Automatic prediction of protein domains from sequence information using a hybrid learning system

Niranjan Nagarajan and Golan Yona*

Department of Computer Science, Cornell University, Upson Hall, Ithaca, NY 14853, USA

Received on August 6, 2003; revised on December 7, 2003; accepted on December 12, 2003
Advance Access publication February 12, 2004

ABSTRACT
Motivation: We describe a novel method for detecting the domain structure of a protein from sequence information alone. The method is based on analyzing multiple sequence alignments that are derived from a database search. Multiple measures are defined to quantify the domain information content of each position along the sequence and are combined into a single predictor using a neural network. The output is further smoothed and post-processed using a probabilistic model to predict the most likely transition positions between domains.

Results: The method was assessed using the domain definitions in SCOP and CATH for proteins of known structure and was compared with several other existing methods. Our method performs well both in terms of accuracy and sensitivity. It improves significantly over the best methods available, even some of the semi-manual ones, while being fully automatic. Our method can also be used to suggest and verify domain partitions based on structural data. A few examples of predicted domain definitions and alternative partitions, as suggested by our method, are also discussed.

Availability: An online domain-prediction server is available at http://biozon.org/tools/domains/
Contact: niranjan@cs.cornell.edu; golan@cs.cornell.edu

1 INTRODUCTION

One of the first steps in analyzing proteins is to detect the constituent domains or the domain structure of the protein. A domain is considered as the fundamental unit of protein structure, folding, function, evolution and design (Rose, 1979; Lesk and Rose, 1981; Holm and Sander, 1994). It combines several secondary structure elements and motifs, not necessarily contiguous, which are packed in a compact globular structure. It is commonly believed that a domain can fold independently into a stable three-dimensional (3D) structure and that it has a specific function. A protein may comprise a single domain or several different domains or several copies of the same domain. It is the domain structure of a protein that determines its function, the biological pathways in which it is involved and the molecules it interacts with.

Detecting the domain structure of a protein is a challenging problem. Given the protein sequence, there are no clear signals or signs that indicate when one domain ends and another begins. Structural information can help in detecting the domain structure of a protein. Domain delineation based on structure is currently best done manually by experts, and the SCOP domain classification (Murzin et al., 1995) which is based on extensive expert knowledge, is an excellent example. However, structural information is available for only a small portion of the protein space. Therefore, there is a strong interest in detecting the domain structure of a protein directly from the sequence.

In our study, we define a domain to be a continuous sequence that corresponds to an elemental building block of protein folds—a subsequence that is likely to be stable as an independent folding unit. As such we believe that this building block was first formed as an independent protein with a specific acquired function. In the course of evolution, the domain might have been combined with additional domains to perform other, possibly more complex, functions. However, if the domain indeed existed at some point as an independent unit, then it is likely that traces of the autonomous unit might exist in other database sequences, possibly in lower organisms. Thus, a database search can sometimes provide us with ample information on the domain structure of a protein. For example, the histogram and profile of sequence matches one can obtain from a database search may help to detect domain boundaries (Yona and Levitt, 2000b; Kuroda et al., 2000; George and Heringa, 2002a). However, one should be cautious in analyzing database matches in search of such signals. One possible difficulty arises from the fact that pairs of sequence domains may appear in many related sequences, thus hindering the ability to discern the two apart. Furthermore, mutations, insertions and deletions blur domain boundaries and make it hard to distinguish a signal from background noise.
1.1 Related studies

Previous methods for sequence-based domain detection could be roughly classified into five categories: (i) Methods based on the use of similarity searches and knowledge of sequence termini to delineate domain boundaries using heuristics. Methods like MKDOM (Gouzy et al., 1999), Domainer (Sonhammer and Kahn, 1994), DIVCLUS (Park and Teichmann, 1998) and DOMO (Gracy and Argos, 1998) fall in this category. These methods were designed to partition all the proteins in a database into domains, but they are in general less accurate due to their heuristic nature. (ii) Methods that rely on expert knowledge of protein families to construct models like hidden Markov models (HMMs) and artificial neural networks (ANNs) to identify other members of the family. PFam A (Sonhammer et al., 1997; Bateman et al., 1999), TigrFam (Haft et al., 2001), SMART (Ponting et al., 1999) and Murvai et al. (2001) in this category. These methods are considerably more accurate but are restricted by their ability to make predictions only for well-studied families. (iii) Methods that try to infer domain boundaries by using sequence information to predict tertiary structure first. SnapDragon (Geor and Heringa, 2002b) and Rigden’s covariance analysis (Rigden, 2002) are examples of this approach. These methods use novel sources of information but are computationally expensive. (iv) Methods that use multiple alignments to predict domain boundaries such as PASS (Kuroda et al., 2000) and Domination (George and Heringa, 2002a). (v) Other methods that do not fall into any of the previous categories [clustering sequence alignments (Guan and Du, 1998; Miyazaki et al., 2002) and domain guess by size (Wheelan et al., 2000)]. A more detailed description of the five categories follows.

1.1.1 Methods based on similarity search Of the similarity search-based algorithms, MKDOM is conceptually the simplest and most efficient and is currently employed in the generation of the ProDom database. The algorithm works on the assumption that the smallest repeat-free sequence fragment in a database is likely to correspond to a single domain (all fragments smaller than a threshold are automatically removed from the database). Significant matches with the fragment are extracted from all sequences in the database, and the process is repeated on the new database until no more fragments remain. The Domainer algorithm works by doing an all-versus-all BLAST search to identify segment pairs with high degree of homology. These segment pairs are then iteratively merged based on overlap measures to form homologous segment sets (HSSs), and links are maintained between HSSs that have fragments that follow each other sequentially in a protein sequence. The resulting HSS graph is then partitioned into domains (sets of HSSs) using sequence termini and information about cycles in the graph as domain transition signals. The DIVCLUS program starts with an all-versus-all search as well, but it uses SSEARCH or FASTA to get gapped alignments. The resulting pairs are then clustered using single linkage clustering. Finally, DIVCLUS attempts to split the clusters into smaller clusters using various measures of overlap between sequences in combination with some thresholds (e.g. overlap of at least 30 amino acids that covers at least 70% of the shorter of the two sequences). The DOMO algorithm clusters sequences into groups by comparing their amino acid and dipeptide composition. Each cluster is represented by one sequence, and the representatives are compiled into a suffix tree. This tree is self-compared to detect ungapped local sequence similarities. The resulting pairs form the seed anchors which are intersected with other anchors based on either the presence of a significantly overlapping common subsequence or common position relative to another anchor. The anchor merging process is accompanied by a controlled interval intersection process that finally determines the domain boundaries for the proteins.

1.1.2 Methods based on expert knowledge The Pfam database (Bateman et al., 1999) combines manual and automatic approaches to classify proteins into domain families. The database is split into two parts, Pfam A, which is composed of families generated from high-quality multiple alignments and verified using structural and functional information with substantial manual involvement, and Pfam B, which is generated using the Domainer algorithm on the rest of the sequence database. No specific rules are used to define domain boundaries other than the judgment of human experts and structural information (when available) from SCOP about the domain structure of proteins. The SMART classification is similar to Pfam A in that it is based on HMMs constructed from manually checked, high-quality multiple alignments, with the difference being that SMART focuses on domains occurring in signaling proteins. The TigrFam database is constructed using the same methodology as in Pfam A and SMART but is geared toward the identification of functionally similar subsequences rather than domains. Instead of using HMMs to learn models for domain families, the work by Murvai et al. (2001) is based on the use of ANNs for this purpose. The data used to construct the models is in the form of statistics gathered from BLAST comparisons with members and non-members of the various domain families.

1.1.3 Methods that use predicted 3D information Recent studies on sequence-based domain delineation have also explored other sources of information to detect domain boundaries. The SnapDragon method works by first generating many ab initio 3D model structures of a protein using the hydrophobicity information in multiple alignments and predicted secondary structure information in Monte Carlo folding simulations. Domain boundaries for each of these 3D models are then computed based on structural considerations as described in Taylor (1999), and finally the consistency between the definitions for the various models is used to partition the protein into domains. Rigden (2002) paper on covariance analysis uses information from the calculation of correlated mutation
values for alignment columns to predict contacts in a protein. The predicted contact information is then used to construct a contact profile where local minima in the profile are used to predict domain boundaries.

1.1.4 Methods based on multiple alignments Domination and Pass are multiple alignment-based algorithms. Domination is an iterative algorithm that uses PSI-BLAST to do a database search and generate an initial pairwise alignment-based multiple alignment. The distributions of N- and C-termini in the alignment are then used to identify potential domains. The putative domains are possibly merged if there is high correlation between the participating sequences and are then used to generate true multiple alignments. Profiles based on these alignments are used with PSI-BLAST for the next round of database search, and this process is iterated to convergence to get domain definitions. Pass uses profiles of sequence counts to locate positions where there is a substantial change in sequence participation. These positions are then paired up to define domains.

1.1.5 Other methods Clustering sequence alignments (CSA) represents sequences as 0–1 vectors based on whether or not they are similar to the sequences in the databases. The sequences are then clustered by constructing an MST on the all-versus-all graph. This method does not give explicit domain definitions but may indicate possible domain families. In the work by Miyazaki et al. (2002), the amino acid composition of the protein sequence for a window of positions is used as input to train a neural network to detect linker sequences in proteins. The DGS system uses the domain size distribution and architecture of previously characterized proteins to make the most likely guess for a protein based solely on the length of the protein.

1.2 The current status

Methodology: Despite the large number of studies, the task of constructing an accurate and efficient general-purpose domain detection system that works solely on sequence information is still an open problem. While methods like SMART and TigrFam are accurate, they require careful manual inspection and provide predictions for a small subset of the sequence database. On the other side of the spectrum, methods like DOMO and ProDom are fully automatic and give predictions for nearly all proteins in the sequence database but are less accurate. In this paper, we suggest a novel approach that incorporates many of the salient features of earlier systems into a probabilistic framework that is extensible and is based on rigorous analysis of information sources in order to predict domain boundaries with high accuracy and coverage.

Evaluation: There is no fixed, universally accepted set of rules for partitioning a protein into its constituent domains. Therefore, it is hard to assess the quality of domain predictions by any of the above algorithms. In the absence of a common framework for analyzing the quality of domain predictions, the various works that we have mentioned above have relied on a variety of qualitative and quantitative evaluation criteria, external resources and manual analysis to verify domain boundaries and study the capabilities of their systems. For example, the quality of domain predictions in DOMO is analyzed by taking domain annotations in PIR (George et al., 1996) and SwissProt (Bairoch and Apweiler, 1999) as being the standards of truth and by comparing the predictions with ProDom predictions. However, their analysis is based only on a few selected examples. Others, such as Domination and Rigden’s covariance analysis, run a more extensive evaluation based on comparisons with structure-based domain definitions as in SCOP (Hubbard et al., 1999), but they did not evaluate the capabilities of other methods with this setup.

The diversity of evaluation criteria has made it impossible to compare objectively the various methods for domain prediction. Here, we propose and use a common framework to evaluate the various methods. This framework is based on using definitions from the SCOP database and, as a more rigorous subset, its intersection with the CATH database (Orengo et al., 1997) as the standard of truth. In addition, we devise scores that can be used in a uniform and unbiased fashion to evaluate the accuracy and coverage of the various methods.

The paper is organized as follows. We first describe the data set, scores and our learning methodology in detail. We then present the results of testing our method on a large collection of proteins with known structures and compare our predictions with structure-based domain definitions as well as with other sequence-based domain partitioning methods. We conclude with a few examples where our predicted domains seem to suggest a plausible alternative to manual classification.

2 METHODS

Given a query sequence, our algorithm starts by searching a large sequence database and generating a multiple alignment of all significant hits. The columns of the multiple alignment are analyzed using a variety of sources to define scores that reflect the domain-information-content of alignment columns. Information theory-based principles are employed to maximize the information content. These scores are then combined using a neural network to label single columns as core-domain or boundary positions with high accuracy. The output of the ANN is then post-processed to smooth and refine predictions while considering local information from multiple columns. Finally, we introduce the domain-generator model that uses global information about the distribution of domain sizes and sequence divergence to test multiple hypotheses, filter out positions that are incorrectly predicted as boundary positions and output the most likely partition. An overview of our method is depicted in Figure 1. We now turn to describe our method in detail.
2.1 The datasets

2.1.1 The query dataset In the absence of general rules or principles that define domain boundaries, one must rely on existing knowledge of protein domains to devise a reliable and accurate method for automatic domain detection. This knowledge, in the form of complete protein chains and their partition into individual domains, can be used to both train and test our method. One of the most extensive collections of protein domains is the one provided by the SCOP classification of protein structures (Hubbard et al., 1999). This classification has a complicated hierarchy with 7-fold classes, several hundred folds and more than one thousand protein families. It is built by the careful manual curation of Dr Alexei Murzin. The domains in this database are defined from Protein Data Bank (PDB) records (Westbrook et al., 2002). Each PDB structure is manually partitioned into the component domains based on their compactness, the contact area with other parts of the protein and resemblance to existing domains and then classified into families, superfamilies, folds and classes.

To train and test our method, we selected complete protein chains from PDB, searched the database and generated multiple alignments. About half of these alignments with their corresponding domain structure as defined by SCOP were used for training. The other half was used for testing.

Our initial data set was the set of protein sequences in the PDB database as of May 2002 with 35 184 protein chains and 11 969 non-identical sequence entries. All sequences shorter than 40 amino acids and fragments of longer sequences were eliminated, leaving 11 294 sequences. Of sequences that are more than 95% identical, only a single representative was retained, yielding a total of 4810 valid queries.

2.1.2 Alignments Each one of the 4810 queries was searched against a composite non-redundant database that contains 933 075 unique sequence entries. The database is composed of 96 different databases, among which are SwissProt, TrEMBL, PIR, PDB, DBJ, GenBank, REF, PATAA, PRF and the complete genomes of 78 organisms.
All entries that are documented as fragments (according to at least one source database) were eliminated, leaving a total of 693,912 non-fragmented entries. The alignment was created in two phases. First, the query was searched against the non-redundant database using BLAST (Altschul et al., 1997) and the related sequences were compiled into a database (a different database for each query sequence). In the second phase, the query was searched against this smaller database, using PSI-BLAST (Altschul et al., 1997) until convergence. Of these alignments, fragmented queries were eliminated and only alignments with more than 20 hits were kept. Finally, the query sequences were grouped into clusters [using the ProtoMap clustering algorithm (Yona et al., 1999) with a conservative E-value threshold], and from each group only one representative was selected (the one with the maximal number of database-aligned sequences). The final set of queries consisted of 3140 PDB sequences, with their corresponding alignments. Alignments are represented as a sequence of alignment columns, with each one being associated with one position in the seed sequence (insertions with respect to the seed sequence are processed as described in Section 2.2.3).

It is important to note that we did not try to refine the alignments by applying other multiple alignment algorithms. Our goal was to develop a tool that can take the output from a database search and immediately partition the query sequence into domains, based on this information, while tolerating noise and misaligned regions. However, an application of more sophisticated alignment algorithms can help in refining the alignment and improving the quality of the predictions.

2.1.3 Domain definitions The domain definitions are retrieved from the SCOP database, version 1.57 as of May 2002. Of the 11,969 unique entries in PDB, 9479 are listed in SCOP. After removing inconsistent entries (identical chains with different domain definitions or inconsistent lengths), we were left with 9185 entries. Of the 3140 PDB queries, 3039 were documented in this list, with the number of domains ranging from one to seven. In a final pruning step, protein chains that are <90% covered by SCOP domains are eliminated. In the final dataset, we retained all the multi-domain proteins (605) and one-fourth of the single domain proteins (576) to ensure an equal representation of both.

For each protein chain, we defined the domain positions to be the positions that are at least \( x \) residues apart from a domain boundary. Domain boundaries are obtained from SCOP definitions or where for a SCOP definition of the form \((\text{start}_1, \text{end}_1) \cdots (\text{start}_n, \text{end}_n)\) the domain boundaries are set to \((\text{end}_i + \text{start}_{i+1})/2\) as in Figure 2. All positions that are within \( x \) residues from domain boundaries are considered boundary positions. This process allows us to classify all the positions in the proteins being considered as domain or boundary positions.

2.2 The domain-information-content of an alignment column

To quantify the likelihood that a sequence position is part of a domain or at the boundary of a domain, we defined several measures based on the multiple alignment that we believe reflect structural properties of proteins and would therefore be informative of the domain structure of the seed protein. While some of these measures are more directly related to structural properties than others, none of these measures actually relies on structural information as our goal was to devise a novel technique that can suggest domain delineation from sequence information alone.

2.2.1 Conservation measures Multiple alignments of protein families can expose the core positions along the backbone that are crucial to stabilize the protein structure or play an important functional role (as in the active site or in an interaction site). These positions tend to be more conserved than others and strongly favor amino acids with similar and very specific physio-chemical properties because of structural and functional constraints.

Amino acid entropy: One possible measure of the conservation of an alignment column is given by the entropy of the corresponding distribution (Fig. 3). For a given probability distribution \( P \) over the set \( A \) of the 20 amino acids, \( P = (p_1, p_2, \ldots, p_{20})^t \), the entropy is defined as

\[
E_a(P) = -\sum_{i=1}^{20} p_i \log_2 p_i.
\]

This is a measure of the disorder or uncertainty we have about the type of amino acid in each position. In information theory terms, the entropy is the average number of bits needed to encode an arbitrary member of \( A \). For a given alignment column, the probability distribution \( P \) is defined from the empirical counts, after adding pseudo counts as described in Henikoff and Henikoff (1996).

Class entropy: Quite frequently one may observe positions in protein families that have a preference for a class of amino acids, all of which have similar physio-chemical properties. The amino acid entropy measure is not effective in such cases since it ignores amino acid similarities. An entropy measure based on suitably defined classes may capture positions with...
subtle preferences toward classes of amino acids. We tried two different classifications that are motivated by different considerations. The first classification was adopted from Ferran et al. (1994) and is based on clustering residues according to similarity scores from a statistical score matrix. The classes that are defined are hydrophobic (MILV), hydrophobic aromatic (FWY), neutral and weakly hydrophobic (PAGST), hydrophilic acidic (NQED), hydrophilic basic (KRH) and cysteine (C). The second classification is basically an attempt to group the amino acids into small chemically similar groups (Linda Nicholson, personal communication). The classes obtained as a result are sulfur (CM), simple aliphatic (AL), side-chain restrictive aliphatic (IV), aromatic (FWY), hydroxyl (ST), amide (NQ), acidic (ED), basic (KRH), proline (P) and glycine (G). This classification worked better than the first and therefore was chosen as the underlying classification for our class entropy measure.

Given the set $\mathbf{C}$ of amino acid classes and the empirical probabilities (with pseudo counts) $\mathbf{P}$, the class entropy is defined in a way similar to the amino acid entropy.

$$E_c(\mathbf{P}) = - \sum_{i \in \mathbf{C}} p_i \log_2 p_i.$$  

**Evolutionary pressure**: The class entropy measure is one possible solution to the aforementioned problem. However, it does not utilize all the prior information we have about amino acid similarities. A better entropy measure would consider the mutual information (similarity) of the amino acids. To the best of our knowledge, this problem has never been addressed directly before. A possible extension may generalize upon the results of Csiszár (citeseer.nj.nec.com). Alternatively, we suggest a measure that estimates the evolutionary pressure in an alignment column by calculating the evolutionary span, approximated by the sum of pairwise similarities of amino acids in a column. Specifically, if the number of sequences participating in an alignment column $k$ is $n$, then the span of this column is defined as

$$\text{Span}(k) = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j<i} s(a_{ik}, a_{jk}),$$

where $a_{ik}$ is the amino acid in position $k$ of sequence $i$ and $s(a, b)$ is the similarity score of amino acids $a$ and $b$ according to a scoring matrix such as BLOSUM50 (Henikoff and Henikoff, 1992).

### 2.2.2 Consistency and correlation measures

Since protein domains are believed to be stable building blocks of protein folds, it is reasonable to assume that all appearances of a domain in database sequences will maintain the domain’s integrity. However, domains may be coupled with other domains, and therefore a simple pairwise sequence alignment (or multiple pairwise alignments) will not be informative. Integrating the information from multiple sequences can generate a strong signal, indicative of domain boundaries by detecting changes in sequence participation and evolutionary divergence. We tested several different measures. These measures quantify the correlation and consistency of neighboring columns in an alignment.

#### Consistency

This simple coarse-grained measure is based on sequence counts. The measure is defined as the difference in sequence counts of a column and the average of the surrounding columns in a window of size $w$. If $c_k$ is the sequence count in position $k$, then

$$\text{Consistency}(k) = |c_k - \frac{1}{2w} \sum_{i \neq k, |i-k| \leq w} c_i|.$$  

**Asymmetric correlation**: This is a more refined measure that considers the consistency of individual sequences and sums their contributions. To measure the correlation of two columns, we first transform each alignment column into a binary vector of dimension $n$ (the number of sequences in the alignment), with 1s signifying aligned residues and 0s, gaps. Given two binary vectors $\vec{u}$ and $\vec{v}$, their asymmetric correlation (bitwise AND) is defined as

$$\text{Corr}_a(\vec{u}, \vec{v}) = \langle \vec{u}, \vec{v} \rangle = \sum_{i=1}^{n} u_i \cdot v_i.$$  

High correlation values reflect consistent sequence participation, while low correlation values signal a region of ambiguous sequence participation and possible domain boundaries (Fig. 4).

#### Symmetric correlation

The asymmetric correlation measure does not reward sequences that are missing from both positions. However, these may reinforce a weak signal based on sequence counts.
only on participating sequences. The symmetric correlation measure corrects this by using bitwise XNOR when comparing two alignment columns, i.e.

\[
\text{Corr}_s(\tilde{u}, \tilde{v}) = \sum_{i=1}^{n} \delta(u_i, v_i),
\]

where \( \delta \) is the delta function \( \delta(x, y) = 1 \iff x = y \).

To enhance the signal and smooth random fluctuations, the contributions of all positions in a local neighborhood around a sequence position are added, and all correlation measures for an alignment column are calculated as the average correlation over a window of size \( w \) centered at the column (the parameter \( w \) is optimized, as described in Section 2.4).

**Sequence termination:** Sequence termination is a strong signal of a domain boundary. However, in a multiple alignment it is not necessarily indicative of a true sequence termination. Although we eliminated all sequences that are documented as fragments from our database, the sequence may still be a fragment of a longer sequence without being documented as such. Moreover, the termination may be premature as end loops are often loosely constrained and tend to diverge more than core domain positions. These diverged subsequences may be omitted from the alignment if they decrease the overall similarity score. Therefore, the sequence termination signal may be misleading if used simple-mindedly. To reduce the sensitivity to sparse signals due to the aforementioned problems with the alignment’s reliability. For every position, we calculate right and left termination scores, based on sequences that terminate and originate from that position, respectively, by taking the sum of the log of the corresponding \( e \)-values. For example if an alignment position has \( n \) sequences, of which \( c \) terminate at that position and the \( e \)-values of the corresponding alignments are \( e_1, e_2, \ldots, e_c \), then the left termination score is defined as

\[
E_{\text{left termination}} = \log(e_1 \cdot e_2 \cdot \ldots \cdot e_c).
\]

In order to correct for premature alignment terminations, if the remaining unaligned segment of a library sequence is less than 20 residues long, we distribute the corresponding \( e \)-value for the termination across these positions as well. Finally, the left and right termination scores are smoothed over a window and then combined through multiplication (joint termination) and addition (combined termination) to get two different sequence termination-based scores (our experiments showed that these scores did better than the use of left and right termination scores for neural network training).

**2.2.3 Measures of structural flexibility** Regions of substantial structural flexibility in a protein often correspond to domain boundaries where the structure is usually exposed and less constrained. We define two different measures that may help us quantify this aspect.

**Indel entropy:** In a multiple alignment of related sequences, positions with indels with respect to the seed sequence indicate regions where there is a certain level of structural flexibility. The larger the number of insertions and the more prominent the variability in the indel length at a position, the more flexible we would expect the structure to be in that region. We define the indel entropy based on the distribution of indel lengths as

\[
E_g(P) = -\sum_i p_i \log_2 p_i,
\]

where the \( p_i \) are the various indel lengths seen at a position.

**Correlated mutations:** Another source of information about the structural flexibility of a position can be obtained from the profile of predicted contacts in a protein. For each sequence position, we count the number of pairwise contacts between residues that reside on opposite sides of that position [see also Rigden (2002)]. Minima in the profile correspond to regions where fewer interactions occur across these sequence positions, implying relatively higher structural flexibility and suggesting a domain boundary.

Contacts between residues in a protein are usually predicted based on correlated mutations. The correlated mutation score between two columns is defined as in Pazos et al. (1997). Specifically, the correlation coefficient for two positions \( k \) and \( l \) is defined as

\[
\text{Corr}_m(k, l) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{[s(a_{ik}, a_{jl}) - \langle s \rangle_k][s(a_{il}, a_{jl}) - \langle s \rangle_l]}{\sigma_k \cdot \sigma_l},
\]

where \( a_{ik} \) is the amino acid in position \( k \) of sequence \( i \) and \( s(a, b) \) is the similarity score of amino acids \( a \) and \( b \) according to the scoring matrix. The term \( \langle s \rangle_k \) is the average similarity in position \( k \), and \( \sigma_k \) is the SD. Here \( n \) is the number of sequences that participate in both columns.

To predict a contact based on a correlated mutation score, one needs a reliable statistical significance measure to discern true correlations from random coincidental regularities. To assess the statistical significance of correlated mutation scores, we calculated the correlation score for a large collection of random alignment columns.\(^3\) Based on the distribution of the random scores, we associate a \( z \)-score with each correlated mutation score. If the average correlated mutation score for random columns is \( \mu \) and the SD is \( \sigma \), then the \( z \)-score of a correlated mutation score \( r \) is defined as \( z\text{-score}(r) = (r - \mu) / \sigma \).

\(^3\)Random columns are generated by choosing a root residue at random and mutating it according to transition probabilities, derived from the BLOSUM50 matrix, to generate the other residues in the column.
alignments. We experimented with sampling of columns in an attempt to reduce the computation time but noticed that the resulting profile can be qualitatively very inaccurate. The sampling of rows on the other hand seems to have a marginal affect on the correlated mutation calculations and so we imposed a limit of 100 sequences for the columns, resorting to uniform sampling when the size of columns is bigger.

Calculating all correlated mutations is prohibitive for large alignments. We experimented with sampling of columns in an attempt to reduce the computation time but noticed that the resulting profile can be qualitatively very inaccurate. The sampling of rows on the other hand seems to have a marginal affect on the correlated mutation calculations and so we imposed a limit of 100 sequences for the columns, resorting to uniform sampling when the size of columns is bigger.

2.2.4 Residue type based measures

Physio-chemical properties of proteins may also help in predicting domain boundaries since they tend to have different characteristics around domain transition points than in domain core positions. For example, hydrophobic residues tend to cluster inside domain cores with hydrophilic residues occupying more exposed locations in a protein structure and therefore more likely to be in interdomain regions. Similarly, certain amino acids such as cystines and prolines are crucial in defining protein structure and therefore tend to occur in different frequencies in core domain and interdomain regions of a protein. The value of considering residue composition in detecting domain boundaries is also demonstrated in the work done by Miyazaki et al. (2002). In order to exploit these sources of information, we defined several measures: for hydrophobicity, for molecular weight and for the amino acids cysteine, valine, proline and glycine, all believed to be instrumental in defining protein structure. In addition, we also used the Rasmol classification of amino acids to create a set of non-redundant classes that we use as measures (acyclic [ARNDEQGILKMTV], aliphatic [AGILV], aromatic [HFWY], burried [ACILMFVWY], hydrophobic [AGILMFVWY], large [REQHILKFWY], negative [DE], positive [RHK] and small [AGS]). For each measure, the score of an alignment column is defined as the average of all residue scores, where residue scores are defined in the range 0–1. Hydrophobicity and molecular weight residue scores were adopted from Black and Mould (1991), and class scores were simply defined by the presence (score 1) or absence (score 0) of the residue in the class.

2.2.5 Predicted secondary structure information

Protein structure is often studied at the level of secondary structure. Most interdomain regions are composed of loops, while beta strands tend to form sheets that constitute the core of protein domains. Alpha helices and beta sheets in proteins are relatively rigid units, and therefore domain boundaries rarely split these secondary structure elements. Indeed, in the study by Sowdhamini and Blundell (1995), a domain delineation algorithm was developed that was based on the clustering of secondary structure units. This algorithm was applied to proteins of known structure and used the available structural information to define the secondary structure elements. However, useful information regarding the secondary structure of a protein can be obtained even when the structure is unknown. We used the neural network based program PSIPRED (McGuffin et al., 2000) to predict the secondary structure of the seed protein. The neural network confidence values in the range 0–1 were then used as alpha helix (alpha), beta strand (beta) and coiled region (coil) measures.

2.2.6 Intron–exon data

It is well known that the alternative splicing mechanism is used extensively in higher organisms to generate multiple messenger RNA (mRNA) and protein products from the same DNA strand. This mechanism raises an interesting combinatorial problem. By sampling (and sometimes shuffling) the set of exons encoded in a DNA sequence, the cell generates different proteins that share different numbers of exons.

Intron–exon data at the DNA level are believed to be correlated with domain boundaries (Gilbert and Glynias, 1993; Gilbert et al., 1997). As building blocks, domains are believed

---

Fig. 5. Predicted contact profile.

We used the correlated mutation information to design two types of scores. In the first case, we considered correlated mutation values that were larger than those in the random distribution as indicating contacts. The number of contacts across every position is then normalized by the total number of possible contacts to generate a contact profile. The other score was based on considering all the values as contacts but weighting them by the z-score to get a weighted profile. An example of a contact profile is given in Figure 5.

Beyond structural integrity, correlated mutations provide another evidence for the domain structure of a protein from an evolutionary point of view. Positions that are strongly correlated through evolution imply that the sequence in between must have evolved in a coordinated manner as one piece. As such, the sequence qualifies as a building block, and it is less likely to observe a domain boundary in between.

2 Calculating all-versus-all correlated mutations is an $O(lmn^2)$ task for an alignment of length $m$ with $n$ sequences. For a typical alignment of length 200 with 500 sequences, this means on the order of $(200 \times 500)^2 = 10^{10}$ computations. This takes roughly 3 h for our implementation on a Pentium III 1 GHz machine.
to have evolved independently. Therefore, it is likely that each domain has a well-defined set of exons associated with it. If the product protein is a multi-domain protein, we expect exon boundaries to coincide with domain boundaries.

The intron–exon data were derived from the EID database (Saxonov et al., 2000). Only genes that were experimentally determined (based on the header information) were included in our analysis (a total of 25 130 sequences, and 21 042 entries after eliminating redundancy). Each seed sequence was compared with all the EID sequences, and all significant ungapped matches were recorded. To quantify the likelihood of an exon boundary, we use a similar equation as in sequence termination. Specifically, if an alignment position has n sequences, of which c coincide with exon boundaries and the e-values of the corresponding alignments are $e_1, e_2, \ldots, e_c$, then the exon termination score is defined as

$$E_{\text{exon}} = \log(e_1 \cdot e_2 \cdot \cdots \cdot e_c).$$

### 2.3 Score refinement and normalization

Two additional steps are executed before the measures are fed into the neural network. First, they are smoothed to eliminate random local fluctuations and improve the discrimination power of the measure. The scores are smoothed by calculating the average over a window of size $w$ (the smoothing factor). This parameter is optimized to maximize the separation between the two types of positions, as described in the next section.

Second, they are normalized to a single scale. Since the different scores are measured in different units, a straightforward combination of scores may introduce a strong bias toward one or a few of them. Moreover, one would like to have comparable values for different proteins. Therefore a proper normalization is essential. To scale all measures to the same unit, we transformed every score to a $z$-score based on the distribution of scores along all alignment positions. The normalization is invoked separately for each alignment. The $z$-score does not only serve as a universal scale but also provides a measure of statistical significance for each position in the alignment, helping to locate atypical positions.

In the case of sequence termination-based scores, the intron score and the consistency score, we found that the distribution of scores is far from normal, making the use of $z$-score normalization inappropriate. In such cases, we used a threshold and linear scaling to map scores to the range $[0, 10]$.

### 2.4 Maximizing the information content of scores

To improve domain recognition, the distributions of domain positions and boundary positions (according to each of the domain-information-content measures suggested above) must be well separated. However, it is hardly ever the case that the two distributions are completely disjoint, and the parameters introduced before (the boundary window size $x$ and the smoothing factor $w$) may greatly affect the separation of these distributions.

To define the best set of parameters, we measured the statistical similarity of the two probability distributions for different sets of parameters and selected the one that maximized separation. To measure statistical similarity, we used the Jensen–Shannon (JS) divergence between probability distributions (Lin, 1991). This is a variation over the KL divergence measure (Kullback, 1959), i.e. both symmetric and bounded (unlike the KL divergence). Formally, given two (empirical) probability distributions $p$ and $q$, for every $0 \leq \lambda \leq 1$, the $\lambda$-JS divergence is defined as

$$D^{JS}_{\lambda}[p||q] = \lambda D^{KL}[p||r] + (1-\lambda)D^{KL}[q||r],$$

where $D^{KL}[p||q] = \sum_i p_i \log_2(p_i/q_i)$ is KL divergence and $r = \lambda p + (1-\lambda)q$ can be considered as the most likely common source distribution of both distributions $p$ and $q$, with $\lambda$ as a prior weight. The parameter $\lambda$ reflects the a priori information. In our case, the priors for in-domain positions $p$ and boundary positions $q$ differ markedly and $\lambda$ is set to the prior probability of in-domain positions. We call the corresponding measure the divergence score and denote it by $D^{JS}$. This measure is symmetric and ranges between 0 and 1, where the divergence for identical distributions is 0.

Two examples of score distributions are given in Figure 6. Even measures with near-identical distributions may be informative in a multi-variate model where higher level correlations can generate an effective boundary surface. Despite the low information content of some of the constituent measures, the total information content is more than the sum of the individual components due to sometimes weak correlations between measures. The optimal complex decision boundary is learned by training a neural network as described next.

The top 10 measures and their JS divergence are given in Table 1. Although better separation was obtained with individual boundary windows, the final boundary window was uniformly set to $x = 10$ (experiments with smaller window sizes decreased final prediction accuracy), and the smoothing window $w$ was set individually for each score based on the optimization of the JS divergence.

It should be noted that not all measures are independent of each other, and as expected, some are highly correlated. It is interesting to analyze the correlation between pairs of measures. The most correlated and anti-correlated pairs of measures are listed in Tables 2 and 3.

Some of these correlations are in support of what is known about sequence–structure relations in proteins. For example, proline residues enable extended chain conformations and are more likely to be seen in coiled regions. Similarly the negative correlation between buried residues and those in coils is along expected lines. In addition, we also see reassuring examples like the correlation between intron and joint termination scores and the negative correlation between alpha helix regions and insertion entropy that provide support for the relevance and correctness of our scores.
Table 1. JS divergence for top 10 scores

<table>
<thead>
<tr>
<th>Score</th>
<th>$\lambda = 0.5$</th>
<th>$\lambda = \text{core/boundary ratio}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Smoothing JS</td>
<td>Smoothing JS</td>
</tr>
<tr>
<td></td>
<td>window divergence</td>
<td>window divergence</td>
</tr>
<tr>
<td>Combined termination</td>
<td>7 0.073</td>
<td>10 0.018</td>
</tr>
<tr>
<td>Joint termination</td>
<td>7 0.055</td>
<td>10 0.014</td>
</tr>
<tr>
<td>Symmetric correlation</td>
<td>10 0.055</td>
<td>10 0.014</td>
</tr>
<tr>
<td>Proline</td>
<td>10 0.048</td>
<td>10 0.011</td>
</tr>
<tr>
<td>Weighted mutation profile</td>
<td>8 0.034</td>
<td>7 0.006</td>
</tr>
<tr>
<td>Class entropy</td>
<td>10 0.024</td>
<td>10 0.004</td>
</tr>
<tr>
<td>Coil</td>
<td>10 0.024</td>
<td>10 0.005</td>
</tr>
<tr>
<td>Introns</td>
<td>10 0.020</td>
<td>8 0.005</td>
</tr>
<tr>
<td>Glycine</td>
<td>10 0.015</td>
<td>7 0.003</td>
</tr>
<tr>
<td>Small</td>
<td>8 0.010</td>
<td>8 0.002</td>
</tr>
</tbody>
</table>

Divergence values are computed using $\lambda = 0.5 \text{ (equal prior)}$ and $\lambda = \text{core/boundary ratio}$. The JS divergence for identical distributions is 0.

Table 2. Most correlated pairs

<table>
<thead>
<tr>
<th>Scores</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydrophobicity and buried</td>
<td>0.704</td>
</tr>
<tr>
<td>Small and glycine</td>
<td>0.646</td>
</tr>
<tr>
<td>Aliphatic and buried</td>
<td>0.619</td>
</tr>
<tr>
<td>Joint termination and combined termination</td>
<td>0.607</td>
</tr>
<tr>
<td>Hydrophobicity and aliphatic</td>
<td>0.528</td>
</tr>
<tr>
<td>Coil and proline</td>
<td>0.500</td>
</tr>
<tr>
<td>Aliphatic and small</td>
<td>0.455</td>
</tr>
<tr>
<td>Molecular weight and positive</td>
<td>0.450</td>
</tr>
<tr>
<td>Aliphatic and acyclic</td>
<td>0.430</td>
</tr>
<tr>
<td>Aliphatic and glycine</td>
<td>0.416</td>
</tr>
</tbody>
</table>

Table 3. Most anti-correlated pairs

<table>
<thead>
<tr>
<th>Scores</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Molecular weight and small</td>
<td>-0.767</td>
</tr>
<tr>
<td>Beta and alpha</td>
<td>-0.747</td>
</tr>
<tr>
<td>Alpha and coil</td>
<td>-0.634</td>
</tr>
<tr>
<td>Molecular weight and aliphatic</td>
<td>-0.628</td>
</tr>
<tr>
<td>Molecular weight and glycine</td>
<td>-0.589</td>
</tr>
<tr>
<td>Acyclic and proline</td>
<td>-0.540</td>
</tr>
<tr>
<td>Buried and coil</td>
<td>-0.487</td>
</tr>
<tr>
<td>Hydrophobicity and positive</td>
<td>-0.469</td>
</tr>
<tr>
<td>Molecular weight and acyclic</td>
<td>-0.392</td>
</tr>
<tr>
<td>Positive and aliphatic</td>
<td>-0.313</td>
</tr>
</tbody>
</table>

2.5 The learning model

Each one of the measures we described in Section 2.2 captures some aspects or properties of domain transition signals. In many cases one or two measures will be significant enough, to indicate a domain boundary (see examples below). However, usually none of them is significant enough, and it is only their combination that reveals the subtle signal. To find the optimal combination, we trained a neural network over the domain information content scores. A neural network is capable of learning complex non-linear decision boundaries between categories and therefore seems to be most suited for this task [an alternative model to try would be support vector machines (SVMs)]. The inputs used were the individual scores in a position, and the output learnt is a number between 0 and 1, where 0 corresponds to a transition point and 1 to a domain. We trained networks using the Matlab neural network toolbox on a train set of 484 proteins with a validation set of 237 proteins and a test set of 460 proteins. We opted for a commonly used framework for neural network training: feed-forward networks trained using the resilient back-propagation algorithm (trainrp under Matlab) with a tangent sigmoid activation function.
Fig. 7. Selecting candidate transition points. The initial predictions are smoothed, and a set of candidate transition points is defined. This set is processed, and a final set of transition points is predicted.

There are several parameters that can affect the performance of the neural network. For example, the ratio of core to boundary positions in the training set, the window size in the input space and the output space, the set of measures used and the architecture of the network. We trained several thousand networks with different values for these parameters (the procedure is described in detail in Appendix A). In general, it was observed that there is a trade-off in performance over core positions and boundary positions. Our final collection of networks consists of 142 top-performing networks (in the sense that they are not strictly dominated by any other network), with varying levels of accuracy over core and boundary positions (see Appendix A for more details).

3 HYPOTHESIS EVALUATION

The neural networks that we trained do not take into account the predictions for neighboring positions (and for the protein as a whole) while making a prediction for a position. Thus, despite the high rate of accurate predictions for single positions, the final predictions may overly fragment proteins into domains.

To refine the initial predictions of the neural net, the following three steps are employed. First, to eliminate spurious transition points, the curve is smoothed. This way, a position is predicted as a candidate transition point only if a significant fraction of the positions around it are predicted as transition points by the neural network (this fraction can be altered as a threshold parameter to give different levels of accuracy and sensitivity as is described in Section 4). Second, for regions below the threshold, all the minima are predicted as candidate transition points (Fig. 7). The third step is the most important one. Each possible combination of candidate transition points is a possible partitioning of the protein into domains (Fig. 7). Given multiple hypotheses, i.e. alternative partitions of the query sequence into domains, we would like to find the most likely one. We experiment with two post-processing setups: the simple model and the domain-generator model. Both methods take the output of the neural network and consider all minima of the smoothed curve as suspected domain boundaries, in search for the best hypothesis (partition). We now turn to describe the two models in detail.

3.1 The domain-generator model

The domain-generator model assumes a random generator that moves repeatedly between a domain state and a linker state and emits one domain or transition at a time according to different source probability distributions. Thus, the probability of a sequence of domains is given by the product of domain-emission probabilities and the transition probabilities.

Formally, we are given a protein sequence and a multiple alignment $S$ of length $L$ and a possible partition $D$ of $S$ into
Posterior probability:

\[ P(D) = \frac{P(S|D)P(D)}{P(S)} \]

The denominator is fixed for all hypotheses, and so we are looking for the partition that will maximize the product of the likelihood \( P(S|D) \) and the prior \( P(D) \).

**Computing the prior:** To calculate the prior \( P(D) \), we have to estimate the probability that an arbitrary protein sequence of length \( L \) will consist of \( d \) domains of the specific lengths \( l_1, l_2, \ldots, l_n \). What we need to calculate then is

\[ P(D) = P[(D_1, l_1)(D_2, l_2) \cdots (D_n, l_n) \text{ s.t. } l_1 + l_2 + \cdots + l_n = L] \]

This can be estimated from the data by considering known domain partitions of proteins of length \( L \). However, the amount of data available is not enough to estimate accurately these probabilities for all possible partitions. We approximate this probability by using a simplified model; given the length of the protein, the generator selects the number of domains first and then selects the length of one domain at a time, considering the domains that were already generated.

For a partition into \( n \) domains, there are \( n! \) possible orderings of the domains and therefore the prior probability of the partition is approximated by

\[ P(D) \approx \text{Prob}(n/L) \]

\[ \cdot \sum_{\pi(l_1, l_2, \ldots, l_n)} P_0(l_1/L)P_0(l_2/L-l_1) \cdots P_0(l_{n-1}/L - \sum_{\ell=1}^{n-2} l_\ell) \]

where \( \text{Prob}(n/L) \) is the prior probability that a sequence of length \( L \) consists of \( n \) domains and \( P_0(l_i/L) \) is the prior probability to emit a domain of length \( l_i \), given a sequence of length \( L \). The term \( \pi(l_1, l_2, \ldots, l_n) \) denotes all possible permutations of \( l_1, l_2, \ldots, l_n \).

The prior probabilities \( P_0(l_i/L) \) are approximated by \( P_0(l_i) \), normalized to the relevant range \([0 \cdots L]\), and are estimated from the empirical distribution of domain lengths in the SCOP database. The empirical distribution is very noisy, sparse for domains longer than 600 amino acids and biased due to uneven sampling of the protein space, even after eliminating redundancy (Fig. 8a). To overcome the bias, we retain only one entry of the same length from each protein family (Fig. 8a). Noise and sparse sampling for domains longer than 600 amino acids are handled by running a few smoothing cycles that resulted in the distribution plotted in Figure 8b. Interestingly, the distribution obtained follows closely the extreme value distribution (see Section 4.6 for discussion).

The second term, \( \text{Prob}(n/L) \) is given by \( \text{Prob}(n/L) = \text{Prob}(n, L)/P(L) \), where \( \text{Prob}(n, L) \) is estimated by the

---

Fig. 8. Distributions of domain lengths (a) before and after eliminating bias; (b) after smoothing.
(n − 1)th order sum.

\[
\text{Prob}(n, L) = \sum_{i=1}^{L} P_0(x_1) \sum_{i=1}^{L} P_0(x_2) \cdots \\
\sum_{i=1}^{L} P_0(x_{n-1}) \cdot P_0(L = x_1 - x_2 - \cdots - x_{n-1}),
\]

and \(P(L)\) is simply given by the complete probability formula

\[
P(L) = \sum_{i=1}^{L} \text{Prob}(i, L).
\]

The extrapolated distributions for \(n = 1 \cdots 7\) are plotted in Figure 9a. It should be noted that the empirical distributions differ quite markedly from these extrapolated distributions (Fig. 9b). However, since the data are noisy, sparse and possibly biased, we consider the extrapolated distributions to be more reliable than the empirical ones. For one, note that the empirical probability for a protein to be a single domain dominates all other scenarios up to proteins of length 400 (!), while the curves meet much earlier (around 200) in the extrapolated distributions. Our observation is also supported by the quite different distributions observed in the CATH database, further deprecating the reliability of the empirical distributions. The impact of the extrapolated distributions is indeed evident in our results (see Section 4). Our model tends to predict more domains than SCOP and in many cases refines SCOP partitions into more compact substructures.

**Computing the likelihood:** To calculate the likelihood of the data given the model \(P(S/D)\), we use the probabilities of the observed scores given the domain structure as predicted by the neural net. We consider the individual domains and the transitions between domains (the linkers) as two different sources. Each source induces a unique probability distribution over the domain-information-content scores (see Section 2.2). Specifically, given the model \(D\) that partitions the sequence \(S\) into \(n\) domains and \(n − 1\) transitions \(D_1, T_1, D_2, T_2, \ldots, T_{n-1}, D_n\) that correspond to the subsequences \(s_1, t_1, s_2, t_2, \ldots, t_{n-1}, s_n\), we estimate the likelihood by

\[
P(S/D) = P(S/D_1, T_1, D_2, \ldots, T_{n-1}, D_n) \\
= P(s_1/D_1)P(t_1/T_1)P(s_2/D_2) \\
\cdot P(t_2/T_2) \cdots P(t_{n-1}/T_{n-1})P(s_n/D_n),
\]

where we already employed the assumption that the domains are independent of each other (see Section 3.3 for discussion). Each one of the terms \(P(s_i/D_i)\) and \(P(t_j/T_j)\) is a product over the probabilities of the individual positions. The probability on an individual position \(j\) in domain \(i\) is estimated by the joint probability distribution of the \(k\) features that are used in our system.

\[
P(s_{ij}/D_i) = P(f_1, f_2, \ldots, f_k/D_i).
\]

However, estimating this probability is impractical, given the amount of data we have. On the other hand, given the correlation between scores (see Section 4.2), the independence assumption for the individual scores does not hold. Therefore we adopt an intermediate approach. We start by writing the exact formulation of the joint probability distribution of \(k\) random variables \(X_1, X_2, \ldots, X_k\) using the expansion

\[
P(X_1, X_2, \ldots, X_k) = P(X_1)P(X_2/X_1)P(X_3/X_1, X_2) \cdots P(X_k/X_1, X_2, \ldots, X_{k-1}),
\]

where the random variables can be ordered in an arbitrary order. We then derive an approximation to these probabilities using first-order dependences\(^\text{5}\) and a heuristic expansion. The

---

\(^{5}\)Pair statistics can be calculated quite reliably from our dataset, but the data are too sparse to derive reliable estimates of higher-order statistics.
methodology is as follows: for each pair of random variables $X$, $Y$, we calculate the distance between the joint probability distribution and the product of the marginal probability distributions

$$DEPEN(X, Y) = \text{Dist}(P_{XY}, P_X P_Y).$$

This distance (measured either using the $L_1$ norm or the JS divergence measure) is a measure of the dependence between the two variables. The larger it is, the more dependent are the variables (one might also consider using the mutual information measure instead). We sort all pairs based on their distance and pick the most dependent one first (denoted by $Y_1$ and $Y_2$) to start the expansion

$$P(X_1, X_2, \ldots, X_k) = P(Y_1)P(Y_2/Y_1)\ldots$$

The next terms are selected based on their strongest dependence with variables that are already used in the expansion. Thus

$$Y_3 = \arg \max_Y \{\max[DEPEN(Y, Y_1), DEPEN(Y, Y_2)]\}.$$ 

Denote by $Z = \textsc{Pillar}(Y)$ the random variable that $Y$ is most dependent on (of the random variables that are already in the expansion); then, of all possible dependences involving $Y_3$, we pick $P[Y_3/\textsc{Pillar}(Y_3)]$ and add it to the expansion

$$P(X_1, X_2, \ldots, X_k) = P(Y_1)P(Y_2/Y_1)P(Y_3/\textsc{Pillar}(Y_3))\ldots$$

The procedure continues until all variables are accounted for. This heuristic attempts to minimize the errors that are introduced by relaxing the dependence assumption to a first-order dependence by maximizing the support for each random variable we introduce in the expansion. Thus, highly correlated variables affect the total probability only marginally, while under the independence assumption they might introduce a substantial error [other, alternative methods for approximating the joint probability distribution from the marginal distributions are described in Ireland and Kullback (1968) and Pearl (1997)]. Note that the expansion for domain regions can be different from the expansion for linker regions, as the source distributions differ. However, once the two expansions (for domains and linkers) are defined based on the pair statistics, the same two expansions are used for all domains and all linkers.

**Hypothesis selection:** Given a set of $N$ candidate transition points (the minima of the neural network output), our algorithm enumerates all possible combinations of transition points to form $2^N$ possible partitions (hypotheses). For each partition, we calculate the posterior probability (using our domain-generator model) and eventually output the most likely one. The whole calculation is very fast. For example, for a protein of length $L = 300$ and a set of $N = 10$ possible transition points, the algorithm will output the most probable hypothesis in a matter of minutes.

### 3.2 The simple model

In the simple model, the candidate transition points are listed in decreasing order of reliability (as measured by the depth of the corresponding minima in the smoothed curve) and considered in this order. Once a minimum is selected, all minima that are within a window of $k$ amino acids around it are rejected (where $k$ is a function of the protein length). This is a greedy approach that seems to work pretty well for many proteins. The depth of the minima is a good approximation of the overall posterior probability of the transition points $P(T_i/t_i)$, as the network essentially assigns a value $O(i)$ that indicates the network’s confidence in this position as being an in-domain position. Thus $1 - O(i)$ (the depth of the minima) is the probability that this position is a boundary position.

### 3.3 The independence index

Both our models explicitly or implicitly assume that the domains across transition points are independent. However, when searching for the best model, one should also consider the validity of this assumption and the ‘quality’ of the predicted transition points. Not only should they indicate domain boundaries but they should also justify the independence assumption over neighboring domains that we employed above.

We define the following confidence or independence index for each transition point. This index estimates the likelihood that the domains on both sides of the transition point are independent of each other:

$$\text{Confidence}(\text{transition}_i) = P(D_{i-1} \text{ and } D_i \text{ are independent}).$$

This likelihood is estimated as follows: if indeed the two domains were formed independently, then the patterns of sequence divergence should be different. By comparing the divergence patterns, one can indirectly measure the statistical similarity of the sources that generated the two domains. The divergence pattern is given by the distribution of evolutionary distances of sequences in the alignment of each domain (using the subset of $n$ common sequences). For each sequence, we approximate its evolutionary distance from the query seed sequence by counting the number of point mutations per 100 amino acids. The specific divergence pattern (the vector of $n-1$ distances) is a reflection of the statistical source that generated the domain. To quantify the likelihood that the source distributions are unique, we compute the Pearson correlation between the two divergence patterns. Zero correlation indicates two unique sources (independent domains).

To assess the quality of each individual transition point, we compute the independence index and report its statistical significance in terms of its $z$-score (computed based on the background distribution of independence indices over a large set of randomly predicted positions). These numbers are reported for each transition point in the final prediction. Thus, the user can evaluate not only the plausibility of the overall
4 RESULTS

To test our approach, we ran our system on a subset of 460 proteins that were excluded from the training set. The test set was well balanced in terms of the number of multi-domain proteins, with 222 single-domain and 238 multi-domain proteins (of which 179 are two-domain, 43 are three-domain, 13 are four-domain and three are five-domain proteins). For each of these proteins, the prediction was compared with that of SMART (Ponting et al., 1999), PFam (Sonnhammer et al., 1997; Bateman et al., 1999), ProDom (Sonnhammer and Kahn, 1994) and Tigr (Haft et al., 2001), based on the information provided by InterPro (Apweiler et al., 2001) as well as predictions from DOMO (Gracy and Argos, 1998) obtained by running BLAST searches against the DOMO database. Interpro predictions for ProDom are limited to a curated subset of ProDom, and so we also present results predicted directly by ProDom for proteins in the test set that can be matched (based on their accession numbers) to the complete ProDom database.

Since the predictions obtained from other systems are often incomplete for the seed proteins in our test set, we needed to design an evaluation procedure that would have different scores for accuracy and coverage. In addition, the predictions may disagree with SCOP on the number of domains in the seed protein. Therefore, one needs to define a procedure for associating predicted transition points with their most probable SCOP counterparts and vice versa. The simplest choice is to assign every transition point that is being considered to the closest reference transition point. Here we adopt this model and define the following four measures:

**Distance accuracy.** This measure evaluates predictions by using SCOP transition points as reference. For each seed protein, we calculate the average distance of the predicted transitions from their associated SCOP transition points. The final value that is reported is the average distance over all proteins in the test set.

**Distance sensitivity.** This measure assesses the sensitivity in detecting true domain boundaries by using the predicted transitions as reference. The average distance of SCOP transitions from the associated predicted transitions is calculated for each protein, with the value reported being the average of this distance over all proteins in the test set.

**Selectivity.** For this measure, we consider predictions that are within $x = 10$ residues of a SCOP transition as being correct with the final value reported being the percentage of predictions that are considered correct for the entire set.

**Coverage.** Analogous to accuracy, SCOP transitions that are associated with a predicted transition point within $x = 10$ residues are considered successfully predicted. The percentage of correctly predicted SCOP transitions for the entire set is reported.

Using these measures, we evaluated the results of post-processing the network output for the final set of 142 optimal networks (described in Appendix A) using both the simple model and the domain-generator model. As can be seen in Figure 10, even though the performance of none of these networks dominates that of the other, the performance after post-processing may do so. We also observe various tradeoffs.
for selectivity versus coverage based on which network we use. The choice of which network to use should depend on the application that we have in mind (and therefore the tradeoff that we would like to work with). For example, application of this method for structural genomics purposes might require high selectivity to avoid fragments that cannot fold independently. On the other hand, domain family classification programs may prefer high coverage to generate accurate subdomain families that can then be merged to get the final domain families. For the purpose of evaluation, we chose a single network for each model, as described in Figure 10.

The tradeoff curves in Figure 10 are not very smooth, and changing the tradeoff requires us to change the network and the inputs used. This setup is therefore not amenable for the construction of a flexible system where we can easily move on the tradeoff curve. We can however get a smooth tradeoff curve similar to that seen in Figure 11, for any fixed network that we choose by varying the threshold parameter (see Section 3) for the network output. This gives us the flexibility of changing the performance of the system by altering a single parameter. The curves seen in Figure 11 both have gentle cusps toward the top of the curves. Both these points correspond to a threshold of 0.5. The results reported next are obtained when setting the threshold parameter to that value.

First, we evaluated our two post-processing methods. The results are summarized in Table 4. Both methods perform almost the same, as measured by the four performance indices described above. Nevertheless, the domain-generator model has some advantages over the simple model. First, as opposed to the greedy approach of the simple model, the domain-generator model enumerates all possible hypotheses. Moreover, it provides us with a critical statistical framework for assessing alternative, competing hypotheses. The model can be used to assign a confidence value to each hypothesis, and by comparing these confidence values (between the best hypothesis and the next best hypothesis or the set of all other hypotheses), one can define a significance measure and associate it with the output hypothesis. In cases where the differences between competing hypotheses are insignificant, one might also want to consider alternative hypotheses.

A summary of the evaluation results for our method and other sequence-based methods is presented in Table 5. Our method improved significantly over all other automatic methods, outperformed only by the manually calibrated Pfam (see next section for discussion). Note that the criterion used to compute the coverage and selectivity is very strict (the agreement must be within 10 residues). One can relax this criterion by increasing the window size. This would result in a 5–10% increase in performance for both measures when using a window of 15 residues.

We also evaluated the overall consistency of the different methods. Specifically, we ask how many proteins are predicted correctly completely, both in terms of the total number of domains and their exact locations. The results are summarized in Table 6. Again, our method performed well compared with all other automatic methods. Moreover, while other methods performed well mostly over single-domain proteins, our method performs well on many multiple-domain proteins as well.
Table 6. Global consistency results

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of predictions</th>
<th>Correct number of domains</th>
<th>Number of completely correct predictions</th>
<th>Correct predictions (single domain)</th>
<th>Correct predictions (multi-domain)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our method</td>
<td>460</td>
<td>267</td>
<td>205</td>
<td>134</td>
<td>71 (35%)</td>
</tr>
<tr>
<td>HMMMPFam</td>
<td>441</td>
<td>309</td>
<td>276</td>
<td>178</td>
<td>98 (36%)</td>
</tr>
<tr>
<td>BlastDom</td>
<td>252</td>
<td>148</td>
<td>118</td>
<td>98</td>
<td>20 (17%)</td>
</tr>
<tr>
<td>BlastProDom (Complete)</td>
<td>218</td>
<td>94</td>
<td>83</td>
<td>51</td>
<td>32 (39%)</td>
</tr>
<tr>
<td>HMMSmart</td>
<td>172</td>
<td>112</td>
<td>91</td>
<td>70</td>
<td>21 (23%)</td>
</tr>
<tr>
<td>BlastProDom (Interpro)</td>
<td>123</td>
<td>83</td>
<td>75</td>
<td>63</td>
<td>12 (16%)</td>
</tr>
<tr>
<td>HMMTigr</td>
<td>51</td>
<td>23</td>
<td>21</td>
<td>20</td>
<td>1 (5%)</td>
</tr>
</tbody>
</table>

The percentages in the last column are the percentages of correct predictions of multi-domain proteins, out of all correct predictions. Among the multi-domain proteins the percentages of correctly predicted two-domain proteins, three-domain proteins, etc. remain roughly the same as their proportions in the test set.

4.1 Inclusion of structural information in prediction

When evaluating the results, one has to keep in mind that incorporation of structural information, when available, can improve the quality of predictions. Indeed, the PFam database uses this information explicitly by defining domains using the SCOP database. It is not surprising therefore, that the manually calibrated PFam performed better on the test set. Its performance, however, may not be as good over an independent dataset. In order to correct this bias, one would ideally like to generate a totally independent test set. However, since PFam is in the process of integrating all SCOP definitions to determine their domain definitions, it is hard or almost impossible to generate such a set.

Instead, we tested the effect of incorporation of structural information on our predictions. We repeated the process, this time including the SCOP sequences in the database. Thus the alignments that we generate might contain SCOP sequences of structural domains. However, these sequences are not used arbitrarily in our system to chop the proteins into domains. Rather, they add to the overall signal in each one of the constituent measures, and it is the cumulative contribution that is detected by our learning system. As a result, both sequences of unknown structures and sequences of known structures can affect the predictions. In other words, our learning system does not use the structural information explicitly, and it processes alignments that contain SCOP sequences in exactly the same way that it processes alignments that are based purely on sequences of unknown structures. The results of this procedure are summarized in Tables 7 and 8. Note the significant improvement in performance of our method. Especially notable are the significant coverage and selectivity.

4.2 Examples

The overall performance of our method shows that the model is capable of learning even subtle signals that indicate domain boundaries. Our first example is a three-domain protein that was predicted accurately for all its domains. This is the PDB protein 1g8h (chain B), 563 residues long. The protein is partitioned by SCOP into three domains that correspond to positions 2–181, 182–360 and 361–556. Our prediction suggests transition points at positions 181 and 354 (Fig. 12) within six residues from SCOP definitions. These positions are correlated with strong combined termination and insertion entropy signals. In addition there is an abundance of proline residues around positions 180 and 360, and there are class entropy spikes around positions 110, 180, 360 and 500. For comparison, PFam predicts three thiamine pyrophosphate enzyme domains at positions 2–180, 197–348 and 361–538. No predictions were available from ProDom, DOMO, SMART or Tigr.

Another example where our method predicted correctly all the domain transition points is for the protein 1g8h (chain B).
However, in this case none of the other sequence-based predictions (including PFam) was able to partition the protein correctly. This protein is 511 amino acids long, and according to SCOP it consists of three independent domains, between positions 2–168, 169–389 and 390–511 (Fig. 13). Our prediction locates domain boundaries at positions 165 and 392, within three residues from the SCOP definition. In PDB, 1gh8 is annotated as an archaeal translation elongation factor. However, a HMM search using PFam reports the main domain being an ATP-sulfurylase between positions 72 and 392. A look at the structure of the protein shows clearly that this is an unsatisfactory domain definition. Similarly ProDom (Interpro) predicts a domain between positions 37 and 393. Both Domo and Tigr make similar predictions (1–396 and 4–386) that merge the first and second domains into one large domain. No predictions are available from SMART. Detailed analysis of our system in this case reveals combined termination signals at positions 80, 180, 290 and 390 and weighted mutation profile troughs at positions 120 and 390. Peaks in insertion entropy are also seen at positions 140, 160 and 250, and an abundance of proline residues is seen around positions 260 and 390.

### 4.3 Suggested novel partitions

The list of proteins on which our method failed to correctly predict domain boundaries as defined by SCOP revealed interesting cases. Many of them raise serious questions about the validity of SCOP definitions. For example, PDB protein 1acc (735 amino acids long) is defined as a single domain in SCOP. Our analysis suggests three domains at positions 1–160, 161–586 and 587–735 (Fig. 14). As the figure illustrates, this partition seems to satisfy better the definition of a domain as a compact, independent foldable unit.

Moreover, given the distribution of domain sizes in proteins (see Section 3.1), it is not very likely to have protein domains that are longer than 700 amino acids, thus further supporting our hypothesis. For comparison, PFam detects one domain at positions 103–544 (PF03495 Clostridial Binary exotoxin B), and Domo predicts two domains at positions 1–647 and 648–735. No predictions are available from ProDom (Interpro), SMART or Tigr.

In this case, we get a clean and strong joint termination signal at positions 160 and 590 and a remarkably consistent alignment between positions 170 and 580. This signal is reinforced by other measures: the hydrophobic curve has three major troughs at 170, 290 and 570, the insertion entropy has major peaks at 180, 310 and 560 and the correlation is pretty low around 200, 280 and 590.
Another interesting example is the PDB protein 1ffv (chain E), which is 803 residues long and partitioned by SCOP into two domains defined by the positions 1–146 and 147–803. Our method predicts four domains at positions 1–141, 142–426, 427–591 and 592–803 (Fig. 15). While our prediction agrees with SCOP in defining the first domain, it partitions the second domain further into three subunits. Analysis of the protein structure indicates that the second domain predicted by our method does define a distinct, reasonably compact structural domain. In addition, while the third and fourth domains are intertwined in space, there seems to be a clear symmetry in their construction, suggesting the possibility that they arose as a result of duplication. Interestingly, CATH also partitions the protein into four domains, though the definitions are much more complicated (domain1, 7–141 and 210–306; domain2, 142–209 and 307–383; domain3, 484–649; domain4, 440–483 and 650–803). The signals that our method gets for predicting the additional domain boundaries at positions 426 and 591 are quite strong. In addition to a strong neural-network output, we also observe strong sequence termination and class entropy signals around all three positions.

In both cases, SCOP definitions might be inaccurate because of the lack of structural information to support the existence of these domains. SCOP domains are defined as recurrent structural subunits, and in the absence of other copies of these domains the proteins are left untouched. Our analysis indicates that had the structures of related proteins been resolved, such evidence would have become available. In the presence of such strong signals based on sequence information, it is clear that the domain structure of proteins cannot be determined based on structural information alone.

4.4 Analysis of errors

Our method does fail in cases where signals are misleading. This usually seems to happen when the domain definition for the protein is complicated by the unusual structure and topology of the protein. One such case is for the beta-barrel protein 1qkc. It is classified by SCOP as a single-domain protein while our method predicts three domains defined by the positions 1–256, 257–394 and 395–725 (Fig. 16). In comparison, PFam predicts a domain between positions 615 and 725 and Domo predicts two domains at positions 21–337 and 338–725. In general, beta-barrel proteins are considered hard test cases, even for structural domain classifiers. While our predictions clash with the standard definition of classifying the entire barrel structure as one domain, it is interesting to note that both boundary predictions made by our method
are in looped regions, even though it is much more likely that a prediction lies in a beta-strand region (based on the beta to loop ratio). In addition, while it is not clear if the domains predicted by our method are the correct pieces, it seems quite plausible that the beta-barrel structure evolved by the fusion of two or more barrel pieces. The domain boundary predicted by DOMO also lends some support to our prediction. Further investigation from a biological perspective of the pieces that we identify as domains may help prove or disprove this hypothesis.

Another unusual case is the PDB protein 1i6v, which is 1118 residues long. SCOP classifies this protein as a single-domain protein. Our method partitions the protein into four domains defined by the positions 1–220, 221–513, 514–830 and 831–1118 (Fig. 17). As can be seen from the rasmol ribbons image, this protein is highly unstructured and has a complicated topology. The domains defined by our method do not partition the protein into clean, structurally distinct units. However, they do indicate that 1i6v is probably not a single-domain protein. Our predictions are supported by significant confidence index values (see Section 3.3) as well. The length of the protein is another factor that suggests that this protein is multi-domain. It is possible that some of the domains in 1i6v are non-continuous, further complicating domain prediction.

We believe that many of the ‘errors’ will be resolved as more structures are solved and SCOP definitions are refined. In some cases, the situation will require a more precise definition of what a domain is. Finally, an increase in sequence data and design of more sophisticated measures employing additional sources of information will help to improve predictions.

4.5 Consistency of domain predictions

Our gold standard so far was the SCOP database of protein domains. The domains in this database are defined manually, based on visual inspection of protein structures; however, there is no assurance that the definitions are indeed accurate and correspond to the ‘true’ definitions. Since no quantitative rules or principles are used, different points of view might lead to somewhat different domain definitions.

To assess the stability and accuracy of our domain prediction algorithm, we tested it on another structure-based domain classification system, CATH (Orengo et al., 1997), which combines sequence analysis with structure comparison algorithms to determine structural domains. Of the 238 multi-domain proteins in our test set, we were able to map 158 proteins to release 2.4 of CATH⁶. Of the 222 single-domain proteins in the test set, we were able to map almost all (217) to CATH. Of the 158 multi-domain proteins, 48 contained discontinuous domains (according to CATH) that cannot be predicted with our method (see discussion below) and therefore were eliminated. To keep the numbers of single- and multi-domain proteins balanced, we sampled 110 proteins from the list of single-domain proteins to get a new test set of 220 proteins.

We repeated our performance evaluations over this set of 220 proteins using the CATH definitions as the standard of truth. The results are given in Table 9. As can be seen from the first line of the table, while CATH and SCOP are in pretty good agreement, they do differ in some cases. Based on comparison with the results in Table 5, we can see that the performance of our method is stable across CATH and SCOP. The stability of our results therefore indicates that our methodology learns a more general concept of domains. In contrast, we see that the performance of PFam on CATH is not as good as on SCOP. This could be explained by the fact that PFam definitions are often guided by SCOP definitions.

We studied example cases where our predictions were different from those of CATH. We found that in general in such cases CATH differs from our method (as well as from SCOP) because of its tendency to assign small structural fragments from one sequence domain to another based on structural compactness considerations. An example of such a situation is the protein 1ekx (chain A), which is 311 residues long. SCOP defines two domains, the first one between positions 2 and 151 and the second between positions 152 and 311 (Fig. 18). Our method predicts one transition point at position 151, in excellent agreement with the SCOP definition. The predictions from PFam (8–150, 153–305) and Prodom (7–150, 157–306) also agree with this definition. CATH defines the first domain as

---

⁶Based on the PDB identifiers, we were able to map most of the proteins (197 out of 238), but since CATH uses the ATOMRES records while we use the SEQRES records of the PDB files, there were some discrepancies (gaps, and length mismatch between ATOMRES and SEQRES records) that deemed some files unusable for testing.
as a combination of two fragments 1–133 and 292–310 and a second domain at positions 134–291. This results in a fragment of an alpha helix being assigned to the first domain based on compactness considerations alone.

The inconsistency with our method is not surprising as our definition of a domain is evolutionarily motivated. Our model assumes that protein domains are ancient and evolutionarily conserved sequence fragments that have emerged as protein building blocks during evolution. This does not cover all possible domain definitions. Multiple studies showed that in practice the structural arrangement of proteins can form compact substructures that are sequence discontinuous. However, such sequence discontinuous domains need accurate structural information to delineate them correctly, and it is not clear if it is possible to detect these domains based on sequence information alone. In the absence of clear evolutionary evidence supporting this assignment, it is also not clear how to translate such definitions to our domain definitions. Moreover, the signals, if they exist, might be different from those for continuous domains, and to learn these signals would require designing a different learning system. These issues make the identification of discontinuous domains a harder and possibly orthogonal problem to the one that we tried to solve in this study.

4.6 The distribution of domain lengths

We were intrigued by the fact that the distribution of domain lengths follows closely the extreme value distribution (EVD), as in Figure 8b. This distribution has been studied extensively in the context of sequence similarity (Karlin and Altschul, 1990; Dembo and Karlin, 1991) and has been used by software packages such as BLAST (Altschul et al., 1997) and FASTA (Pearson and Lipman, 1988) to associate statistical significance measures (e-values) with similarity scores. However, its
appearance in the context of domain lengths is surprising and suggests the presence of an underlying maximization process. These fairly ancient domains (as the traces of sequence matches indicate) are perhaps the earliest form of fully functional proteins. Their formation is the result of a random process that puts together shorter coding elements. One possible explanation for the EVD is the fact that this random process of generating a protein domain favored longer coding regions, as is explained next.

The random process itself is carried at the DNA level and manifests itself in events such as duplications and insertions. However, the outcome of such a process is likely to end up with a non-transcribable sequence of DNA. For a new gene to be created, several more conditions must be met. For one, a start and a stop codon must be present. Second, a promoter or a DNA-binding site should exist just before the start codon to enable transcription initiation. The likelihood of these two conditions being met increases as the length of the elements used in the construction (the subsequences that are being duplicated or inserted) increases. Therefore, this process creates inherently a bias toward longer fragments. Thus, if we consider each change that the DNA undergoes due to random mutations and insertions as a random variable (localized to that region where the change occurs), then of all these random processes, the longest ones are those that are more likely to survive and eventually be transcribed and translated into new domains. Thus, this process is implicitly, by natural selection, a maximization process, and the END follows. We hypothesize that in the beginning, most duplications and insertions introduced fairly short elements that were transformed into local secondary structures. With time, longer domains were formed by insertions and duplications that put a few of these motifs adjacent to each other. Since it is more likely that a random copy will include a promoter site as the length of that copy increases, this process led to the formation of longer and longer domains. Later on in the course of evolution, complete domains might have been copied or moved to form complexes or multi-domain proteins.

5 DISCUSSION

In this paper, we presented a novel method for detecting the domain structure of a protein from sequence information alone. Our method utilizes the information in sequence databases and starts by comparing the query sequence with all the sequences in the database. The search generates a multiple alignment, and the alignment is processed fully automatically in search of domain transition signals. There are several novel elements in our method. First, our method uses multiple scores. Some of the scores we designed are variations on measures that were suggested in earlier studies (e.g. sequence participation and correlation scores were used in DOMO, ProDom and PASS, and correlated mutations were used in Rigden’s work). However, we introduce many novel scores based on the analysis of basic sequence properties or predicted properties, scores that are calculated from multiple alignments and scores that are extracted from external resources such as intron–exon data. Second, we use information theory principles to optimize the scores and select the subset that maximizes the domain information content. Third, a neural network is trained to learn a non-linear mapping from the original scores to a single output. Finally, a probabilistic domain-generator model is developed to assess multiple hypotheses and predict the most likely one. Unlike local or heuristic methods that employ a greedy search through the hypothesis space, our model enumerates exhaustively all possible partitions of the protein into domains until it finds the optimal one. This multi-stage system is not only robust to alignment inaccuracies, but can also tolerate partial information. It can be extended and generalized to include other types of scores. Most importantly, our method suggests for the first time a rigorous model that can test all possible hypotheses and output the one that is most consistent with the data. We also developed an evaluation framework that will hopefully provide a clearer understanding of the strengths and weaknesses of the algorithms that have been designed so far and thus aid in the design of better algorithms. Moreover, our domain-generator model can associate a statistical significance score for every hypothesis, thus enabling us to compare different hypotheses by the same method or even different hypotheses by several different methods.

We trained and tested our method on what is considered to be the gold standard in protein structure classification, the SCOP database of protein domains. Our method performed very well compared with all other methods currently available while being fully automatic. One should keep in mind that SCOP is a man-made classification and the definitions of domains do not necessarily conform with ‘nature’s definitions’. Indeed many of our supposed errors seem to make sense when inspected visually. Moreover, SCOP might be inaccurate near domain boundaries, as the selection of the actual transition point is quite arbitrary. Our method provides a rigorous and accurate way to predict not only the domain structure but also the most likely transition points and can be used to augment or guide predictions based on structural data.

The utility of our tool goes beyond simple structural analysis of proteins. It can help in predicting the complete 3D structure of a protein as the task can be divided into smaller tasks, given the predicted domain structure of the protein. It can have significant impact on structural genomics efforts. The high-throughput structural determination of proteins is more likely to succeed when the proteins are broken into smaller, structurally stable units. Using our model to predict domain boundaries can help in that aspect too. Finally, it is essential for the study of proteins’ building blocks and for functional analysis.

There are several variations to the model described here that we consider introducing in the future. Although our algorithm
is not overly sensitive to alignment accuracy, obviously better multiple alignment algorithms are expected to improve the performance. Since the system uses the domain-generator model to process hypotheses, it is less sensitive to the exact details of the learning system; however, replacing the neural network with another learning system (such as SVMs) might also improve performance slightly. Another possible improvement is the integration of a weighting scheme into the multiple alignment. Currently all sequences are weighted equally. However, due to the biased representation of protein families in sequence databases and the nature of sequence comparison algorithms, diverged sequences that might provide us with crucial information about domain boundaries are usually under-represented in these alignments. To eliminate this bias, one should decrease the weight of highly similar sequences and increase the weight of highly diverged sequences. Preliminary attempts in that direction [implementing the schema described in Henikoff and Henikoff (1994)] did not show a significant improvement; however, the results are not conclusive. Hopefully, these variations will further fine-tune the performance of our system.

Finally, our method can be extended easily to include structural information to aid in the process of domain prediction. All it takes is to include these sequences in the alignment. If the learning system recognizes a strong signal (e.g. sequence termination) that is consistent with other sequences of unknown structure, a prediction will be made that is in agreement with the structural information. This approach can help in unifying manual expert-based approaches with more rigorous information-content-based methods to produce more reliable predictions of domains.

ACKNOWLEDGEMENTS

This work is based on earlier work, ‘A multi-expert system for the automatic detection of protein domains from sequence information’, in the proceedings of RECOMB, ACM 2003. G.Y. is supported by the National Science Foundation under Grant No. 0133311.

REFERENCES


APPENDIX A—NEURAL NETWORK TRAINING

There are various parameters that can influence the performance of the neural networks. First, since our training set is composed largely of core-domain positions, the neural network is biased toward learning these positions well. In order to circumvent this bias, we used only a sampling of the core-domain positions. Various choices of the ratio of core to boundary columns in the training set give various tradeoffs in the predictive power of core and boundary positions in a test set, and so we experimented with this ratio as a parameter in our system. Second, since a domain transition point is not singular we also tried to learn more complex networks that map multiple inputs (several positions along the sequence) to multiple outputs. Our preliminary investigations showed that using multiple outputs always decreased performance, and so we restricted ourselves to varying the input window size. Third, while in theory using all the measures that we designed to train the network should be optimal, in practice a smaller set of inputs can decrease the search space for the neural network training system and thus improve performance by a crucial role in how well it learns a function. We restricted ourselves to networks with two hidden layers (as in theory this is enough to model any function) and varied the sizes of the first and second hidden layers of the network.

We varied the above set of parameters in the ranges specified in Table 10. In choosing the features for the network, we tried two different strategies. In the first case, we sorted the set of 22 measures in the order of their JS divergence score (largest to smallest) and chose the various measures as features in that order. This framework allows us to select the best individual features but is not guaranteed to produce the set that would be optimal when combined together. As an alternative we took the approach of selecting the principal components of the vector space defined by the measures, sorted in the order of their eigenvalues (largest to smallest) as features in that order. This approach has the advantage that addition of more components is expected to improve the performance of
Prediction of protein domains from sequence

Fig. 19. Performance of networks (on test set) (a) trained using the scores as features and (b) trained using principal components as features.

Fig. 20. Performance of networks (on test set) due to variation in (a) ratio of core to boundary columns, (b) number of features, (c) input window size and (d) network size.

Table 10. Ranges for parameters in network training

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core–boundary ratio</td>
<td>0.4, 0.8, 1.2, 1.6</td>
</tr>
<tr>
<td>Number of features</td>
<td>1, 2, 4, 7, 10, 15, 22</td>
</tr>
<tr>
<td>Input window size</td>
<td>1, 5, 9, 13, 17</td>
</tr>
<tr>
<td>Size of first layer</td>
<td>0, 5, 10, 15, 20, 25, 30</td>
</tr>
<tr>
<td>Size of second layer</td>
<td>0, 5, 10, 15, 20, 25, 30</td>
</tr>
</tbody>
</table>
a reasonably smooth tradeoff curve between prediction accuracy on core and boundary columns and defines distinct regions of the curve as seen in Figure 20. Increasing the number of features seems to improve the overall performance of the networks, but after the top 10 measures have been used the improvement is negligible (similar behavior is seen when we use the principal components as features). Increasing the input window size does not lead to an overall increase in performance. In fact, the performance seems to decrease slightly with larger window sizes (leading to networks with higher accuracy on core positions but lower accuracy for boundary positions). Finally the results seem to be remarkably independent, in an overall sense, of the size of the network as can be seen in Figure 20. We also tried combining the predictions of the top performing networks (bagging), however without any apparent significant improvement in performance.

The predictions of the neural network in our system are further post-processed (see Section 3) to produce the final predictions. As a result the choice of the network that will optimize the overall performance of the system is not obvious. In addition, there is a tradeoff between the accuracy and coverage of domain boundary predictions (see Section 4). To resolve the question of which neural network to use, we start by pruning our set of networks to only those networks that are not strictly dominated by any other network in terms of network performance (this corresponds to the points on the outer boundary of the curve in Figure 19). Since the performance for the principal component-based networks is similar to the performance of the networks that use the scores as features, we retain only the 142 networks that are trained on the scores. Some of the representative points in this set are presented in Table 11. We continue the discussion of the appropriate network to choose in Section 4.

### Table 11. A sample from the set of selected networks

<table>
<thead>
<tr>
<th>Number of features</th>
<th>Input window size</th>
<th>Core–boundary ratio</th>
<th>Size of first layer</th>
<th>Size of second layer</th>
<th>Percentage of correct core predictions</th>
<th>Percentage of correct boundary positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>9</td>
<td>0.4</td>
<td>20</td>
<td>15</td>
<td>0.04</td>
<td>0.98</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0.4</td>
<td>25</td>
<td>30</td>
<td>0.23</td>
<td>0.95</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0.4</td>
<td>20</td>
<td>30</td>
<td>0.39</td>
<td>0.90</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>0.8</td>
<td>15</td>
<td>5</td>
<td>0.56</td>
<td>0.80</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>0.8</td>
<td>5</td>
<td>30</td>
<td>0.63</td>
<td>0.76</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0.8</td>
<td>30</td>
<td>25</td>
<td>0.70</td>
<td>0.70</td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>1.6</td>
<td>25</td>
<td>5</td>
<td>0.81</td>
<td>0.57</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>1.6</td>
<td>5</td>
<td>10</td>
<td>0.88</td>
<td>0.43</td>
</tr>
<tr>
<td>2</td>
<td>17</td>
<td>1.6</td>
<td>5</td>
<td>30</td>
<td>0.91</td>
<td>0.30</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>0.4</td>
<td>25</td>
<td>30</td>
<td>0.96</td>
<td>0.20</td>
</tr>
</tbody>
</table>