SMURFLite: combining simplified Markov random fields with simulated evolution improves remote homology detection for beta-structural proteins into the twilight zone

Citation

As Published
http://dx.doi.org/10.1093/bioinformatics/bts110

Publisher
Oxford University Press

Version
Final published version

Accessed
Fri Mar 29 00:26:35 EDT 2019

Citable Link
http://hdl.handle.net/1721.1/110175

Terms of Use
Creative Commons Attribution-NonCommercial 3.0 Unported licence

Detailed Terms
http://creativecommons.org/licenses/by-nc/3.0/
SMURFLite: combining simplified Markov random fields with simulated evolution improves remote homology detection for beta-structural proteins into the twilight zone

Noah M. Daniels, Raghavendra Hosur, Bonnie Berger, Lenore C. Cowen

1Department of Computer Science, Tufts University, Medford, MA 02155 and 2Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

ABSTRACT
Motivation: One of the most successful methods to date for recognizing protein sequences that are evolutionarily related has been profile hidden Markov models (HMMs). However, these models do not capture pairwise statistical preferences of residues that are hydrogen bonded in beta sheets. These dependencies have been partially captured in the HMM setting by simulated evolution in the training phase and can be fully captured by Markov random fields (MRFs). However, the MRF can be computationally prohibitive when beta strands are interleaved in complex topologies. We introduce SMURFLite, a method that combines both simplified MRFs and simulated evolution to substantially improve remote homology detection for beta structures. Unlike previous MRF-based methods, SMURFLite is computationally feasible on any beta-structural motif.
Results: We test SMURFLite on all propeller and barrel folds in the mainly-beta class of the SCOP hierarchy in stringent cross-validation experiments. We show a mean 26% (median 16%) improvement in area under curve (AUC) for beta-structural motif recognition as compared with HMMER (a popular HMM method), and a mean 10% improvement in area under curve for beta-structural motif detection for beta-structural proteins into the twilight zone.

Availability and implementation: A webserver that runs SMURFLite is available at http://smurf.cs.tufts.edu/smurflite/. For this reason, many have suggested that evolution and Smith et al. (1999) that more powerful Markov random fields (MRFs) be used. MRFs employ an auxiliary dependency graph which allows them to model more complex statistical dependencies, including statistical dependencies between hydrogen bonded residues that are hydrogen bonded in beta sheets.

However, as the dependency graph becomes more complex, major design difficulties emerge. First, the MRF becomes more difficult to train. Second, it quickly becomes computationally intractable to find the optimal-scoring parse of the target to the model.

We have built a fully automated system, SMURFLite, that combines the power of MRFs with Kumar and Cowen’s simulated evolution (Kumar and Cowen, 2010) (which offloads information about pairwise dependencies in beta sheets into new, artificial training data). In order to build the first MRF models that are computationally tractable for all beta-structural proteins, we use a dependency graph which allows them to model more complex statistical dependencies, including statistical dependencies between hydrogen bonded residues that are hydrogen bonded in beta sheets.

To whom correspondence should be addressed.

© The Author(s) 2012. Published by Oxford University Press.

Permission is granted for reprints for the purpose of scientificaviation experiments. We show a mean 26% (median 16%) improvement in area under curve (AUC) for beta-structural motif recognition as compared with HMMER (a popular HMM method; Eddy, 1998) and a mean 33% (median 19%) improvement as compared with RAPTOR (a well-known threading method) and even a mean 18% (median 10%) improvement in AUC over HHpred (a profile-profile HMM method), despite HHpred’s use of extensive additional training data. We demonstrate SMURFLite’s ability to scale to whole genomes by running a SMURFLite library of 207 beta-structural SCOP superfamilies against the entire genome of Thermotoga maritima, and make over a 100 new fold predictions.

Availability and implementation: A webserver that runs SMURFLite is available at http://smurf.cs.tufts.edu/smurflite/. For this reason, many have suggested that evolution and Smith et al. (1999) that more powerful Markov random fields (MRFs) be used. MRFs employ an auxiliary dependency graph which allows them to model more complex statistical dependencies, including statistical dependencies between hydrogen bonded residues that are hydrogen bonded in beta sheets.

However, as the dependency graph becomes more complex, major design difficulties emerge. First, the MRF becomes more difficult to train. Second, it quickly becomes computationally intractable to find the optimal-scoring parse of the target to the model.

We have built a fully automated system, SMURFLite, that combines the power of MRFs with Kumar and Cowen’s simulated evolution (Kumar and Cowen, 2010) (which offloads information about pairwise dependencies in beta sheets into new, artificial training data). In order to build the first MRF models that are computationally tractable for all beta-structural proteins, we use a dependency graph which allows them to model more complex statistical dependencies, including statistical dependencies between hydrogen bonded residues that are hydrogen bonded in beta sheets.

To whom correspondence should be addressed.

© The Author(s) 2012. Published by Oxford University Press.

Permission is granted for reprints for the purpose of scientific
2 METHODS

2.1 Review of SMURF MRF framework

SMURF and SMURFLite rely on training data in the form of a multiple structure alignment with beta strand annotation. This alignment is created using the Matt program. Beta strand annotation is done on a structure-by-structure basis, where the beta-strand residue pairing is determined using the same algorithm implemented by the RaptorX and Molnar-White visualization program. A post-processing step annotates those beta-strand residues that appear in more than half the structures in the alignment as conserved. As gaps in beta strands would complicate training, this post-processing step makes beta-conserved template strands contiguous in the alignment exactly as in [Menke et al. 2010]. The result at this stage is a sequence alignment (resulting from the Matt structural alignment) with conserved beta-strand pairs annotated according to the residue positions and conformation (buried or exposed to solvent).

The pairwise probability portion of the MRF is based on the beta probability tables that were computed by collecting a set of amphipathic beta sheets from the Protein Data Bank (PDBe: [Reese et al. 2009]) and tabulating the frequencies of pairs of hydrogen-bonded residues in two tables, one for buried residues and one for residues exposed to solvent [Menke et al. 2009]. For each residue position, the most likely conformation (buried or exposed) is chosen based on whether that residue pairing is most probable from the buried or exposed beta-pairing tables.

Given a trained MRF, SMURF and SMURFLite align a query sequence to the MRF. The query phase of SMURF and SMURFLite computes the alignment of the sequence to the states of the MRF that maximizes the combined score:

\[ \log (\text{HMM score}) + \log (\text{pairwise score}) \]

In this combined score, the HMM score is the conditional probability of observing the sequence given the HMM portion of the model, and the pairwise score is the conditional probability of observing the paired beta-strand components of the sequence given the beta-pair portion of the model. Let the sequence have residues \( r_1, r_2, \ldots, r_n \) and the MRF have match states \( m_1, m_2, \ldots, m_k \), deletion states \( d_1, d_2, \ldots, d_l \) and insertion states \( i_1, i_2, \ldots, i_m \). Suppose that \( r_i, r_j \) and match states \( m_k, m_l \) have been assigned. Then, the probability of assigning \( r_{l+1} \) to the next match state \( m_{k+1} \) is:

\[ P[r_{l+1}, m_{k+1} | \text{HMM}][m_k, m_l] \]

transition \( [u_{j-1}, m_k] \)

strand \([j, l+1, m_{k+1}]\)

where \( r_{l+1} \) can be either \( d_{l+1}, i_{l+1} \) or \( m_{l+1} \) depending on whether the current state is a deletion, insertion or match state. When the current state is a match state, the SMURFLite template replaces the transition \([u_{j-1}, m_k] \) term with a value of 1. The \( \beta \) strand component set to be identically 1 unless the particular match state \( m_k \) participates in a beta strand that is matched with a state \( m_l \) earlier in the template. This component is the primary difference between our MRF and an ordinary HMM [Menke et al. 2009].

SMURFLite computes the maximum score of a sequence using multidimensional dynamic programming on the MRF. This dynamic programming resembles the classic Viterbi algorithm [Viterbi, 1967] used on HMMER’s ‘plan7’ HMMs, except that some states are beta-strand states, which are required to be match states, and which are paired with other beta-strand nodes in the model. Because the pairwise component of the score can only be calculated for a given MRF node once it is determined what residue occupies the paired MRF node earlier in the sequence, each time the dynamic programming reaches a state in the MRF that corresponds to the first residue of the first beta strand in a set of paired beta strands, we need to keep track of multiple cases, depending on what residue in sequence is mapped to that state. SMURFLite uses a multidimensional array to memoize these possible subproblem solutions. A maximum gap size is set to the longest gap seen in the training data plus 20, for computational efficiency. When paired beta strands follow each other in sequence with no interfering beta strands between them, the number of dimensions in the table for the dynamic programming is directly proportional to the maximum gap length. Let us call the last MRF state for the first of every pair of beta strands a ‘split’ state and the first MRF state for the second of that pair a ‘join’ state. Then, at every split state, the number of dimensions of the dynamic program will be multiplied by the maximum gap length, because the dynamic program must
keep track of scores for each possible sequence position (up to the maximum gap length) that could be mapped to that state. At the corresponding join state, the number of dimensions will be reduced by the maximum gap length, because the scoring function can calculate all the pairwise probabilities of placing that residue into the join state, and thus simply take the maximum of all ways to have placed its paired residue into the split state. However, when other beta strands are interleaved, the dynamic program must open additional multidimensional tables before clearing the previous ones from memory. Thus, the number of elements in the multidimensional table is never more than the sequence length times the maximum gap length raised to the interleaving number power.

2.2 Datasets

From SCOP (Murzin et al. 1995), version 1.75, we chose the folds '5-bladed Beta-Propellers', '6-bladed Beta-Propellers', '7-bladed Beta-Propellers' and '8-bladed Beta-Propellers'. We also chose superfamilies from all of the mostly-beta folds containing the word 'barrel' in their description, whether open or closed, restricted to those superfamilies comprising at least four families (in order to facilitate leave-family-out cross-validation). These superfamilies were: 'Nucleic acid-binding proteins' (50249), 'Translation proteins' (50447), 'Trypsin-like serine proteases' (50494), 'Barnase-like endogluca nases' (50685), 'Cyclolphilin-like' (50891), 'Sm-like ribonuclease proteins' (50182), 'PDZ-domain-like' (50136), 'Prokaryotic SH3-related domain' (82057), 'Tadto/PWW/PMB' (63748), 'Electron Transport accessory proteins' (50990), 'Translation proteins SH3-like domain' (50104), 'Lipocalins' (50814) and 'FMN-binding split barrel' (50475). Of these, we removed the superfamilies 'Lipocalins' as 1-of-1 and 'FMN-binding split barrel' (50475). Of these, we removed the superfamilies 'Lipocalins' as 1-of-1 and 'FMN-binding split barrel' (50475).

Thus, we obtained an aligned set for SMURFLite training in which all superfamilies containing the word 'barrel' in their description, whether open or closed, were included. Each superfamily was left out, a training set was established from the protein chains in the corresponding superfamilies, with duplicate sequences removed. An HMM (in the case of HMMER and HHPre d) or MRF (in the case of SMURFLite and SMURFLite) was trained on the training set. Protein chains from the left-out family were used as positive test examples. Negative test examples were protein chains from all other superfamilies in SCOP classes 1, 2, 3 and 4 (including other barrel superfamilies), indicated as representatives from the nr-PDB (Berman et al. 2000) database with non-redundancy set to a BLAST E-value of 10–7.

Each test example was aligned to the trained HMM (from HMMER and HHPre d) and MRF, and was also threaded, using RAPTOR, against each individual chain in the training set (RAPTOR parameters are discussed below). The score reported for HHPre d and HHPre d was the output HMM score, and the score reported for SMURFLite and SMURFLite was the combined HMM and pairwise score from the MRF. For RAPTOR, the score reported for a test example was the highest score from all the scores resulting from threading that test example onto each chain in the training set. For each threading set, the scores for each method were collected and a ROC curve (a plot of true positive rate versus false positive rate) computed. We report the area under the curve (AUC) statistics from this ROC curve (Konig and Pongor 2002).

2.4 P-values

SMURFLite computes the P-value for an alignment similarly to HMMER, using an extreme value distribution (EVD) (Menke et al. 2008). An EVD is fitted to the distribution of raw scores over a random sampling of 5000 protein chains from across the SCOP hierarchy. The P-value is then simply computed as 1−cdf(s), where cdf is the cumulative distribution function for the EVD.

2.5 SMURFLite augmented training data

Kumar and Cowan (2009) showed that ‘simulated evolution,’ augmenting limited training data with additional sequences produced by mutating the original sequences, improved the performance of HMMER at recognizing the same-superfamily level of homology. Kumar and Cowan (2009) used two types of simulated evolution: point-wise and pairwise. Here, we add only pairwise mutations based on beta-strand pairings, as we expect long-range interactions between beta strands to be highly conserved across similar structures. We postulated that the elimination of the beta-strand pairs SMURFLite must disregard because of computational complexity might be compensated for by augmenting the training data with artifical sequences based on likely mutations in those paired beta strands. This training data augmentation comes at insignificant runtime cost and is done before beta-strand pairs are removed from the template (but after their interleave number has been identified, where we define interleave number next below). The mutation frequencies are given by the Betaswap and SMURF (Bradley et al. 2009) authors. Beta-strand probability tables. Using the same algorithm as Kumar and Cowan (2009), we generate 150 new artificial training sequences from each original training sequence. For each artificial sequence, we mutate at a 50% mutation rate per length of the beta strands. The parameters 150 and 50% were recommended by Kumar and Cowan (2009), we also evaluated 10% mutation rate (a secondary peak according to their work) and performance was slightly worse (data available from the authors).

2.6 SMURFLite simplified random field

SMURFLite trains a MRF on a template built from a multiple structure alignment. Beta strands in the aligned set of structures are found by the program SmurfPreparse which is part of the SMURF package (Menke et al. 2009) and HAMMER. The program not only outputs the positions of the consensus beta strands in the alignment, it also declares a position buried or exposed based on which of the two tables is the best fit to the amino acids that appear in that position in the training data. SMURFLite then assigns an interleave value to each beta-strand pair, as follows: any pairwise
interaction between beta strands whose interleave value equals or exceeds the SMURFLite threshold is removed from the training data.

Consider three beta strands: A, B and C. Suppose strand A interacts with strand B and the (A,B) pair has an interleave value of 4, whereas strand B also interacts with strand C and that (B,C) pair has an interleave value of just 1. With a SMURFLite threshold of 2, the (A,B) pair would be discarded, but the (B,C) pair would remain in the training data. Thus, interleave numbers are properties of pairs of beta strands; a beta strand may be involved in multiple pairings, each of which may have a distinct interleave value. Discarding beta-strand pairs whose interleave value equals or exceeds the threshold guarantees that the MRF will have no-beta-strand pairs greater than or equal to that threshold, and thus bounds the computational complexity, which is exponential in the maximum interleave value found in a training template.

Note that SMURFLite with an interleave threshold of 0, which will discard all beta-beta pair-strand information, is simply an HMM.

2.7 HMMER implementation

SMURFLite was tested against HMMER version 3.0α2 with the ‘–seqZ 1’ and ‘–seqZ 1000’ options applied to hmmssearch, and the ‘–symfrac 0.2’ and ‘–ere 0.7’ options applied to hmmbuild. The –symfrac 0.2 option ensures that only 20% of sequences need to be in agreement to cause a match in a given column (the default is 0.59). Given the remote homology at which we were performing experiments, 50% was an unreasonably high threshold that led to few match states being found. This option was also used by Kumar and Cowan (2004).

The –ere option sets the minimum relative entropy per position target to 0.7 bits (the default is 0.59). Note that HMMER versions 2.0 and 3.0 both SAM sequence entropy (Kumar and Heringa, 2004) by default. This entropy weighting scheme has been shown to be superior for remote homology detection tasks. Restriction to 0.7 increases the computational load.

HMMER 3.0α2 was used despite having been superseded by version 3.0, because it uniformly performs better on this task. This is because version 3.0 contains computational optimizations that cause it to reject a sequence (with no score provided) quickly if it does not appear to align well. These optimizations, however, cause nearly all query sequences outside the family level of homology to fail and return no score, with the result that HMMER version 3.0 never surpasses an AUC of 0.5.

2.8 RAPTOR implementation

SMURFLite was tested against RAPTOR, which was run with the options ‘–a n’ indicating that the default threading algorithm described in the RAPTOR paper (Kumar et al., 2003) was used. In addition, RAPTOR used the weighting parameters ‘weightMutation = 1.400976151,’ ‘weightSingleton = 1,’ ‘weightLoopGap = 16.841836238,’ ‘weightPair = 0,’ ‘weightGapPenalty = 1’ and ‘weightSStruct = 3.037894223.’ RAPTOR uses both sequence and structural features, and these options represent the recommended balance of these features (Kumar et al., 2003).

2.9 HHpred implementation

SMURFLite was tested against HHpred version 1.5.1. HHpred HMMs for each SCOP family were downloaded from the HHpred web site, and queried using hhsearch. The score of the best-scoring family HHM within each superfamily was used in computing ROC curves.

2.10 Whole-genome search

All 1852 protein sequences from *T. maritima* were queried against beta-structural templates constructed from the nr-PDB (Bateman et al., 2000) with non-redundancy determined by an *E*-value of 10⁻¹₀, organized according to those 207 beta-structural superfamilies from SCOP that were able to be aligned using the Matt structural alignment program, using SMURFLite with an interleave threshold of 2 and simulated evolution mutation rate of 50% on the residues that participate in beta strands. We computed *E*-values and alignments for all 1852 x 207 possible hits.

3 RESULTS

3.1 SMURFLite validation

SMURFLite’s ability to recognize beta propellers and barrels was compared with HMMER (Eddy, 1998), RAPTOR (Kumar et al., 2003) and HHpred (Bateman et al., 2000) in a stringent cross-validation experiment. From SCOP (Murzin et al., 1995; version 1.75, we chose the folds ‘5-bladed Beta-Propellers’, ‘6-bladed Beta-Propellers’, ‘7-bladed Beta-Propellers’ and ‘8-bladed Beta-Propellers’. We also chose superfamilies from all of the mostly-beta folds containing the word ‘beta’ in their description, whether open or closed, restricted to those superfamilies comprising at least four families (in order to facilitate leave-family-out cross-validation). These superfamilies were: ‘Nucleic acid-binding proteins’ (50249), ‘Translation proteins’ (50447), ‘Trypsin-like serine proteases’ (50494), ‘Barwin-like endogucanases’ (50685), ‘Cyclophilin-like’ (50891), ‘Sm-like ribonuclease proteins’ (50182), ‘PDZ domain-like’ (50156), ‘Prokaryotic SH3-related domain’ (82057), ‘Tudor/PWW/PMB’ (63748), ‘Electron Transport accessory proteins’ (50990), Translation proteins SH3-like domain’ (50104), ‘Lipocalins’ (50814) and ‘FMN-binding split barrel’ (50475). Of these, we removed the superfamilies ‘Lipocalins’ and ‘Trypsin-like serine proteases’, which were not structurally consistent enough to permit a multiple structure alignment for training HMMER or the SMURFLite variants, and which were broken into distinct superfamilies by Daniels et al. (2013), with the result that 11 superfamilies containing barrels were selected.

SMURFLite was tested on these 5 propeller folds and 11 barrel superfamilies, with interleave thresholds of 1, 2 and 3, and with and without simulated evolution on the beta-strands (Kumar and Cowan, 2004). Here, the interleave threshold is a parameter of SMURFLite that trades off the computational complexity with the ability of the MRF to capture complicated long-range dependencies.

The balance between accuracy and computational efficiency is determined by the interleave threshold at which SMURFLite is run. In particular, we found that SMURFLite set to an interleave threshold of 3 or less was always fast. Thus, our first question is how SMURFLite with and without simulated evolution performs on our test set when the interleave threshold is set to 3 or less. We found that SMURFLite became extremely slow at an interleave threshold of 4, and essentially intractable at an interleave threshold of 5 or above. While SMURFLite with an interleave threshold of 1 or 2 requires roughly 1 s of wall-clock time on a 12-core 2.4 GHz AMD Opteron server, an interleave threshold of 4 raises this runtime requirement to 7–10 min. Restricting the interleave threshold to 3 or less has different impacts on the different folds in our test set. In particular, the beta strands in the propeller folds never have an interleave >3, which means that full SMURFLite, as we know, is tractable on these folds. However, we were still interested in how simplifying the random field to an interleave of 2 or 1 would impact performance, and also whether simulated evolution would help. In contrast, the barrel superfamilies in our test set contain a maximum beta-strand interleave of between 4 and 8. Interestingly, none of these barrels contained any beta strands with an interleave of 3 in...
As discussed above, SMURFLite begins to test the limits of computational tractability when interleave numbers of 4 are allowed. Since many barrel structures had beta-strand pairs with interleave numbers of 4, we wished to test if incorporating these more long-range pairwise dependencies into our MRF would improve performance. Some barrel superfamilies on which we tested have only strand pairs of interleave 1 or 2, excepting a pair of beta strands that close the barrel and thus have an interleave equivalent to the number of strands in the barrel. Certainly, including that last strand is beyond the computational power of SMURFLite. Other barrels, whether open or closed, have more complex strand topology and interleaves of 3 or 4 are common even in the middle of the barrels. We chose to run SMURFLite with an interleave of 4 on one of the barrel superfamilies of moderately complex topology, the ‘Barwin-like endoglucanase’ superfamily, of which an example appears in Figure 1. The ‘Barwin-like endoglucanase’ superfamily contains ‘Barwin,’ a protein that may be involved in a common defense mechanism in plants (Svensson et al. 1992).

On the ‘Barwin-like endoglucanase’ superfamily, we find an enormous improvement in performance from capturing that last strand pair, with AUC improving from 0.63 for SMURFLite with an interleave threshold of 2 and simulated evolution, to 0.94 for SMURFLite with an interleave threshold of 4 and simulated evolution (Fig. 2). Note that both HHMmer and RAPTOR fail entirely on this superfamily.

3.2 SMURFLite on whole genomes

We considered all 1852 genes from the bacterium T.maritima, a thermophilic organism that bears some similarity to Archaea and whose cell is wrapped in an outer membrane, or ‘toga’ (Huber et al., 1992). Out of 354 total superfamilies within the SCOP class ‘All beta proteins’, 288 (81%) of which contain at least two protein chains, 207 superfamilies (71%) were structurally consistent enough to be aligned using the Matt Menke et al. (2008) structural alignment program. We built SMURFLite templates for these 207 superfamilies, and obtained from UniProt the protein sequences for each of 1852 genes. We predict 139 of the 1852 genes from T.maritima to belong to one of the 207 beta-strand superfamily superfamilies we consider, with a P-value of <0.005. Of the 139 genes about which we make predictions, 28 already have solved structures in the PDB, however, since there is a substantial time lag before new PDB structures are assigned to SCOP, only one of those structures was already given a SCOP assignment (and thus only one of these 28 structures potentially informed SMURFLite training). Thus, determining the correct SCOP assignments of the remaining 27 (an easy computational problem given full structural information) allows us to estimate the accuracy of SMURFLite predictions on these structures. Using the Matt Menke et al. (2008) structural alignment program and the methodology of (Daniels et al., 2013), we computed SCOP superfamilies for all 27, and in 100% of the cases, SMURFLite’s predictions matched the structural alignments and hence SCOP superfamily assignments. We now survey the remaining 111 structures on which SMURFLite makes predictions, for which structural information is not yet available. In total, 8 of these 111 structures also had their SCOP superfamilies predicted in the study of (Phane et al., 2004), and in all 8 cases, our predictions are in agreement with the other study. We note that for most of these 111 structures, not only is there not solved
Table 1. AUC on beta-propeller folds. Best AUC for each structure is marked in bold

<table>
<thead>
<tr>
<th>Structure</th>
<th>HMMER</th>
<th>RAPTOR</th>
<th>HHpred</th>
<th>SMURF-Lite 1</th>
<th>SMURF-Lite 1, SimEv</th>
<th>SMURF-Lite 2</th>
<th>SMURF-Lite 2, SimEv</th>
<th>SMURF-Lite 3</th>
<th>SMURF-Lite 3, SimEv</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-bladed</td>
<td>–</td>
<td>–</td>
<td>0.75</td>
<td>0.89</td>
<td>0.73</td>
<td>0.89</td>
<td>0.73</td>
<td>0.89</td>
<td>0.73</td>
</tr>
<tr>
<td>6-bladed</td>
<td>0.82</td>
<td>0.82</td>
<td>0.88</td>
<td>0.92</td>
<td>0.93</td>
<td>0.96</td>
<td>0.95</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>7-bladed</td>
<td>0.89</td>
<td>0.95</td>
<td>0.99</td>
<td>0.92</td>
<td>0.91</td>
<td>0.93</td>
<td>0.93</td>
<td>0.93</td>
<td>0.93</td>
</tr>
<tr>
<td>8-bladed</td>
<td>0.64</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
</tr>
</tbody>
</table>

Note: for SMURF-Lite, the number (1, 2, 3) indicates the interleave threshold, and SimEv is simulated evolution. A dash (‘–’) in a result entry indicates the method failed on these structures, i.e. an AUC of <0.6.

Table 2. AUC on beta-barrel superfamilies

<table>
<thead>
<tr>
<th>Superfamily</th>
<th>HMMER</th>
<th>RAPTOR</th>
<th>HHpred</th>
<th>SMURF-Lite 1</th>
<th>SMURF-Lite 1, SimEv</th>
<th>SMURF-Lite 2</th>
<th>SMURF-Lite 2, SimEv</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMURF-Lite performs best</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation proteins</td>
<td>–</td>
<td>–</td>
<td>0.66</td>
<td>0.93</td>
<td>0.92</td>
<td>0.93</td>
<td>0.93</td>
</tr>
<tr>
<td>Barwin-like endoglucanases</td>
<td>–</td>
<td>–</td>
<td>0.75</td>
<td>0.77</td>
<td>–</td>
<td>0.82</td>
<td>0.82</td>
</tr>
<tr>
<td>Cyclophilin-like</td>
<td>0.67</td>
<td>0.61</td>
<td>0.7</td>
<td>0.82</td>
<td>0.71</td>
<td>0.76</td>
<td>0.85</td>
</tr>
<tr>
<td>Sm-like ribonucleoproteins</td>
<td>0.73</td>
<td>0.71</td>
<td>0.77</td>
<td>0.76</td>
<td>0.71</td>
<td>0.76</td>
<td>0.85</td>
</tr>
<tr>
<td>Prokaryotic SH3-related domain</td>
<td>0.81</td>
<td>–</td>
<td>–</td>
<td>0.83</td>
<td>0.82</td>
<td>0.83</td>
<td>0.83</td>
</tr>
<tr>
<td>Tudor/PPP1-binding</td>
<td>0.78</td>
<td>0.74</td>
<td>0.67</td>
<td>0.83</td>
<td>0.77</td>
<td>0.83</td>
<td>0.79</td>
</tr>
<tr>
<td>Nucleic acid-binding proteins</td>
<td>0.75</td>
<td>–</td>
<td>0.67</td>
<td>0.76</td>
<td>0.89</td>
<td>0.76</td>
<td>0.92</td>
</tr>
<tr>
<td>RAPTOR performs best</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation proteins SH3-like</td>
<td>0.83</td>
<td>0.81</td>
<td>0.86</td>
<td>0.62</td>
<td>–</td>
<td>0.62</td>
<td>–</td>
</tr>
<tr>
<td>PDZ domain-like</td>
<td>0.96</td>
<td>1.0</td>
<td>0.99</td>
<td>0.97</td>
<td>0.97</td>
<td>0.97</td>
<td>0.97</td>
</tr>
<tr>
<td>FMN-binding split barrel</td>
<td>0.62</td>
<td>0.82</td>
<td>0.61</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>HHpred performs best</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electron Transport accessory proteins</td>
<td>0.84</td>
<td>–</td>
<td>0.77</td>
<td>0.63</td>
<td>–</td>
<td>0.63</td>
<td>0.66</td>
</tr>
</tbody>
</table>

Note: for SMURF-Lite, the number (1, 2) indicates the interleave threshold, and SimEv is simulated evolution. A dash (‘–’) in a result entry indicates the method failed on these structures, i.e. an AUC of <0.6.

4 DISCUSSION

We have presented SMURF-Lite, a method that combines long-range pairwise beta-strand interactions via a simplified MRF with simulated evolution, a method that augments training data to capture pairwise beta-strand interactions as well. SMURF-Lite in most cases performs considerably better than HMMER and RAPTOR; however, we examine those structures for which this is not so. We postulate that RAPTOR performs best in the case when there is significant structural conservation across families, whereas HMMER excels when there is a small but highly conserved sequence signature in members of a superfamily. In all four beta-barrel superfamilies on which RAPTOR achieves an AUC of <0.5, we see considerable structural variation in the protein backbones within each superfamily, according to the metric of Daniels et al. (2012), as compared with the other barrel superfamilies. In contrast, the barrels on which RAPTOR performed best exhibited little structural variation. The cases in which SMURF-Lite performs poorly exhibit an interesting property: the structural alignment of the protein chains used in the training set preserves few, or sometimes none, of the beta strands as ‘consensus’ beta strands. When a significant number of beta strands are missing in this manner from the training data, SMURF-Lite exhibits poor specificity, scoring some non-homologous sequences comparably to homologous ones. The ‘Translation Proteins SH3-Like Domain,’ a superfamily in which HMMER significantly outperforms SMURF-Lite, is one in which the consensus alignment obtained from Matt retains zero beta strands, even though each individual structure has four strands. Thus, SMURF-Lite behaves like HMMER, except without HMMER’s heuristic for quickly failing bad alignments, leading SMURF-Lite to report more false positives. The very premise of SMURF-Lite rests on the conservation of beta strands, and this finding emphasizes the importance of evolutionarily faithful structural alignments. In future work, we will also consider alternative structural aligners, such as TMalign (Zhang and Skolnick, 2005), in cases where they produce alignments that better conserve secondary structure.
We also compared SMURFLite to HHpred, though in a sense this is not an apples-to-apples comparison, because HHpred uses all of protein sequence space to build profiles for training; thus it can leverage a much larger training set than THMMER, RAPTOR, or SMURF or SMURFLite. Thus it is somewhat surprising that SMURFLite outperforms HHpred in median AUC on the propellers and barrels. We expect HHpred to excel in particular on superfamilies and folds with a high HHpred NEFF. \cite{Soding2000}, where NEFF is the "number of effective families" available for training the HHpred HMM.

In contrast, simulated evolution seems to help SMURFLite most on those structural motifs where the HHpred NEFF is lowest; i.e. it can generate diverse training data when diverse training data is lacking. A profile version of SMURFLite would be close in spirit to HHpred, but unlike HHpred it can build profiles without knowing the structures; we would expect profiles might improve performance; this will be a subject for future investigation. We observed that simulated evolution either improves or does not affect AUC for beta-barrel superfamilies and beta-propeller profiles with a HHpred NEFF of 20 or lower. The only cases in which we observed simulated evolution decreasing AUC were those cases where the NEFF was >20.

While the intent of using simulated evolution in conjunction with simplified MRFs is to compensate for the removal of highly-interleaved beta-strand pairs required for computational feasibility, we find that simulated evolution can still improve full-fledged SMURF in cases of sparse training data. For instance, the 5-bladed beta-propeller fold, combining SMURF and simulated evolution improves AUC from 0.73 for full SMURF alone to 0.89. We have demonstrated that SMURFLite is a powerful MRF methodology for beta-structural motif recognition that is computationally tractable enough to scale to whole genomes, requiring approximately 3h to scan the Thermotoga maritima genome on a small compute cluster. We have also shown that increasing the interleaved number for SMURFLite can have dramatic effects on small compute cluster. We have also shown that increasing the number of interleaved beta-strand pairs required for computational feasibility, we find that simulated evolution combining SMURF and simulated evolution improves AUC from 0.73 for full SMURF alone to 0.89.

We have demonstrated that SMURFLite is a powerful MRF methodology for beta-structural motif recognition that is computationally tractable enough to scale to whole genomes, requiring approximately 3h to scan the Thermotoga maritima genome on a small compute cluster. We have also shown that increasing the number of interleaved beta-strand pairs required for computational feasibility, we find that simulated evolution combining SMURF and simulated evolution improves AUC from 0.73 for full SMURF alone to 0.89.

ACKNOWLEDGEMENTS

We thank A.Kumar, M.Menke and J.Ju.

Funding: N.M.D. and L.J.C. were funded in part by NIH grant R01GM080330. R.H. and B.B. were funded in part by NIH grant R01GM08187.

Conflict of Interest: none declared.

REFERENCES

HubER,R. et al. (1986) Thermotoga maritima sp. nov. represents a new genus of unique extremely thermophilic eubacteria growing up to 90C. Arch. Microbiol., 144, 324-333.