Using Mutual Information Content of Protein Sequences for Classification

Chris Hemmerich

March 28, 2005

1 Abstract

Protein sequence classification is a challenging problem. We are attempting to use the Mutual Information Content of protein sequences to provide an alternative method of classification. Since the 3D structure, and ultimately classification depends on the relationship between amino acids within a sequence, measuring the dependency of acids at different distances in a sequence may prove a valuable classification tool. This document contains some preliminary results, but is primarily a description of methods used to this point.

2 Mutual Information Content

The goal of analyzing Information Content is to provide an alternative method of measuring the relatedness of protein sequences. We hope to use the protein sequence to generate numerical data that is useful for comparing sequences. Converting a character sequence to numerical data opens up many possibilities for using existing areas of study, including machine learning techniques, to study biological sequences,

The method being tested for scoring sequences is to traverse the protein sequence from beginning to end, and at each position record the protein composition of a certain number of blocks, each separated by a set distance, termed the gap length. The simplest version of this involves two blocks, each of a single amino acid. Using these parameters, the protein sequence

KTCVV

would generate the following pairs for the given gap lengths

0	1	2	3
KT	KC	KV	KV
TC	TV	TV	
CV	CV		
VV			

Once this information has been processed, you can determine how significant each pair is by comparing the number of occurrence of that pair within the sequence to what would be expected by chance. The

$$(P_i P_j)$$

where P_i and P_j are determined by counting bases in the sequence. The mutual information content of a given pair for gap length r is

$$P_{ij}(r) \cdot log_2 \left(\frac{P_{ij}(r)}{P_i P_j} \right)$$

To find the total Information Content Score (I) for a given gap length, sum over a subset of the alphabet of i and j including only the amino acids found in the sequence:

$$I(r) = \sum_{i,j} P_{ij}(r) \cdot log_2 \left(\frac{P_{ij}(r)}{P_i P_j} \right)$$

This score measures how much information knowing the identity of the first amino acid provides you about the second amino acid. In the special case that the positions are independent

$$(P_i P_j) = P_{ij}(r)$$

and the logarithm, and thus I reduces to 0.

2.1 Score Vectors

Computing I for a single gap length is unlikely to differentiate sequences enough to provide for successful classification. Considering multiple gap lengths allows us to increase the dimensionality of the search space. These scores can be stored in a vector that can represents a coordinate in a space with dimensionality equal to the length of the vector.

$$V = (x, y, z, ...)$$

By using a continuous set of gap lengths that begin with 0, you can generate a score vector, where the a score is retrieved by accessing the vector at the index corresponding to the desired gap length. The length of the vector is only limited by the length of the sequence being analyzed.

Intuitively, smaller values of r should have more deterministic power than larger values, as amino acids near each other on a sequence are generally more strongly dependent on each other than remote amino acids. Preliminary tests of the classification ability of these scores show that this is the case.

Once a vector of I values has been obtained for a set of sequences or families, any two vectors can then be compared with the following equation.

$$[V_1 - V_2] = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + ...}$$

This equation determines the distance between two vectors in a space containing a number of dimensions equal to the number of r values contained in a single vector. For this equation to function correctly, the vectors being compared must contain the same r values. The vector can be of any length, and the values of r used do not need to be in any particular order or contiguous.

2.2 Increasing Block Count

This method of calculating information content is not limited to pairs of amino acids. Larger groupings contain more positional information, and information content. However larger groups result in a smaller number of groups to work with. We hope to find the point that maximizes our ability to classify a protein sequence.

We have looked at amino acid triplets, defined as three acids where aa 1 and 2 are separated by a gap of length r and aa 2 and 3 are also separated by a gap of length r. For example, the following sequence shows 3 blocks with a gap length of 2.

LWWGNPGFL

The information content equation can be expanded to calculate scores for triplets.

$$I(r) = \sum_{i,j,k} P_{ijk}(r) \cdot log_2 \left(\frac{P_{ijk}(r)}{P_i P_j P_k} \right)$$

The equations can be expanded beyond three blocks as well, by increasing the number of block variables, but is limited by the length of the protein sequence being studied.

2.3 Increasing Block Width

Another potential method for gaining additional information content is to increase the width of the blocks considered beyond a single amino acid, to include multiple adjacent amino acids. For example, the following sequence shows 4 blocks with a block width of 3 and a gap length of 3.

QLMWKIAPPFLAFLTYSGDVTGVVM

The equation for the information content of this configuration would be:

$$I(r) = \sum_{I,J,K,L} P_{IJKL}(r) \cdot log_2 \left(\frac{P_{IJKL}(r)}{P_I P_J P_K P_L} \right)$$

Where I,J,K, and L represent all blocks of three adjacent amino acids found in the sequence.

It is worth noting that only exact matches between blocks are considered by this program, and there is probably room for improvement if a more sophisticated system could be used to allow some degree of error within this match. As we are summing over a dictionary, allowing fuzzy matches would be considerably more complicated.

2.4 Gap Handling

When dealing with aligned protein sequences, gaps and other irregularities may have been introduced. In the instance where a sequence gap occurs within a block, that iteration of the algorithm is discarded. Sequence gaps between blocks do not affect the results. Blocks with Gaps in them are not counted when determining the probability of a block occurring randomly in a sequence, therefore, even in a gapped sequence

$$\sum_i P_i = 1$$

This decision leads to artificially high I scores when gaps significantly reduce the number of pairing considered. For example, when working with block size 5 and block count 4, there maybe 60 ungapped blocks of length 5, but only 7 ungapped sets of 4 blocks. Under this condition

$$P_{ij} >> P_i P_j$$

2.5 Score Parameters and Sequence Length

As the block width (w), block count (c), and gap length (r) increase, the length of a single entity considered by this algorithm can become prohibitively long. The length can be determined by the following equation

$$L = (w \cdot c) + (r \cdot (c - 1))$$

Incomplete entities are ignored in the calculations, so the number of considered matches in an ungapped sequence of length S is:

$$(S + 1) - L$$

3 Results

Formal results are forthcoming, but test among 5-20 protein families encompassing 200-20,000 sequences, shows a success rate above 90% in most cases. In cases so far, the dimensionality can be paired down greatly and still give results above 90%. Chosing a subset of dimensions that work well in all cases is a top priority. A machine learning toolset called Weka was used for these computations, which is discussed in more detail in the next section.

4 Tools and Procedures

4.1 PFAM

The database of Protein Families and HMMs is a collection of protein domains and conserved protein regions. The database is quite large, including over 5000 families, some of which contain tens of thousands of sequences. I work with a hand-curated subset of PFAM called the PFAM-A.seed collection, which is a small representative group of the family. A multiple sequence alignment of these seed sequences is used to generate the full family.

Sequences in the PFAM database have already been aligned, had gaps inserted, and been trimmed to best fit into their family.

4.2 PostgreSQL

The portion of the PFAM database I was working with is available for download in text file format. This file is quite large, and retrieving sequences from it could be quite slow. Additionally, from testing different families within PFAM, I was generating a lot of different score vectors and having trouble keeping them organized. To resolve, this I designed a relational database to store PFAM families, sequences, and pre-generated score vectors for 20 variations of block count and width for each sequence in PFAM. I wrapped the tables with perl modules for easy manipulation of the data.

With this database, I can quickly perform tests on new families, and test different combinations of block parameters. The goal is to be able to automate large scale tests more easily.

4.3 Weka

We are using a java-based machine learning package, called Weka for our classification tasks. Current work has involved only k-nearest neighbor classification, although more robust methods may prove better. Weka provides automated tools for data normalization, random extraction of test data from training data, and coss-validation. Weka is freely available at http://www.cs.waikato.ac.nz/ ml/weka/

5 Current and Future Work

I am currently analyzing the quality of the various gap, width, count scores, looking to remove noisy dimensions. I expect to be able to generate comparable results with a small subset of the 400 scores computed for each sequence.

I am also considering different methods of modeling the prior probabilities of sequence blocks. Using the frequency of blocks within the entire PFAM database rather than a single sequence is one possible method. Another option is generate a different frequency for each species.

The current method of handling gaps is computationally easy, but the gaps introduce significant changes to a sequence score, so that the I values of a

heavily gapped sequence would compare poorly with an ungapped, but highly homologous sequence.

References

[1] The Pfam protein familes database
A. Bateman, E. Birney, et al.
Nucleic Acids Research, 30(1):276-280, 2002.

[3] Occurrence Probability of Structured Motifs in Random Sequences
S. Robin, J.-J. Daudin, et al.
Journal of Computational Biology, 9(6):761-773, 2002.