methods on the SCOP classification task (left) and the enzyme classification task (right). The plain BLAST method assigns a class label according to the class of the sequence with which it has the largest BLAST similarity score.

though it was noted that the Smith–Waterman algorithm is sometimes better than BLAST in detecting remote homology.

We found that the performance of the method cannot be attributed only to the similarity measure (kernel): both the motif kernel and the BLAST kernel worked significantly better when used in conjunction with an SVM rather than a nearest neighbor classifier. This is a result of the difficulty of the learning problem and the high dimensionality of the feature space. SVMs have consistently shown to perform better than other methods in such cases (Cristianini and Shawe-Taylor, 2000; Schölkopf and Smola, 2002).

The motif kernel is related to other string kernels recently proposed in the literature. In the spectrum kernel (Leslie *et al.*, 2002) the feature space is the set of all n-grams. The mismatch kernel (Leslie *et al.*, 2002) uses n-grams, and allows a bounded number of mismatches, while the method of Vishwanathan and Smola (2002) uses all substrings and weighs them by their length. These and other string kernels are completely general in their applicability. The motif kernel on the other hand is specific to the problem of protein classification, with domain knowledge such as the chemical properties of amino acids built into it in the process of the motif construction (Nevill-Manning *et al.*, 1998). It would be interesting to see if this prior information makes the motif kernel perform better than these more general methods. Another difference is in the dimensionality of the feature space: an n-gram kernel has a feature space whose dimensionality is 20^n for amino acid sequences, with n between 4 and 6. This makes feature selection difficult to apply. The motif kernel on the other hand contains a much smaller number of features (a few hundred thousand), and each feature is very informative: in many cases the presence of a single motif is sufficient to classify a sequence. Thus feature selection is more practical; in addition to the improvement in speed, one obtains a greater interpretability of the resulting classifier.

Despite the relative success of the motif method, there is much room for improvement: there were many SCOP families and EC classes that were not detected using this method. We are currently building motifs from SCOP families and EC classes in an effort to obtain even better sets of motifs.

ACKNOWLEDGEMENTS

This work is supported by NIH grant 5R01 HG02235-08 and the Bio-X Interdisciplinary Initiatives Program.

REFERENCES

Altschul,S., Madden,T., Schaffer,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 Res.*, **25**, 3389–3402.

Boser,B.E., Guyon,I.M. and Vapnik,V.N. (1992) A training algorithm for optimal margin classifiers. In *5th Annual ACM Workshop on COLT*. ACM Press, Pittsburgh, pp. 144–152.

Brenner,S., Koehl,P. and Levitt,M. (2000) The ASTRAL compendium for sequence and structure analysis. *Nucleic Acids Res.*, **28**, 254–256.

Chang,C. and Lin,C. (2002) Libsvm: a library for support vector machines (version 2.36) http://www.csie.ntu.edu.tw/∼cjlin/libsvm/.

Cristianini,N. and Shawe-Taylor,J. (2000) *An Introduction to Support Vector Machines*. Cambridge, UP.

Egan,J. (1975) *Signal detection theory and ROC analysis*, Series in Cognition and Perception, Academic Press, New York.

Falquet,L., Pagni,M., Bucher,P., Hulo,N., Sigrist,C., Hofmann,K. and Bairoch,A. (2002) The PROSITE database, its status in 2002. *Nucliec Acids Res.*, **30**, 235–238.

Henikoff,S., Henikoff,J. and Pietrokovski,S. (1999) Blocks+: a non-redundant database of protein alignment blocks derived from multiple compilations. *Bioinformatics*, **15**, 471–479.

Huang,J. and Brutlag,D. (2001) The emotif database. *Nucleic Acids Res.*, **29**, 202–204.

Jaakkola,T., Diekhans,M. and Haussler,D. (1999) Using the Fisher kernel method to detect remote protein homologies. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, Menlo Park, CA, pp. 149–158.

Knuth,D. (1998) *The art of computer programming volume 3: sorting and searching*. Addison Wesley.

Leslie,C., Eskin,E. and Noble,W. (2002) The spectrum kernel: a string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing*. World Scientific, pp. 564–575.

Leslie,C., Eskin,E., Weston,J. and Noble,W.S. (2002) Mismatch string kernels for SVM protein classification. In *Advances in Neural Information Processing Systems*.

Liao,L. and Noble,W.S. (2002) Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology*. pp. 225–232.

Murzin,A., Brenner,S., Hubbard,T. and Chothia,C. (1995) Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, **247**, 536–540.

Nevill-Manning,C., Wu,T. and Brutlag,D. (1998) Highly specific protein sequence motifs for genome analysis. *Proc. Natl Acad. Sci. USA*, **95**, 5865–5871.

O'Donovan,C., Martin,M., Gattiker,A., Gasteiger,E., A.B.A., and Apweiler,R. (2002) High-quality protein knowledge resource: SWISS-PROT and TrEMBL. *Brief Bioinform.*, **3**, 275–284.

Nomenclature Committee of the International Union of Biochemistry and Molecular Biology (NC-IUBMB) (1992) Enzyme Nomenclature. Recommendations 1992, Academic Press.

Schölkopf,B. and Smola,A. (2002) *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA.

Smith,T. and Waterman,M. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.

Su,Q., Saxonov,S. and Brutlag,D. (2003) eBLOCKS: an automated database of protein conserved regions maximizing sensitivity and specificity. *http://motif.stanford.edu/eblocks*.

Vishwanathan,S. and Smola,A. (2002) Fast kernels for string and tree matching. In *Advances in Neural Information Processing Systems*.