Discrete profile comparison using information bottleneck

Sean O'Rourke*[1], Gal Chechik[2] , Robin Friedman[1] and Eleazar Eskin[1]

[1]Department of Computer Science and Engineering,University of California San Diego, 9500 Gilman Dr.,San Diego, CA 92093
[2]Department of Computer Science, Stanford University,353 Serra Mall,Stanford University,Stanford CA 94305

Email: Sean O'Rourke*- seano@cs.ucsd.edu; Gal Chechik - gal@stanford.edu; Robin Friedman - rcfriedm@ucsd.edu; Eleazar Eskin - eeskin@cs.ucsd.edu;

*Corresponding author

Abstract

Sequence homologs are an important source of information about proteins. Amino acid profiles, representing the position-specific mutation probabilities found in profiles, are a richer encoding of biological sequences than the individual sequences themselves. However, profile comparisons are an order of magnitude slower than sequence comparisons, making profiles impractical for large datasets. Also, because they are such a rich representation, profiles are difficult to visualize. To address these problems, we describe a method to map probabilistic profiles to a discrete alphabet while preserving most of the information in the profiles. We find an informationally optimal discretization using the Information Bottleneck approach (IB). We observe that an 80-character IB alphabet captures nearly 90% of the amino acid occurrence information found in profiles, compared to the consensus sequence's 78%. Distant homolog search with IB sequences is 88% as sensitive as with profiles compared to 61% with consensus sequences (AUC scores 0.73, 0.83, and 0.51, respectively), but like simple sequence comparison, is 30 times faster. Discrete IB encoding can therefore expand the range of sequence problems to which profile information can be applied to include batch queries over large databases like SwissProt, which were previously computationally infeasible.

Introduction

One of the most powerful techniques in protein analysis is the comparison of a target amino acid sequence with phylogenetically related or homologous proteins. Such comparisons can give insight into which portions of the protein are important by revealing the parts that were conserved through natural selection. While mutations in non-functional regions may be harmless, mutations in functional regions are often lethal. For this reason, functional regions of a protein tend to be conserved between organisms while non-functional regions diverge.

Many of the state-of-the-art protein analysis techniques incorporate homologous sequences by representing a set of homologous sequences as a probabilistic *profile*, a sequence of the marginal distributions of amino acids at each position in the sequence. For example, PSI-BLAST [1] uses profiles to refine database searches. The PHD algorithm [2] uses them purely for structure prediction. Yona et al. [3] used profiles to align distant homologs from the SCOP database [4]; the resulting alignments are similar to results from structural alignments, and tend to reflect both secondary and tertiary protein

structure.

Although profiles provide a lot of information about the sequence, their use comes at a steep price. While efficient algorithms exist for aligning protein sequences and performing database queries (e.g. BLAST [1]), these algorithms operate on strings and are not applicable to profile alignment or profile database queries. Profile-based comparisons can be substantially more accurate than sequence-based ones, but are about 30 times slower, since substitution penalties must be calculated by computing distances between probability distributions rather than simply looked up in a table. This makes probabilistic profiles impractical for use with large bioinformatics databases like SwissProt, which recently passed 160,000 sequences and 64 million amino acids [5].

We propose a new discrete representation of proteins that incorporates information from homologs in a textual form we call *IB (Information Bottleneck) sequences*. Once a profile is represented using this discrete alphabet, alignment and database search can be performed using the efficient string algorithms developed for sequences, making profile information applicable to a greater range of problems. For example, the runtime for full pairwise Smith-Waterman [6] alignment between this sequence and all of SwissProt decreases from 250 hours to less than 8; a query for high-scoring alignments to 100 sequences of interest would take nearly three CPU-years with profiles, but just over a month with IB sequences. Either the resulting IB sequence alignments can be used directly, or a small set of high-scoring matches from this initial query can be re-aligned using profiles for greater precision. Therefore with IB sequences, profile information may be applied to a greater range of sequence problems with no loss in precision and minimal loss in recall.

IB sequences have another incidental benefit: By representing each class as a letter, discretized profiles can be presented in plain text, conveying more profile information than the original sequences in the same amount of space. These IB sequences are more accurate than consensus sequences and denser than profile matrices or sequence logos (see Figure 1). While sequence logos are likely a better representation for examining individual alignments, terse IB sequences are useful for presenting many alignments at once, such as when interpreting database query results. For example, Figure 1(c) shows that while logos more accurately reflect the first profile column, information about lower-conservation regions is completely lost at ordinary text size.

The main idea behind our approach is to compress profiles in a data-dependent manner by clustering the actual profiles and representing them by a small alphabet of distributions. Since this discretization removes some of the information carried by the full profiles, we cluster the distribution in a way that is directly targeted at minimizing the information loss. This is achieved using a variant of the Information Bottleneck (IB) method [7], a distributional clustering approach for informationally optimal discretization. To preserve a clear textual representation, we want this discretization to also reflect biologically meaningful categories by forming a superset of the standard 20-character amino acid alphabet. For example, we use "A" and "a" for strongly- and weakly-conserved Alanine. This formulation demands two types of constraints: similarities of the clusters' conditional amino acid distributions to predefined values, and specific structural similarities between strongly- and weakly-conserved variants. We show below how the original IB formalism can be extended to naturally account for such constraints.

We apply our algorithm to SCOP [4], a database of proteins grouped hierarchically by structural similarity, and analyze the results in terms of both information loss and alignment quality. We show that IB discretization preserves much of the information in the original profiles using a small number of classes. We then show that like profile alignments, high-scoring IB alignments reflect distant homology, but that IB alignments can be computed 30 times faster than profile ones. IB discretization is therefore an attractive way to gain some of the additional sensitivity of profiles on tasks for which profile-profile comparison is not computationally feasible.

Information Bottleneck

Information Bottleneck [7] is an information theoretic approach for distributional clustering. Given a joint distribution $p(X, Y)$ of two random variables X and Y, the goal is to obtain a compressed representation C of X, while preserving the information about Y. The two goals of compression and information preservation are quantified by the same measure

of mutual information,

$$I(X;Y) \stackrel{\text{def}}{=} \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = H(X) + H(Y) - H(X,Y)$$

where $H(X)$ is the entropy of $p(X)$, and $H(X,Y)$ of $p(X,Y)$. $I(X;Y)$ is symmetric and non-negative, and is zero only when X and Y are conditionally independent. The IB problem is defined as the constrained optimization problem

$$\min_{p(c|x): I(C;Y) > K} I(C;X) \quad (1)$$

where K is a constraint on the level of information preserved about Y. The solution must also obey the constraints $p(y|c) = \sum_x p(y|x)p(x|c)$ and $p(y) = \sum_x p(y|x)p(x)$. This constrained optimization problem can be reformulated using Lagrange multipliers, and turned into a tradeoff optimization function with Lagrange multiplier β:

$$\min_{p(c|x)} \mathcal{L} \stackrel{\text{def}}{=} I(C;X) - \beta I(C;Y) \quad (2)$$

As an unsupervised learning technique, IB aims to characterize the set of solutions for the complete spectrum of constraint values K. This set of solutions is identical to the set of solutions of the tradeoff optimization problem obtained for the spectrum of β values.

When X is discrete, its natural compression is fuzzy clustering. In this case, the problem is not convex and cannot be guaranteed to contain a single global minimum. Fortunately, its solutions can be characterized analytically by the following set of self consistent equations:

$$p(c|x) = \frac{p(c)}{Z(x,\beta)} \exp\left(-\beta D_{KL}[p(y|x)||p(y|c)]\right) \quad (3)$$

$$p(y|c) = \sum_x p(y|x)p(x|c) \quad (4)$$

$$p(c) = \sum_x p(c|x)p(x)$$

where

$$Z(x,\beta) = \sum_c p(c) \exp\left(-\beta D_{KL}[p(y|x)||p(y|c)]\right)$$

By first computing $p(c|x)$ using Eq. (3), then recomputing the other distributions via Eqs. (4), these equations yield an iterative algorithm that is guaranteed to converge to a local minimum [7]. While the optimal solutions of the IB functional are in general soft clusters, hard clusters are often more easily interpreted in practice. A series of algorithms was developed for hard IB, including an algorithm that can be viewed as a one-step look-ahead sequential version of K-Means [8]. See the Methods section for a description and comparison of the iterative and sequential IB algorithms.

Applying IB to our profile discretization problem, X ranges over the set of single-position probabilistic profiles obtained from a set of aligned sequences and Y ranges over the set of 20 amino acids. In other words, $p(y|x)$ is the probability of observing amino acid y at profile position x. Our goal is to construct clusters c of positions x sharing similar amino acid distributions: for each position x assigned to each cluster c, the profile at x, $p(Y|X=x)$, should be well-approximated by c's representative profile $p(Y|C=c)$.

Constraints on cluster conditional distributions

The application studied in this paper differs from standard IB in that we are interested in obtaining a representation that is both efficient and biologically meaningful. This requires that we add two kinds of constraints on clusters' distributions.

First, some clusters' meanings are naturally determined by limiting them to correspond to the common 20-letter alphabet used to describe amino acids. From the point of view of distributions over amino acids, each of these symbols is used today as the delta function distribution which is fully concentrated on a single amino acid. For the goal of finding an efficient representation, we require the conditional distributions $p(Y|C=c)$ to be close to these delta distributions. More generally, we require $p(Y|C=c)$ for a specific cluster c to be close to predefined value $p(Y|C=\hat{c})$, thus adding constraints to the IB target function of the form $D_{KL}[p(y|\hat{c})||p(y|c)] < K(c)$ for each such constraint. While solving the constrained optimization problem is difficult, the corresponding tradeoff optimization problem can be made very similar to standard IB. With the additional constraints, the IB trade-off optimization problem becomes

$$\min_{p(c|x)} \mathcal{L}' \equiv I(C;X) - \beta I(C;Y) + \beta \sum_{c \in C} \beta(c) D_{KL}[p(y|\hat{c})||p(y|c)] \quad (5)$$

We now use the following identity

$$\begin{aligned}
&\sum_{x,c} p(x,c) D_{KL}[p(y|x)||p(y|c)] \\
= &\sum_x p(x) \sum_y p(y|x) \log p(y|x) \\
&- \sum_c p(c) \sum_y \log p(y|c) \sum_x p(y|x) p(x|c) \\
= &-H(Y|X) + H(Y|C) = I(X;Y) - I(Y;C)
\end{aligned}$$

to rewrite the IB functional of Eq. (2) as

$$\begin{aligned}
\mathcal{L} = &\ I(C;X) + \beta \sum_{c\in C} \sum_{x\in X} p(x,c) D_{KL}[p(y|x)||p(y|c)] \\
&-\beta I(X;Y)
\end{aligned}$$

When $\sum_{c\in C} \beta(c) \leq 1$, we can similarly rewrite Eq. (5) as

$$\begin{aligned}
\mathcal{L}' = &\ I(C;X) \\
&+\beta \sum_{x\in X} p(x) \sum_{c\in C} p(c|x) D_{KL}[p(y|x)||p(y|c)] \\
&+\beta \sum_{c\in C} \beta(c) D_{KL}[p(y|\hat{c})||p(y|c)] - \beta I(X;Y) \\
= &\ I(C;X) \\
&+\beta \sum_{x'\in X'} p(x') \sum_{c\in C} p(c|x') D_{KL}[p(y|x')||p(y|c)] \\
&-\beta I(X;Y)
\end{aligned} \tag{6}$$

The optimization problem therefore becomes equivalent to the original IB problem, but with a modified set of samples $x \in X'$, containing X plus additional pseudo-counts x' with prior probability $p(x') = \beta(c)$ (hence the requirement that $\sum_{c\in C} \beta(c) \leq 1$). This is similar to the inclusion of priors in Bayesian estimation.

Formulated this way, the biases can be easily incorporated in standard IB algorithms as additional pseudo-data. From an initial dataset defined by $p(y|x)$ and $p(x)$ (typically $\frac{1}{|X|}$ for profiles) and biases $\hat{C} = \{\hat{c}\}$ with values $p_\beta(y|\hat{c})$, we construct a new dataset $X' = X \cup \hat{C}$ defined by

$$p'(y|x') = \begin{cases} p(y|x') & \text{if } x' \in X \\ p_\beta(y|x') & \text{if } x' \in \hat{C} \end{cases}$$

and

$$p'(x') = \begin{cases} (1 - \sum_c \beta(c))\, p(x') & \text{if } x' \in X \\ \beta(x') & \text{if } x' \in \hat{C} \end{cases}$$

Finally, Eq. (5) is augmented to assign each pseudo-datapoint $\hat{c}$ to its cluster c, with $p(c|\hat{c}) = 1$, thereby fixing the biases to their clusters.

Constraints on relations between cluster distributions

We want our discretization to capture both strongly- and weakly-conserved variants of the same symbol. While this can be done with standard IB using separate classes for the alternatives, the strong and weak variants' distributions are likely to be correlated. It is therefore preferable to define a single shared prior for both variants, and to learn a model capturing their correlation.

Friedman et al. [9] describe *multivariate information bottleneck* (mIB), an extension of information bottleneck to joint distributions over several correlated input and cluster variables. Instead of a single observed variable X and a single cluster variable C, mIB incorporates sets of observed variables $\mathcal{X}$ and compression variables $\mathcal{C}$ with a specific conditional dependency structure. Intuitively, mIB's goal is to find distributions $p(\mathcal{C}|X)$ and $p(Y|\mathcal{C})$ such that $\sum_{\mathbf{c}\in\mathcal{C}} p(Y|\mathbf{c}) p(\mathbf{c}|X)$ approximates $p(Y|\mathcal{X})$. For further details, including a formulation of the problem for arbitrary compression structures and a derivation of an analogous loss function, see Friedman et al. [9].

For profile discretization, we define two compression variables connected as in Friedman et al.'s "parallel IB": an amino acid class $C \in \{A, C, \ldots\}$ with an associated prior, and a conservation strength $S \in \{0, 1\}$. Our goal is to maximize the information about amino acid distribution Y contained in C and S together, while independently minimizing the information about position X contained in C and S. The IB loss function therefore becomes

$$\mathcal{L}_m \stackrel{\text{def}}{=} I(C;X) + I(S;X) - \beta I(Y;S,C) \tag{7}$$

Figure 2 illustrates our two models' dependency structures and parameterizations. Since the multivariate model correlates strong and weak variants of each category, it requires fewer priors than simple IB. It also has fewer parameters: a multivariate model with n_s strengths and n_c classes has as many categories as a univariate one with $n_{c'} = n_s n_c$ classes, but has only $n_s + n_c - 2$ free parameters for each x, instead of $n_s n_c - 1$.

Results

We evaluate IB alignment's ability to detect distant homologs by comparing the orders of profile, IB, and consensus alignment scores for a set of proteins with known evolutionary and structural relations. We also compare the pattern of gaps in individual profile alignments to those in the equivalent IB and consensus alignments. IB scores, like profile scores, capture a significant number of relations missed by consensus scores, and individual IB alignments more closely reflect the pattern of insertions in the original profile alignments.

Our data come from SCOP [4], a manually-constructed database of proteins grouped hierarchically by structural similarity and evolutionary relatedness. We expect proteins within the same SCOP family, which have clear evolutionary relationships and $\sim 30\%$ sequence identity, to have high-scoring profile and sequence alignments. We also expect proteins from different families in the same superfamily, which have probable evolutionary relationships but low sequence identity, to have significant but lower-scoring profile alignments but no significant sequence alignments.

For each protein, we first generate a profile from a CLUSTALW [10] multiple alignment with other proteins in its family, yielding 425,150 individual sequence positions, then compute probabilistic profiles ignoring gap characters. We then compute IB classes from these profiles using iIB and the priors described below. Finally, we discretize the profiles into the resulting classes, using the Jensen-Shannon (JS) distance with mixing coefficient 0.5 rather than the KL distance optimized in encoding profiles to be consistent with Yona et al. [3].

In the following sections, we first examine how the amount of information from the original profiles encoded by IB categories varies with the number of clusters. We then consider how model structure, i.e. priors and relations between clusters, affects this information. Next, we compare individual IB, consensus, and profile alignments, and compare the order of alignment scores between distantly-related and unrelated proteins. Finally, we show how running time for profile and IB alignment varies with sequence length.

Information loss from discretization

One measure of the quality of IB clusters is the amount of information about Y (the amino acid distribution) lost through discretization, $I(Y;X) - I(Y;C)$. This represents the total information distance between profiles and the centers of their assigned clusters, and is a task-independent measure of the quality of a discretization. The change in $I(Y;X) - I(Y;C)$ between successive values of $|C|$ represents the amount of information gained by adding more categories, and thus the number of actual clusters of a particular scale in the data. Figure 4 shows the cluster information $I(Y;C)$ and position information $I(C;X)$ for consensus sequences, profiles, and (iterative) IB with no priors for $|C| = 40, \ldots, 500$. With $|C| \geq 80$ the IB alphabet captures over 90% of the available information.

Figure 3 shows the sequence logos for discretizations with $|C| = 20, 40, 80$ illustrating compression's effects. First, when the number of labels equals the the number of amino acids ($|C| = 20$), the frequently-occurring amino acid `A` is allocated two labels, forcing `D` and `R` share a label. Second, as the number of labels increases, the least common amino acid `C` is allocated only a single label, while the number of labels assigned to the more common `A` and `L` consistently increases. This shows the data-dependence of our discretization compared to the simpler approach of allocating one or more clusters to each amino acid with varying levels of sequence conservation.

Effect of category constraints

For univariate IB, we have used four types of priors reflecting biases on stability, physical properties, and observed substitution frequencies: (1) *Strongly conserved* classes, in which a single symbol is seen with $S\%$ probability. These are the only priors used for multivariate IB. (2) *Weakly conserved* classes, in which a single symbol occurs with $W\%$ probability; $(S - W)\%$ of the remaining probability mass is distributed among symbols with non-negative log-odds of substitution. (3) *Physical trait* classes, in which all symbols with the same hydrophobicity, charge, polarity, or aromaticity occur uniformly $S\%$ of the time. (4) A *uniform* class, in which all symbols occur with their background probabilities.

The choice of S and W depends upon both the data and one's prior notions of "strong" and "weak" conservation. Unbiased IB on a large subset of SCOP with several different numbers of unbiased categories yielded a mean frequency approaching 0.7 for the most common symbol in the 20 most

sharply-distributed classes (0.59 ± 0.13 for $|C| = 52$; 0.66 ± 0.12 for $|C| = 80$; 0.70 ± 0.09 for $|C| = 100$). Similarly, the next 20 classes have a mean most-likely-symbol frequency around 0.4. These numbers can be seen as lower bounds on S and W. We therefore chose $S = 0.8$ and $W = 0.5$, reflecting a bias toward stronger definitions of conservation than those inferred from the data.

Figure 4 shows the effect on information loss of varying the prior weight $\sum_c \beta(c)$ with three sets of priors: 20 strongly conserved symbols and one background; these plus 20 weakly conserved symbols; and these plus 10 categories for physical characteristics. As expected, increasing the number or weight of priors increases information loss. However, with a small additional pool of unbiased categories information loss is nearly independent of prior strength. This suggests that our priors correspond to actual regularities in the data. Finally, note that despite having fewer free parameters than the univariate models, mIB's achieves comparable performance, suggesting that our decomposition into conserved class and degree of conservation is reasonable.

Alignment similarity and distant homolog search

Since we are ultimately using the resulting IB classes in alignments, the true cost of discretization is best measured by the amount of change between profile and IB alignments, and the significance of this change. The latter is important because the best path can be very sensitive to small changes in the sequences or scoring matrix; if two radically different alignments have similar scores, neither is clearly "correct".

We can represent an alignment as a pair of index-insertion sequences, one for each profile sequence to be aligned (e.g. "1,2,-,-,3,..." versus "1,-,2,-,3,..."). The edit distance between these sequences for two alignments then measures how much they differ. However, even when this distance is large, the difference between two alignments may not be significant if both choices' scores are nearly the same. That is, if the optimal profile alignment's score is only slightly lower than the optimal IB class alignment's score *as computed with the original profiles*, either might be correct. We therefore report both the edit distance between alignments and this change in profile alignment score.

The score for aligning two IB symbols c and d is

$$\frac{1}{2}\big(1-D^{JS}[p(y|c)||p(y|d)]\big)\big(1+D^{JS}[q(y)||\bar{p}(y)]\big)-k_s$$

where $q(y) = \frac{1}{2}(p(y|c) + p(y|d))$, $\bar{p}(y)$ is the average probability of amino acid y across all profiles, and k_s is a constant chosen so that the average alignment score between pairs of randomly-chosen symbols is negative. We use $k_s = 0.45$ and gap open and extension penalties of $k_o = 2$ and $k_e = 0.2$, where k_s, k_o, and k_e have been chosen by Yona et al. [3] so that local alignment scores between random sequences follow the expected extreme value distribution.

Figure (6a) shows both the edit distance and score change per length between profile alignments and those using IB classes, mIB classes, and the original sequences with the BLOSUM62 scoring matrix. Unless otherwise noted, IB alignments use $|C| = 52$ clusters, a number chosen to be conveniently represented by the 26 upper- and lower-case letters. To compare the profile and sequence alignments, profiles corresponding to gaps in the original sequences are replaced by gaps, and resulting pairs of aligned gaps in the profile-profile alignment are removed. We consider both sequences from the same family and those from other families in the same clan, the former being more similar than the latter, and therefore having better alignments. Assuming the profile-profile alignment is closest to the "true" alignment, IB alignment significantly outperforms sequence alignment in both cases, with mIB showing a slight additional improvement.

Since alignment scores predict structural relatedness, sequences with distant structural relationships, defined as those in the same SCOP superfamily, should have positive-scoring alignments. Yona et al. [3] compare the ranking of high-scoring profile-profile alignments to that of PSI-BLAST e-values, and show that profiles consistently assign high scores to more distant homologs. We perform this same test to compare profile, IB, and consensus sequence alignment scores. Figure (6b) shows the ROC curve for detecting superfamily relationships between 117 families contained in 10 randomly-chosen SCOP superfamilies with between 3 and 35 members. While IB fares worse than profiles, consensus sequences perform essentially at chance.

Alignment running time

Most of the cost of aligning two profile sequences comes from computing JS distances between pairs of profiles. Encoding unencoded profile sequences before alignment, by significantly reducing the number of JS distance computations, yields a 4- to 20-fold improvement in alignment running time. Furthermore, sequences can be pre-encoded to perform repeated comparisons, yielding a 30-fold improvement.

Encoding two sequences of lengths n and m for IB alignment requires computing the $|C|(n+m)$ JS distances between each profile and each category, a significant improvement over the mn distance computations required for profile-profile alignment when $|C| \ll \frac{\min(m,n)}{2}$. Once the sequences are encoded or the pairwise distances computed, both methods take essentially the same amount of time to perform Smith-Waterman alignment. Figure 5 compares the running time of profile and IB alignment for different sequence lengths, showing best fit curves for both to $f(x) = ax^b$. The results show that the number of JS distance computations dominates running time for typical sequence lengths: despite both methods' performing $O(n^2)$ work in Smith-Waterman alignment, IB alignment time is essentially linear in sequence length, while profile alignment is quadratic. Figure 5 also plots the time taken to encode both input profiles in a 40-character IB alphabet, showing that encoding accounts for most of the cost of alignment. Since useful values of $|C|$ are much smaller than the average sequence length, and since most database applications can use pre-encoded sequences, the effect of $|C|$ on real running times is negligible. In particular, the time taken to align pre-encoded sequences is independent of $|C|$ for the values presented here.

On current hardware, each profile distance computation takes about $16.5\mu s$. At this rate, just the distance computations for a full pairwise alignment of SwissProt's 160,000 sequences, comprising 60 million residues, requires 3×10^{10} CPU-seconds or ~ 950 years. The distance computations required to encode the database in a 40-character alphabet take 11 hours. Similarly, the distance computations for aligning a single 200-element sequence with every element of SwissProt take about 260 hours, or nearly the entire observed running time for performing full profile alignments. To compare, our unoptimized implementation of Smith-Waterman takes around 8 hours to align a typical sequence against SwissProt. This agrees well with the figure obtained by assuming an average sequence length of 365 residues and the observed single alignment times shown in figure 5 (minus encoding time). Even with IB encoding, full pairwise alignment of SwissProt would take an impractical 60 years with our simplistic Smith-Waterman code. However, discretizing the data makes it possible to apply more efficient algorithms and indexing schemes like BLAST [1], which were developed for simple sequences.

Conclusions

We have described IB sequences, a discrete encoding of amino acid profiles that allows profile information to be used for alignment and search at essentially the same computational cost as simple sequence alignment. The encoding is based on minimizing information loss, and its classes can be constrained to correspond to the standard amino acid representation, thus yielding an intuitive, compact textual form for profile information. Alignments of IB sequences encoded with a modest number of classes correspond significantly better to the original profile alignments than do alignments of the consensus sequences (edit distance 0.15 versus 0.39). High-scoring IB alignments reflect distant homology detected with profiles but not with consensus sequences (AUC score 0.73 versus 0.51).

Our model is potentially advantageous in three ways: First, it models rich conditional distribution structures and class constraints. It can, for example, be extended to incorporate structural information in the input representation, and to assigning structural significance to the resulting categories. Second, it allows us to apply existing fast discrete algorithms to continuous profile sequences when either profile comparison is computationally impractical, or only discrete-sequence algorithms exist.

Third, discretization avoids undersampling problems while going beyond single-position profiles. Ordinary profile applications are limited by high dimensionality to considering only single positions of a multiple alignment. This ignores significant correlation both between adjacent profile positions and between adjacent symbols in individual aligned homologs. Single-position profiles thus represent a drastic simplification of the underlying data. For example, while the average entropy of a single profile in our dataset is 0.99, the average entropy of

an adjacent pair is only 1.23, suggesting an information loss far greater than the 10% lost by IB discretization. Therefore profile pairs can be represented more compactly than the cross-product of the single-position alphabet. Instead of considering sequences of adjacent single-position profiles, our method can be extended to discretize distributions over pairs or k-tuples of symbols in a multiple alignment. By applying IB to the 20^k-dimensional space of k-tuple profiles, we can avoid undersampling and obtain a richer sequence representation incorporating previously-ignored local correlation. Extending this approach to variable-length substrings yields an algorithm similar to suffix trees, known to be some of today's most efficient text compression methods [11].

Acknowledgments

The authors would like to thank the anonymous reviewers for their many helpful suggestions.

Methods: Iterative vs. sequential IB

Slonim [8] compares the performance and runtime of several IB algorithms. The first, iterative IB (iIB) (Figure 7), alternately updates the cluster assignment $p(c|x)$ and the resulting cluster distributions $p(y|c)$ and weights $p(c)$ via Eqs. (3,4). If hard clusters are desired, hard assignments are made in the first step. Since this algorithm only guarantees convergence to a local extremum, we repeated our experiments with five random initializations. In the current implementation, iteration was stopped when the current and previous distributions were sufficiently close together, as measured by $\sum_{y,s,c} |p_{t+1}(y|s,c) - p_t(y|s,c)|$.

The second IB algorithm, sequential IB (sIB) (Figure 8), first assigns elements to a fixed number of clusters, then individually moves them from cluster to cluster while calculating a 1-step lookahead score, until the score converges. Like iIB, sIB only guarantees convergence to a local extremum, and was therefore initialized with the results of five separate iIB runs.

Slonim [8] found that hard-clustering sIB outperformed soft-clustering iIB on a document clustering task with 5,000 to 500,000 documents, finding fewer and better solutions on 100 random restarts. However, while sIB is more efficient than exhaustive bottom-up clustering methods like agglomerative clustering, sIB is still more expensive than iIB, since each reassignment of an instance requires recomputing the class conditional distributions. Therefore we used iIB with hard clustering, which only recomputes the conditional distributions after performing all updates. This reduces the convergence time from several hours to around ten minutes.

Slonim argued that sIB outperforms soft iIB in part because sIB's discrete steps allow it to escape local optima. We expect hard iIB to have similar behavior. To test this, we applied three complete sIB iterations to clusters obtained by multivariate iIB. sIB decreased the loss $\mathcal{L}$ by only about 3 percent (from 0.380 to 0.368), with most of this gain occurring in the first iteration. Up to exchanging labels, the 20 strongly-conserved categories were nearly unchanged, while about half of the weakly-conserved categories changed only slightly. This suggests that hard iIB and sIB find similar regularities in our data.

References

1. Altschul S, Gish W, Miller W, Myers E, Lipman D: **Basic local alignment search tool**. *J Mol Biol* 1990, **215**(3):403–10.
2. Rost B, Sander C: **Prediction of protein secondary structure at better than 70% accuracy**. *J Mol Biol* 1993, **232**:584–99.
3. Yona G, Levitt M: **Within the twilight zone: A sensitive profile-profile comparison tool based on information theory**. *J Mol Biol* 2002, **315**:1257–75.
4. Murzin A, Brenner S, Hubbard T, Chothia C: **SCOP: a structural classification of proteins database for the investigation of sequences and structures**. *J Mol Biol* 1995, **247**:536–40.
5. Bairoch A, et al: **The Universal Protein Resource (UniProt)**. *Nucl. Acids Res.* 2005, **33**(suppl. 1):D154–159.
6. Smith T, Waterman M: **Identification of common molecular subsequences**. *J Mol Biol* 1981, **147**:195–197.
7. Tishby N, Pereira F, Bialek W: **The information bottleneck method**. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing* 1999:368–77.
8. Slonim N: **The Information Bottleneck: Theory and Applications**. *PhD thesis*, Hebrew University, Jerusalem, Israel 2002.
9. Friedman N, Mosenzon O, Slonim N, Tishby N: **Multivariate Information Bottleneck**. In *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference (UAI-2001)*, San Francisco, CA: Morgan Kaufmann Publishers 2001:152–161.
10. Thompson J, Higgins D, Gibson T: **CLUSTAL W: improving the sensitivity of progressive-multiple sequence alignment through sequence weighting,position-specific gap penalties and weight matrix choice**. *Nucleic Acids Res* 1994, **22**:4673–80.
11. Ron D, Singer Y, Tishby N: **The Power of Amnesia**. In *NIPS*, Volume 6. Edited by Cowan J, Tesauro G, Alspector J, Morgan Kaufmann Publishers, Inc. 1994:176–83.
12. Crooks G, Hon G, Chandonia J, Brenner S: **WebLogo: a sequence logo generator**. *Genome Research* 2004, **14**(6):1188–90.

Figures

Figure 1

Five representations of a part of an alignment of Pepsin A precursor P00790: (a) probabilistic profile; (b) sequence logo [12]; (c) four textual representations. The IB sequence is more compact than profiles or logos, but retains much of the conservation information lost by other textual formats. In the IB sequence, uppercase letter `X` represents strong conservation ($\sim 80\%$) of amino acid `X`, while lowercase `x` represents low conservation ($\sim 50\%$) of `X`.

Figure 2

Graphical model representations of multivariate and univariate information bottleneck showing input (dashed) and output (solid) conditional dependencies.

Figure 3

Sequence logos for $|C| = 20, 40, 80$, showing several features of IB discretization. First, variable numbers of clusters are assigned to different amino acids according to their overall frequencies: `A` and `L` are more common, while `C` is least common. Second, clusters capture strongly- and weakly-conserved variants, as well as some chemical similarities: `I`, `V`, `L`, and `M` are all hydrophobic.

Figure 4

Left: Information versus sequence type for consensus sequence, profiles, and IB without priors. Right: $I(Y;X) - I(Y;C)$ as a function of w for different groups of priors. The information loss for 52 categories without priors is 0.359, for 10, 0.474.

Figure 5

Running times for profile-profile and IB-profile alignment, and (twice) running time for IB discretization. Alignments were performed using the Smith-Waterman algorithm and computing the complete dynamic programming matrix. For IB, each sequence was first discretized using 50 categories. For profiles, distances were precomputed between every pair of sequence positions.

Figure 6

Left: Alignment differences for IB models and sequence alignment, within and between superfamilies. Right: ROC curve for same/different superfamily classification by alignment score. 52 IB categories are used throughout.

Figure 7

Pseudocode for the iterative IB algorithm.

Figure 8

Pseudocode for the sequential IB algorithm.

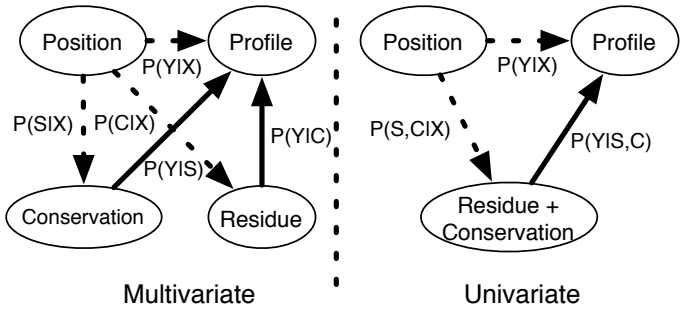
A	0.0	0.0	0.0	0.09	0.34	0.23	0.12	0.0	0.0	0.0
C	0.0	0.0	0.0	0.04	0.01	0.01	0.03	0.0	0.0	0.0
D	0.0	0.0	1.0	0.01	0.05	0.14	0.09	0.0	1.0	0.0
E	0.0	0.0	0.0	0.38	0.04	0.00	0.04	0.0	0.0	0.0
F	0.0	0.0	0.0	0.06	0.00	0.08	0.04	0.0	0.0	1.0
G	0.0	0.0	0.0	0.00	0.06	0.01	0.03	1.0	0.0	0.0
H	0.0	0.0	0.0	0.02	0.00	0.04	0.00	0.0	0.0	0.0
I	0.0	0.0	0.0	0.00	0.00	0.03	0.00	0.0	0.0	0.0
K	0.0	0.0	0.0	0.04	0.01	0.01	0.00	0.0	0.0	0.0
L	0.0	0.0	0.0	0.01	0.01	0.00	0.09	0.0	0.0	0.0
M	0.0	0.0	0.0	0.00	0.00	0.03	0.00	0.0	0.0	0.0
N	0.5	1.0	0.0	0.05	0.05	0.01	0.01	0.0	0.0	0.0
P	0.0	0.0	0.0	0.02	0.00	0.23	0.00	0.0	0.0	0.0
Q	0.0	0.0	0.0	0.04	0.05	0.00	0.00	0.0	0.0	0.0
R	0.0	0.0	0.0	0.04	0.01	0.00	0.00	0.0	0.0	0.0
S	0.5	0.0	0.0	0.16	0.10	0.06	0.29	0.0	0.0	0.0
T	0.0	0.0	0.0	0.02	0.10	0.05	0.20	0.0	0.0	0.0
V	0.0	0.0	0.0	0.00	0.14	0.03	0.04	0.0	0.0	0.0
W	0.0	0.0	0.0	0.00	0.00	0.00	0.00	0.0	0.0	0.0
Y	0.0	0.0	0.0	0.01	0.00	0.04	0.04	0.0	0.0	0.0

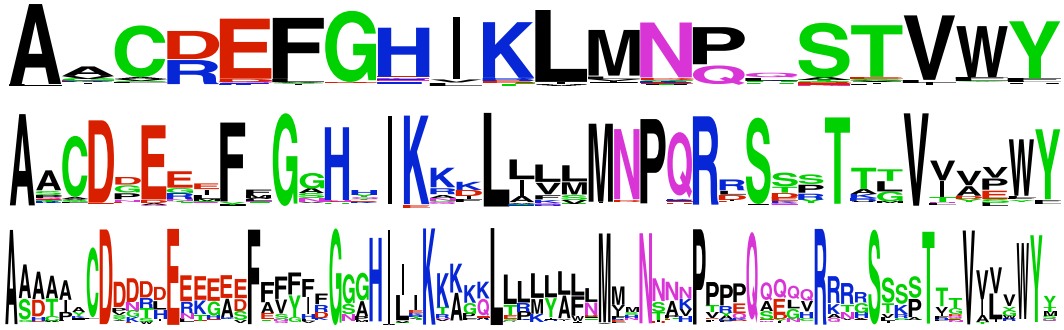
(a)

(b)

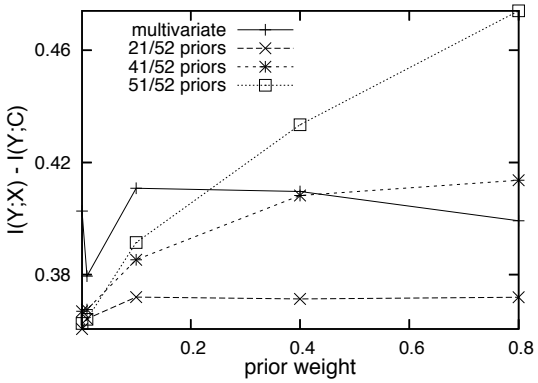
P00790 Seq.:	---EAPT---
Consensus Seq.:	NNDEAASGDF
IB Seq.:	NNDeaptGDF
Logo:	

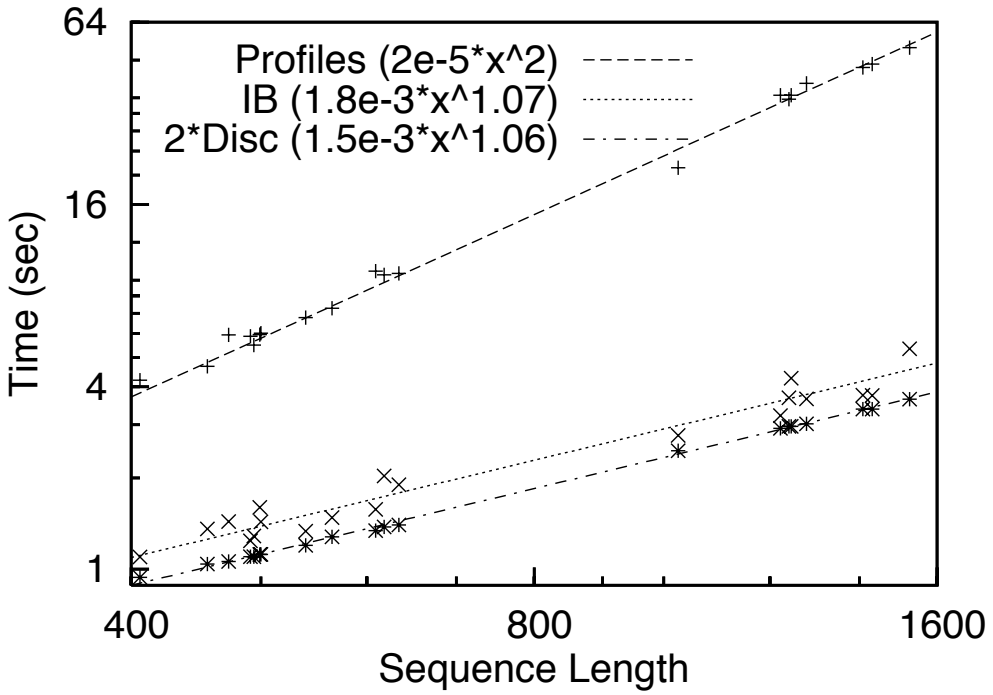
(c)


Position
Profile
P(YIX)
P(SIX)
P(CIX)
P(YIC)
P(YIS)
Conservation
Residue
Multivariate
Position
Profile
P(YIX)
P(S,CIX)
P(YIS,C)
Residue +
Conservation
Univariate

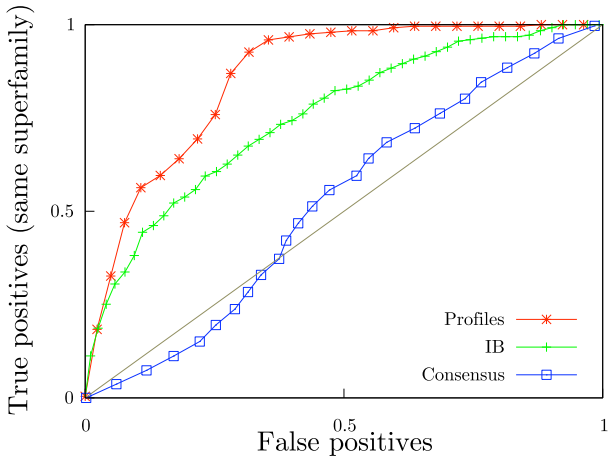


Seq. type	$I(Y;C)$	$I(C;X)$
Consensus	2.8503	
$\|C\| = 40$	3.0596	5.0793
$\|C\| = 80$	3.2083	5.7160
$\|C\| = 160$	3.3248	6.2442
$\|C\| = 320$	3.3986	6.6517
$\|C\| = 500$	3.4267	6.8297
Profiles	3.643	




Profiles (2e-5*x^2)
IB (1.8e-3*x^1.07)
2*Disc (1.5e-3*x^1.06)
Time (sec)
64
16
4
1
400
800
1600
Sequence Length

	Edit distance	Score change
	Same Superfamily	
mIB	0.154 ± 0.182	0.086 ± 0.166
IB	0.170 ± 0.189	0.107 ± 0.198
BLOSUM	0.390 ± 0.065	
	Same Clan	
mIB	0.124 ± 0.209	0.019 ± 0.029
IB	0.147 ± 0.232	0.022 ± 0.037
BLOSUM	0.360 ± 0.062	



Randomly initialize $p(c \mid x)$
Find the corresponding $p(c)$, $p(y \mid c)$ through Eqs. (4)
repeat

$p_{i+1}(c|x) \leftarrow \frac{p_i(c)}{Z_{i+1}(x,\beta)} \exp\left(-\beta D_{KL}[p_i(y|x)||p_i(y|c)]\right)\ ,\ \forall\, c \in C,\ \forall\, x \in X$

if hard clustering,

$$p_{i+1}(c|x) = \begin{cases} 1 & \text{if } c = \text{argmax}_c\, p(c|x) \\ 0 & \text{otherwise} \end{cases}$$

endif

$p_{i+1}(c) \leftarrow \sum_x p(x) p_{i+1}(c|x)\ ,\ \forall\, t \in C$

$p_{i+1}(y|c) = \frac{1}{p_{i+1}(c)} \sum_x p_{i+1}(c|x) p(x,y)\ ,\ \forall\, c \in C,\ \forall\, y \in Y$

until (stopping criterion)

$C \leftarrow$ random partition of X into K clusters
while not *done*
  $done \leftarrow TRUE$
  for every $x \in X$:
    Remove x from current cluster $c(x)$
    $c'(x) \leftarrow \text{argmin}_{c \in C} \Delta L(\{x\}, c)$
    if $c'(x) \neq c(x)$
      $done \leftarrow FALSE$.
    Merge x into $c'(x)$
  end for
end while