Automated Alphabet Reduction Method with Evolutionary Algorithms for Protein Structure Prediction

Biological Applications Track

Jaume Bacardit*
ASAP research group, School of Computer Science and IT, University of Nottingham, Jubilee Campus, Nottingham, NG8 1BB, UK
jqb@cs.nott.ac.uk

Michael Stout
ASAP research group, School of Computer Science and IT, University of Nottingham, Jubilee Campus, Nottingham, NG8 1BB, UK
mqs@cs.nott.ac.uk

Jonathan D. Hirst
School of Chemistry, University of Nottingham, University Park, Nottingham NG7 2RD, UK
jonathan.hirst@nottingham.ac.uk

Kumara Sastry
Illinois Genetic Algorithms Laboratory (IlliGAL), Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, Urbana IL 61801
ksastry@uiuc.edu

Xavier Llorà
National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, 1205 W.Clark Street, Urbana, IL 61801, USA
xllora@uiuc.edu

Natalio Krasnogor†
ASAP research group, School of Computer Science and IT, University of Nottingham, Jubilee Campus, Nottingham, NG8 1BB, UK
nxk@cs.nott.ac.uk

ABSTRACT

This paper focuses on automated procedures to reduce the dimensionality of protein structure prediction datasets by simplifying the way in which the primary sequence of a protein is represented. The potential benefits of this procedure are faster and easier learning process and generation of more compact and human-readable solutions. This simplification consists of an alphabet reduction procedure to map the 20-letter Amino Acid (AA) alphabet into a much lower cardinality alphabet by grouping similar AA types. This procedure is guided by a fitness function based on the Mutual Information between the AA-based input attributes of the dataset and the PSP feature that is being predicted. To search for the optimal reduction, the Extended Compact Genetic Algorithm was used, and afterwards the results of this process were validated by learning the reduced dataset with BioHEL, a genetics-based machine learning algorithm that induces sets of rules. The results are mixed; we succeed in reducing the alphabet size to three which leads to faster computation and more compact rules. However, the accuracy suffers slightly although the difference is not statistically significant when compared to the performance obtained from learning the full AA alphabet of 20 symbols, based on the protein-wise accuracy performance metric.

*Corresponding author
†Corresponding author

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Concept Learning, Induction*; G.1.6 [**Numerical Analysis**]: Optimization; J.3 [**Computer Applications**]: Life and Medical Sciences—*Biology and Genetics*

General Terms

Algorithms, Experimentation, Performance

Keywords

Evolutionary Algorithms, Estimation of Distribution Algorithms, Learning Classifier Systems, Rule Induction, Bioinformatics, Protein Structure Prediction, Coordination Number Prediction, Alphabet Reduction

1. INTRODUCTION

One of the main open problems in computational biology is the prediction of the 3D structure of protein chains (PSP). This domain usually requires enormous amounts of computational resources due to its size and difficulty. For instance, one of the PSP predictors that obtained top results in the last CASP (Critical Assessment of Techniques for Protein Structure Prediction) experiment was the Rosetta@home system [24] which used a massive collaborative computing system to predict protein structures with up to 10000 computing hours per protein.

One of the possible ways in which the dimensionality of this problem can be alleviated is by reducing the size of the alphabet of the variables that are involved in this problem and also in its subproblems, such as the prediction of protein secondary structure, solvent accessibility or coordination number (CN). An example of a high cardinality alphabet that can have its size reduced is the primary sequence of a protein that consist of a 20-letter alphabet, the 20 amino acids (AA).

An example of a widely explored dramatic alphabet reduction option is to transform the 20 letters AA alphabet into a two letters hydrophobic/polar (HP) alphabet. This reduction is usually followed by constraining the residue locations of the predicted protein to those of a 2D/3D lattice [32, 17], although sometimes this HP alphabet reduction is applied to real non-constrained proteins. A recent paper [29] compared the performance of several learning methods applied to predict the coordination number for lattice-based proteins, real proteins with HP alphabet and real proteins with AA alphabet. The experiments showed that there is a significant although not big performance difference between the HP and AA alphabets. Moreover, the criterion to divide the AA types between hydrophobic and polar was the one of Broome and Hecht [8], although there are alternative HP assignment policies [20] as well as real-valued hydrophobicity scales [9].

These facts prompt several questions. Is is possible to obtain statistically similar performance between the original AA alphabet and another alphabet with reduced number of symbols? Moreover, can we tailor the alphabet reduction criteria to each specific problem we want to solve? The aim of these paper is to answer these two questions. We propose an automated method to perform alphabet reduction. This method uses the Extended Compact Genetic Algorithm [14] to optimize the distribution of the 20 letters of the AA alphabet into a predefined number of categories. We apply this alphabet reduction to the coordination number prediction domain. For the fitness function of such reduction process we have chosen a rigorous information theory measure, the Mutual Information [10]. This measure relates the dependence between two variables, in this case the input attributes and the predicted class of the coordination number domain. By optimizing this measure we are looking for the alphabet reduction criterion that maintains as much as possible the useful information existing in the input attributes related to the predicted feature.

We have performed experiments trying to reduce the AA alphabet into two, three for and five groups, and verified the performance of the reduction criteria found by this optimization process by learning the reduced dataset, and comparing the predictive performance to the one obtained by learning the original 20-letters alphabet. The learning process used BioHEL (Bioinformatics-Oriented Hierarchical Evolutionary Learning) [3], a recent Learning Classifier System based on the Iterative Rule Learning [31] approach. We have also analyzed the relation between the resulting groups of attributes and some standard categories in which the amino acids can be grouped [6].

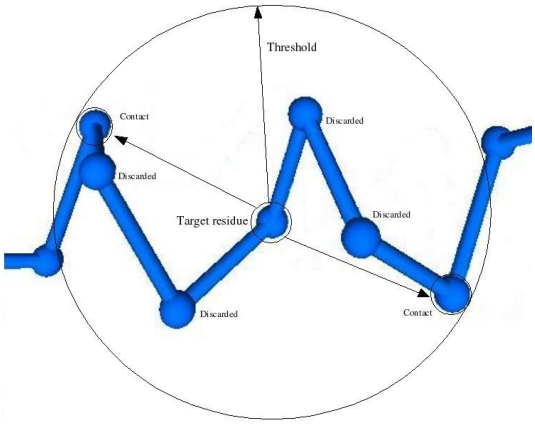
The rest of the paper is structured as follows: Section 2 will contain a brief summary of background information and related work. Section 3 will describe all techniques used for the optimization and learning stages of the experiments reported in the paper, while section 4 will describe the dataset and experimental setup. Section 5 will report the results of the experiments. Finally, section 6 will describe the conclusions and further work.

2. BACKGROUND AND RELATED WORK

Proteins are heteropolymer molecules constructed as a chain of residues or amino acids of 20 different types. This string of amino acids is known as the primary sequence. In its native state, the chain folds to create a 3D structure. It is thought that this folding process has several steps. The first step, called secondary structure, consists of local structures such as alpha helix or beta sheets. These local structures can group in several conformations or domains forming a tertiary structure. Secondary and tertiary structure may form concomitantly. The final 3D structure of a protein consists of one or more domains.

In this context, the coordination number of a certain residue is a profile of this folding process indicating the number of other residues that, after the folding process, end up being near the target residue. Some of these contacts can be close in the protein chain but other can be quite far apart. Some trivial contacts such as those with the immediate neighbour residues are discarded. Figure 1 contains a graphical representation of the CN of a residue for an alpha helix, given a minimum chain separation (discarded trivial contacts) of two. In this example, the CN is two. This problem is closely related to contact map (CM) prediction that predicts, for all possible pairs of residues of a protein, if they are in contact or not.

Figure 1: Graphical representation of the CN of a residue



There is a large literature in CN and CM prediction, in which a variety of machine learning paradigms have been used, such as linear regression [16], neural networks [5], hidden markov models [28], a combination of self-organizing maps and genetic programming [19] or support vector machines [33].

The research in alphabet reduction techniques applied to PSP is quite active in recent years [21, 22, 23]. One example of a previous application of an Evolutionary Algorithm for alphabet reduction is [22]. In this case, a GA was used to optimize a reduction into 5 letters applied to sequence alignment. The fitness function was based on maximizing the difference between the sequence identity of the training alignments and a set of random alignments based on the same sequences.

Another approach [23] applies Mutual Information to optimize alphabet reduction for contact potentials. This approach unlike ours is tailored to the problem being solved because the mutual information is computed between AA types (or AA groups) potentially in contact, instead of being a generic process connecting the inputs and the output of the domain.

Finally, Meiler et al. [21] propose a slightly different approach. Instead of treating the AA types at a symbolic level, they characterize them numerically with several physical features and then apply a neural network to generate a lower dimensionality representation of these features.

3. EVOLUTIONARY COMPUTATION METHODS AND FITNESS FUNCTION

The experiments reported in this paper have two stages. In the first one, ECGA is used to optimize the alphabet reduction mapping given the number of symbols of the final alphabet by using Mutual Information as fitness function. In the second stage, BioHEL is used to validate the goodness of the alphabet groupings found by ECGA by learning the reduced dataset.

3.1 Extended Compact Genetic Algorithm

Extended Compact Genetic Algorithm [14] is an optimization method belonging to the family of Estimation of Distribution Algorithms [18]. This method iteratively optimizes a population of candidate solutions by first heuristically estimating the structure of the problem being solved and later recombining the population based on this structure. The structure of the problem in the specific case of ECGA is defined as non-overlapping groups of variables that interact among them. A greedy approach using a fitness function based on the Minimum Description Length (MDL) principle [26] applied to a type of probabilistic models called Marginal Product Models is used to find these groups of variables. The recombination of the population uses uniform crossover [30] based on the identified structure of the domain. Specifically we use the χ-ary version of ECGA [11] and the code is an slightly modified version of the one available at `ftp://www-illigal.ge.uiuc.edu/pub/src/ECGA/chiECGA.tgz`. The default parameters of ECGA were used except for the population size that was set to 10000 individuals when optimizing the alphabet reduction for two or three letters, and 40000 when optimizing for four or five letters.

3.2 The BioHEL learning system

BioHEL (Bioinformatics-oriented Hierarchical Evolutionary Learning) is Genetics–Based Machine Learning (GBML) system following the Iterative Rule Learning or Separate-and-Conquer approach [13], first used in the GBML field by Venturini [31]. BioHEL is strongly influenced by GAssist [1] which is a Pittsburgh GBML system. Several of BioHEL features have been inherited from GAssist. The system applies an almost standard generational GA, which evolves individuals that are rules for the problem that we are solving.

The final solution of the learning process is a set of rules that is obtained by applying iteratively a GA. After each rule is obtained, the training examples that are covered by this rule are removed from the training set, to force the GA of the next iteration to explore other areas of the search space. The rules are inserted into a rule set with an explicit default rule that covers the majority class of the domain. The evolved rules will cover all the other classes. Therefore, the stopping criteria of the learning process is when it is impossible to find any rule where the associated class is not the majority class of the matched examples. When this happens, all remained examples are assigned to the default rule. Also, several repetitions of the GA with the same set of instances are performed, and we will only insert in the rule set (and therefore remove examples from the training set) the best rule from all the GA runs. Figure 2 contains the C++ code of the general workflow of BioHEL.

Each individual is a rule, which consists of a predicate and an associated class. We use the GABIL [12] knowledge representation for the predicates of these rules. The system also uses a windowing scheme called ILAS (incremental learning with alternating strata) [2] to reduce the run-time of the system, especially for dataset with hundreds of thousands of instances, as in this paper. This mechanism divides the training set into several non-overlapping subsets and chooses a different subset at each GA iteration for the fitness computations of the individuals.

The fitness function of BioHEL is based on the Minimum Description Length (MDL) principle [26]. The MDL principle is a metric applied to a theory (a rule) which balances its complexity and accuracy. BioHEL MDL formula is adapted from GAssist one as follows:

$$Fitness = TL \cdot W + EL \tag{1}$$

where TL stands for theory length (the complexity of the solution) and EL stands for exceptions length (the accuracy of the solution). This fitness function has to be minimized.

W is a weight that adjusts the relation between TL and EL. BioHEL uses the automatic weight adjustment heuristic proposed for GAssist [1]. The parameters of this heuristic are adjusted as follows: Initial TL ratio: 0.25, weight relax factor: 0.90, max iterations without improvement: 10.

TL is defined as follows:

$$TL(R) = \frac{\sum_i = 1^{NA} NumZeros(R_i)/Card_i}{NA} \tag{2}$$

where R is a rule, NA is the number of attributes of the domain, R_i is the predicate of rule R associated to attribute i, NumZeros counts the number of bits set to zero for a given predicate in GABIL representation and $Card_i$ is the cardinality of attribute i. TL always has a value between 0 and 1. It has been designed in this way in order to simplify the tuning of W. The number of zeros in the GABIL predicates are a measure of specificity. Therefore, promoting the minimization of zeros means promoting general and thus less complex rules.

The design of EL has to take into account that we have to achieve an equilibrium between accuracy and coverage. We have to promote rules that cover as much examples as possible without sacrificing accuracy. In order to achieve this objective, we will design a coverage measure that strongly promotes covering a certain minimum of examples, but that reduces its effect after the coverage has surpassed this threshold. The measure is defined as follows:

Figure 2: General workflow of BioHEL

```
instanceSet *is=new instanceSet(argv[2], TRAIN);
classifier_aggregated ruleSet;
classifierFactory cf;

do {
        classifier *best=NULL;
        for(int i=0;i<tGlobals->numRepetitionsLearning;i++) {
                classifier *bestIt=runGA();
                if(best==NULL || bestIt->compareToIndividual(best)>0) {
                        if(best) cf.deleteClassifier(best);
                        best=bestIt;
                }

                if(i<tGlobals->numRepetitionsLearning-1) {
                        is->restart();
                }
        }
        if(isMajority(*best)) {
                ruleSet.addClassifier(best);
                is->removeInstancesAndRestart(best);
        } else {
                cf.deleteClassifier(best);
                break;
        }
} while(1);
```

$$EL(R) = 2 - ACC(R) - COV(R) \quad (3)$$

$$ACC(R) = \frac{corr(R)}{matched(R)} \quad (4)$$

$$COV = \begin{cases} MCR \cdot \frac{RC}{CB} & \text{If } RC < CB \\ MCR + (1 - MCR) \cdot \frac{RC - CB}{1 - RC} & \text{If } RC \geq CB \end{cases} \quad (5)$$

$$RC = \frac{matched(R)}{|T|} \quad (6)$$

COV is the adjusted coverage metric that promotes the coverage of at least a certain minimum number of examples, while RC is the raw coverage of the rule. ACC is the accuracy of the rule, $corr(R)$ is the number of examples correctly classified by R, $matched(R)$ is the number of examples matched by R, MCR is the weight given in the coverage formula to achieving the minimum coverage, CB is the minimum coverage threshold and $|T|$ is the total number of training examples. For all the tests reported in the paper, MCR has 0.9 value, and CB has value 0.01.

Finally, we have used a mechanism wrapped over BioHEL to boost its performance. We generate several rule sets using GAssist with different random seeds and combine them as an ensemble, combining their predictions using a simple majority vote. This approach is similar to Bagging [7]. BioHEL used the values for the parameters defined in [1] except for the followings: population size 500; GA iterations 200; repetitions of rule learning process: 2; rule sets per ensemble; 10.

3.3 Mutual Information

The aim of the alphabet reduction optimization is to simplify the representation of the dataset in a way that maintains the underlying information that is really needed for the learning process. Therefore, the fitness function for such a process should give an estimation of what can the reduced input information tell about the output. Ideally, we could simply try to learn the reduced dataset geneated by each individual that we evaluate and use the training accuracy as fitness function, but this option is not feasible due to the enormous computational cost that would requiere. Therefore, we need to use some *cheap* estimation of the relationship between inputs and output as fitness function, and we have chosen the Mutual Information metric for this task.

The mutual information is an information theory measure that quantifies the interrelationship that two discrete variables have among each other [10], that is, how much information can one variable tell about the other one. The mutual information is defined as follows:

$$I(X;Y) = \sum_{y \in Y} \sum_{x in X} p(x,y) log \frac{p(x,y)}{p(x)p(y)} \quad (7)$$

Where $p(x)$ and $p(y)$ are the probabilities of appearance of x and y and $p(x,y)$ is the probability of having x, y at the same time. In our specific case, we use the mutual information to measure the quantity of information that the input variables of the alphabet-reduced dataset have related to the class of the domain. Therefore, for a given instance x will be a string that concatenates the input variables of an instance, and y the associated class of the instance.

4. PROBLEM DEFINITION AND EXPERIMENTAL DESIGN

4.1 Problem definition

The dataset that we have used in this paper is the one identified as *CN1* with two states and uniform length class partition criteria in [4]. Its main characteristics are briefly described as follows:

4.1.1 Coordination number Definition

The CN definition is the one proposed by Kinjo et al. [16]. The distance used is defined using the C_β atom (C_α for glycine) of the residues. Next, the boundary of the sphere around the residue defined by the distance cutoff $d_c \in \Re^+$ is made smooth by using a sigmoid function. A minimum chain separation of two residues is required. Formally, the CN, N_i^p, of residue i in protein chain p is computed as:

$$N_i^p = \sum_{j:|j-i|>2} \frac{1}{1 + exp(w(r_{ij} - d_c))} \quad (8)$$

where r_{ij} is the distance between the C_β atoms of the ith and jth residues. The constant w determines the sharpness of the boundary of the sphere. The dataset used in this paper had a distance cutoff d_c of 10 $\mathring{A}$.

The real-valued definition of CN has been discretized in order to transform the dataset into a classification problem that can be mined by BioHEL. The chosen discretization algorithm is the well-known unsupervised uniform-length (UL) discretization. Moreover, for the experiments of this paper we have used the simplest version of the dataset, dividing the domain into two states, low or high CN.

4.1.2 Protein dataset

We have used the dataset and training/test partitions proposed by Kinjo et al. The protein chains were selected from PDB-REPRDB [25] with the following conditions: less than 30% of sequence identity, sequence length greater than 50, no membrane proteins, no nonstandard residues, no chain breaks, resolution better than 2 Å and a crystallographic R factor better than 20%. Chains that had no entry in the HSSP [27] database were discarded. The final data set contains 1050 protein chains and 257560 residues.

4.1.3 Definition of the training and tests sets

The set was divided randomly into ten pairs of training and test set using 950 proteins for training and 100 for test in each set, using bootstrap. The proteins included in each partition are reported in `http://maccl01.genes.nig.ac.jp/~akinjo/sippre/suppl/list/`.

The definition of the input attributes is the one identified as *CN1* in [4]. The input data will consists of the AA type of the residues in a window around the target one. A window size of 4 (4 residues at each side of the target) has been used. Therefore, each instance consist in 9 nominal attributes of cardinality 21 (the 20 AA types plus the symbol that represents end of chain, in case that the window overlaps with the beginning or the end of a protein chain). Figure 3 contains a representation of the windowing process that creates the instances of the dataset.

4.1.4 Performance measure

In section 5 we will report the performance of the learning process of BioHEL over the reduced datasets in two different ways. On one hand we will use the standard machine learning accuracy metric (#correct examples/#total examples) (called residue-wise accuracy). On the other hand, as usual in the protein structure prediction field [16, 15], we will take into account the fact that each example (a residue) belongs to a protein chain. Therefore, we will first compute the standard accuracy measure for each protein chain, and then average these accuracies to obtain the final performance measure (called protein-wise accuracy). Because different chains have different lengths, residue-wise accuracy and protein-wise accuracy can differ greatly. The rationale for reporting the second measure is to mimic the real-life situation, in which a new protein is sequenced, and researchers are interested in the predicted properties based on the entire protein sequence, independent of its length.

Figure 3: Representation of the windowing process used to generate the instances of the dataset

```
Sequence (AA,CN): (S,0),(K,0),
(Y,0),(V,0),(D,0),(R,0),(V,1),
(I,0),(A,0),(E,0)

            X,X,X,X,S,K,Y,V,D | 0
            X,X,X,S,K,Y,V,D,R | 0
            X,X,S,K,Y,V,D,R,V | 0
            X,S,K,Y,V,D,R,V,I | 0
Instances   S,K,Y,V,D,R,V,I,A | 0
            K,Y,V,D,R,V,I,A,E | 0
            Y,V,D,R,V,I,A,E,V | 1
            V,D,R,V,I,A,E,V,E | 0
            D,R,V,I,A,E,V,E,K | 0
            R,V,I,A,E,V,E,K,K | 0
```

4.2 Experimental design

4.2.1 Steps of the experimental process

The experiments reported in this paper follow these steps:

- For a number of symbols of the final alphabet going from two to five
  1. ECGA is used to find the optimal alphabet reduction based on the MI-based fitness function.
  2. The alphabet reduction policy is applied to the dataset
  3. BioHEL is used to learn the reduced dataset

4.2.2 Representation and fitness function of the alphabet reduction process

The chromosome optimized by ECGA is very simple. It has one gene for each letter of the original alphabet (the 20 AA types plus the end-of-chain symbol) meaning the group of letters where this AA type is assigned. This gene can take a value from the range $0..N-1$, where N is the predefined number of symbols of the reduced alphabet. Figure 4 illustrates an example of such chromosome for a reduction process into two groups.

Figure 4: Representation of the chromosome for the alphabet reduction process for a two-letter reduction

Orig. Alphabet	ACDEFGHIKLMNPQRSTVWXY
Genotype	00110000101111110010
Phenotype	Group 1: $ACFGHILMVWY$ Group 2: $DEKNPQRSTX$

Each fitness computation follows these steps:

1. The reduction mappings are extracted from the chromosome

2. The instances of the training set are transformed into the low cardinality alphabet based on the extracted mappings

3. The mutual information between class attribute and the string formed by concatenating the input attributes is computed

4. This mutual information is the fitness value of the chromosome

5. RESULTS

5.1 Results of the alphabet reduction process

ECGA was used to find reductions to alphabets of two, three, four and five symbols; the results are reported in Table 1. ECGA was used to find alphabet reductions into two, three, four and five symbols alphabet. Table 1 describes the obtained reductions. In order to visualize better what are the physical properties of the groups of AA types obtained by this method, we are going to assign to some of the AA type a colour depending on its properties, the colored groups are also described in table 1. The physical properties used to group in colours the AA types are taken from [6].

Table 1: Alphabet reductions generated by ECGA

#letters	Groups of letters
2	ACFGHILMVWY DEKNPQRSTX
3	ACFILMVWY DEKNPQRX GHST
4	AFHTY CILMV DEKPQX GNRSW
5	AIS CHLV DEPQY FGMWX KNRT

CLV - hydrophobic
AIM - hydrophobic
FWY - aromatic, neutral, hydrophobic
DE - negatively charged
KHR - positively charged

When optimizing for a two letters alphabet, the MI-based optimization process ends up finding two groups of AA types that separate the most hydrophobic residues ($ACFGHILMVWY$) from the rest. Hydrophobicity is one of the main factors in the folding process of proteins, so it is natural that a reduction process into only two symbols is equivalent to identifying the hydrophobic residues.

From the experiment with three-letter alphabets we can observe that one of the groups still contains all-hydrophobic AA types ($ACFILMVWY$) while the other two groups contain more mixed AA types. From the visualization of the obtained groups with colours we can see how only two of the colour groups (CLV and DE) are conserved across all reduced alphabet sizes. The colour mix for four-letter and five-letter alphabets is considerable, meaning that the obtained groups have very mixed physical properties difficult to explain and, as next subsection will show, make the problem more difficult to learn.

5.2 Validation of the obtained reduced alphabets

The reduced alphabets were used to represent the datasets, and then BioHEL was used to learn these datasets. Table 2 contains the test accuracy and average rule-set size of the solutions found by the learning system. As a baseline, results for BioHEL learning the original dataset with full AA type representation (labelled *Orig.*) are also included.

Table 2: Residue-wise accuracy (RWA), Protein-wise accuracy (PWA) ave. rule set size and run-time of BioHEL applied to the reduced datasets. • marks the cases where the reduced dataset had significantly worse performance than the original dataset with AA type representation

#letters	RWA	PWA	#rules	Run-time (s)
Orig.	74.0±0.5	77.0±0.7	22.5±1.8	6435.5±462.3
2	72.3±0.5•	75.8±0.7•	11.3±0.6	3351.8±195.8
3	73.0±0.6•	76.4±0.7	16.7±1.4	4899.6±363.5
4	72.6±0.6•	76.1±0.8	15.4±1.3	4547.6±361.9
5	72.0±0.6•	75.7±0.8•	14.6±1.5	4362.2±424.2

Moreover, these results were analyzed using statistical t-tests (95% conf.) to determine if the accuracy differences were significant, using the Bonferroni correction for multiple comparisons. The test determined that the original dataset with full AA representation significantly outperformed the representations with two and five letter alphabets for protein-wise accuracy and all of the reduced datasets for residue-wise accuracy.

All the reduced datasets could be learned in less time, for the 2-letter alphabet, the run time was almost 50% of the time needed to learn the dataset using the AA type representation. Also, the rule sets obtained from all the reduced alphabets were more compact than the dataset using the AA type representation. Figure 5 contains a rule set obtained from the original dataset with full AA representation, while figure 6 contains a rule-set using the 2-letter alphabet, being much more simple and human-readable.

Figure 5: Rule-set obtained by BioHEL over the dataset with full AA type representation. $AA_{\pm X}$ means AA type for residue in position $\pm X$ in respect to the target residue

1:If $AA_{-4} \notin \{E,N,Q,R,W\}$, $AA_{-3} \notin \{D,E,N,P,Q,R,S,X\}$, $AA_{-2} \notin \{E,P,S\}$, $AA_{-1} \notin \{D,E,G,K,N,P,Q,T\}$, $AA \in \{A,C,F,I,L,M,V,W\}$, $AA_1 \notin \{D,E,G,P,Q\}$, $AA_2 \notin \{D,H,K,P\}$, $AA_3 \notin \{D,E,K,N,P,Q,R,S,T\}$, $AA_4 \in \{A,C,F,G,I,L,M,V\}$ then class is 1
2:If $AA_{-4} \notin \{T,X\}$, $AA_{-3} \notin \{E,N\}$, $AA_{-2} \notin \{E,K,N,Q,R,S,T\}$, $AA_{-1} \notin \{D,E,G,K,N,P,Q,R,S\}$, $AA \in \{C,F,I,L,V\}$, $AA_1 \in \{C,F,G,I,L,M,V,W,X,Y\}$, $AA_2 \notin \{E,K,N,P,Q,R\}$, $AA_3 \notin \{E,K,P,R\}$, $AA_4 \notin \{E,K,Q,X\}$ then class is 1
.
.
.
18:If $AA_{-4} \notin \{E,K,N,P,X\}$, $AA_{-3} \in \{G,I,L,M,V,W,X,Y\}$, $AA_{-2} \notin \{D,E,K,N,P,Q,R,S\}$, $AA_{-1} \notin \{E,K,N,P,Q,R\}$, $AA \notin \{D,E,K,N,P,Q,R,S,T\}$, $AA_1 \notin \{D,E,K,L,Q\}$, $AA_2 \notin \{D,E,L\}$, $AA_3 \notin \{D,K,M,N,P,Q,T\}$, $AA_4 \notin \{C,N,T,X\}$ then class is 1
19:Default class is 0

It was expected, based on earlier existing work [29] that the reduction into two groups would suffer from a significant performance hit, as the alphabet reduction is simplifying dramatically the data and thus losing information. Moreover, as the number of groups increases, we should see a reduction of this performance drop. However, the experimental results show how the reduced dataset with higher performance is the one with three groups. A higher number of symbols does not help increasing the performance.

Figure 6: Rule-set obtained by BioHEL over the dataset with 2-letter alphabet representation. Letter 0 = ACFGHILMVWY, Letter 1 = DEKNPQRSTX. $AA_{\pm X}$ means the group of AA types for residue in position $\pm X$ in respect to the target residue

1:If $AA_{-1} \in \{0\}$, $AA \in \{0\}$, $AA_1 \in \{0\}$, $AA_2 \in \{0\}$, $AA_3 \in \{0\}$, $AA_4 \in \{0\}$ then class is 1
2:If $AA_{-3} \in \{0\}$, $AA_{-2} \in \{0\}$, $AA_{-1} \in \{0\}$, $AA \in \{0\}$, $AA_3 \in \{0\}$, $AA_4 \in \{0\}$ then class is 1
.
.
.
10:If $AA_{-3} \in \{0\}$, $AA \in \{0\}$, $AA_1 \in \{0\}$, $AA_2 \in \{0\}$, $AA_3 \in \{0\}$ then class is 1
11:Default class is 0

Why is this? Our hypothesis is that the Mutual Information measure is not a fitness measure robust enough for this dataset. Table 3 contains the number of unique instances (inputs+output) and unique input vectors for the different reductions of the training fold 0 of the dataset, which contains 234638 instances. We can see how the datasets with four or five letters present a unique number of instance or inputs with is at least 64% of the dataset, meaning that there are many chances that there is a single instance representing certain unique inputs (which means that $p(X)$ in the MI formula becomes 1/234638 in most cases).

Mutual Information needs redundancy in order to estimate properly the relation between inputs and outputs, and there is almost no redundancy in the signal generated by the dataset. If the fitness function cannot provide appropiate guidance we cannot rely much on the obtained results. This is probably the reason why it is difficult to extract physical explanations from the groups of AA types that ECGA finds and, therefore, why it is not possible to obtain good performance when learning the reduced dataset. In this reduction process we have lost too much information by forming wrong groups of AA types.

Table 3: Counts of unique instances and unique input vectors for the training fold 0 of the dataset

# letters	Unique inputs	Unique instances
2	512	1024
3	19254	33839
4	150914	175156
5	219943	224747

6. CONCLUSIONS AND FURTHER WORK

This paper has studied an information theory based automated procedure to perform alphabet reduction for protein structure prediction datasets that use an Amino Acid (AA) alphabet for its attributes. Several groups of AA types share some physical and chemical properties among them and therefore could be grouped into a single category thus reducing the dimensionality of the data that has to be learned.

This procedure uses an Estimation of Distribution Algorithm, ECGA to optimize the distribution of the AA types into a predefined number of groups, using the Mutual Information metric as fitness function applied over the training set of the data that is being reduced. After this procedure, an Evolutionary Computation based learning system (BioHEL) was used to learn the reduced datasets to validate is the reduction process was correct or not.

Our experiments showed that it is possible to perform a reduction into a new alphabet with only three letters that has a performance lower but not significantly different than the performance obtained by learning a 20 letters alphabet when using the protein-wise accuracy metric. We think that this metric measure is more relevant than the flat residue-wise accuracy, because it show the goodness of the reduction in a more broad context with proteins of different lengths. Therefore we think that obtaining a three-letters alphabet with similar performance than a full AA type representation, even if it is only at a protein-wise level, has already some merit. Moreover the learning process for all the reduced datasets is faster and the obtained rule-sets are more compact and human-readable.

However, it was not possible to find appropriate groups of AA types when increasing the number of groups to four or five groups, because the dataset does not have enough number of instances for the Mutual Information to provide a reliable fitness function. Our automated alphabet reduction procedure showed some promising performance, and if we are able to increase the robustness of the fitness function, it has the potential to be a very useful tool to simplify the learning process of several datasets related to protein structure prediction.

Therefore, the first step of further work into this line of research is to adjust the Mutual Information based fitness function or find a suitable alternative. Also, it would be interesting to test this procedure in other datasets beside Coordination Number prediction to check two issues: (1) to see how general is this procedure and (2) to check if the reduction groups found by the procedure differ between domains. If this is true, it will be an answer of why this kind of automated alphabet reduction procedures are necessary. It would also be interesting to learn the reduced datasets with other kind of machine learning methods to check how well do they react to the dimensionality reduction performed by this alphabet reduction process.

7. ACKNOWLEDGMENTS

We acknowledge the support of the UK Engineering and Physical Sciences Research Council (EPSRC) under grants GR/T07534/01, GR/62052/01 and GR/S64530/01 and the Biotechnology and Biological Sciences Research Council (BBSRC) under grant BB/C511764/1. We are grateful for use of the University of Nottingham's High Performance Computer.

8. REFERENCES

[1] J. Bacardit. *Pittsburgh Genetics-Based Machine Learning in the Data Mining era: Representations, generalization, and run-time*. PhD thesis, Ramon Llull University, Barcelona, Catalonia, Spain, 2004.

[2] J. Bacardit, D. Goldberg, M. Butz, X. Llorà, and J. M. Garrell. Speeding-up pittsburgh learning classifier systems: Modeling time and accuracy. In *Parallel Problem Solving from Nature - PPSN 2004*, pages 1021–1031. Springer-Verlag, LNCS 3242, 2004.

[3] J. Bacardit and N. Krasnogor. Biohel: Bioinformatics-oriented hierarchical evolutionary

learning. Nottingham eprints, University of Nottingham, 2006.

[4] J. Bacardit, M. Stout, N. Krasnogor, J. D. Hirst, and J. Blazewicz. Coordination number prediction using learning classifier systems: performance and interpretability. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 247–254. ACM Press, 2006.

[5] P. Baldi and G. Pollastri. The principled design of large-scale recursive neural network architectures dag-rnns and the protein structure prediction problem. *Journal of Machine Learning Research*, 4:575 – 602, 2003.

[6] M. Betts and R. Russell. Amino acid properties and consequences of subsitutions. In *Bioinformatics for Geneticists*. Wiley, 2003.

[7] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[8] B. Broome and M. Hecht. Nature disfavors sequences of alternating polar and non-polar amino acids: implications for amyloidogenesis. *J Mol Biol*, 296(4):961–968, 2000.

[9] J. Cornette, K. Cease, H. Margalit, J. Spouge, J. Berzofsky, and C. DeLisi. Hydrophobicity scales and computational techniques for detecting amphipathic structures in proteins. *J Mol Biol*, 195(3):659–685, 1987.

[10] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & sons, 1991.

[11] L. de la Ossa, K. Sastry, and F. G. Lobo. χ-ary extended compact genetic algorithm in c++. Technical Report 2006013, Illinois Genetic Algorithms Lab, University of Illinois at Urbana-Champaign, 2006.

[12] K. A. DeJong and W. M. Spears. Learning concept classification rules using genetic algorithms. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 651–656. Morgan Kaufmann, 1991.

[13] J. Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, February 1999.

[14] G. Harik. Linkage learning via probabilistic modeling in the ecga. Technical Report 99010, Illinois Genetic Algorithms Lab, University of Illinois at Urbana-Champaign, 1999.

[15] D. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *J Mol Biol*, 292:195–202, 1999.

[16] A. R. Kinjo, K. Horimoto, and K. Nishikawa. Predicting absolute contact numbers of native protein structure from amino acid sequence. *Proteins*, 58:158–165, 2005.

[17] N. Krasnogor, B. Blackburne, E. Burke, and J. Hirst. Multimeme algorithms for protein structure prediction. In *Proceedings of the Parallel Problem Solving from Nature VII. Lecture Notes in Computer Science*, volume 2439, pages 769–778, 2002.

[18] P. Larranaga and J. Lozano, editors. *Estimation of Distribution Algorithms, A New Tool for Evolutionnary Computation*. Genetic Algorithms and Evolutionnary Computation. Kluwer Academic Publishers, 2002.

[19] R. MacCallum. Striped sheets and protein contact prediction. *Bioinformatics*, 20:I224–I231, 2004.

[20] Y. Mandel-Gutfreund and L. Gregoret. On the significance of alternating patterns of polar and non-polar residues in beta-strands. *Journal of Molecular Biology*, 323(9):453–461, 2002.

[21] J. Meiler, M. M. A. Zeidler, and F. Schmähke. Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks. *J Mol Model*, 7:360–369, 2001.

[22] F. Melo and M. Marti-Renom. Accuracy of sequence alignment and fold assessment using reduced amino acid alphabets. *Proteins*, 63:986–995, 2006.

[23] J. Mintseris and Z. Weng. Optimizing protein representations with information theory. *Genome Informatics*, 15(1):160–169, 2004.

[24] K. M. Misura, D. Chivian, C. A. Rohl, D. E. Kim, and D. Baker. Physically realistic homology models built with rosetta can be more accurate than their templates. *Proc Natl Acad Sci U S A*, 103(14):5361–5366, 2006.

[25] T. Noguchi, H. Matsuda, and Y. Akiyama. Pdb-reprdb: a database of representative protein chains from the protein data bank (pdb). *Nucleic Acids Res*, 29:219–220, 2001.

[26] J. Rissanen. Modeling by shortest data description. *Automatica*, vol. 14:465–471, 1978.

[27] C. Sander and R. Schneider. Database of homology-derived protein structures. *Proteins*, 9:56–68, 1991.

[28] Y. Shao and C. Bystroff. Predicting interresidue contacts using templates and pathways. *Proteins*, 53:497–502, 2003.

[29] M. Stout, J. Bacardit, J. D. Hirst, N. Krasnogor, and J. Blazewicz. From hp lattice models to real proteins: Coordination number prediction using learning classifier systems. In *Applications of Evolutionary Computing, EvoWorkshops 2006*, pages 208–220. Springer LNCS 3907, 2006.

[30] G. Syswerda. Uniform crossover in genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms*, pages 2–9. Morgan Kaufmann Publishers Inc., 1989.

[31] G. Venturini. Sia: A supervised inductive algorithm with genetic search for learning attributes based concepts. In P. B. Brazdil, editor, *Machine Learning: ECML-93 - Proc. of the European Conference on Machine Learning*, pages 280–296. Springer-Verlag, Berlin, Heidelberg, 1993.

[32] K. Yue, K. M. Fiebig, P. D. Thomas, C. H. Sun, E. I. Shakhnovich, and K. A. Dill. A test of lattice protein folding algorithms. *Proc. Natl. Acad. Sci. USA*, 92:325–329, 1995.

[33] Y. Zhao and G. Karypis. Prediction of contact maps using support vector machines. In *Proceedings of the IEEE Symposium on BioInformatics and BioEngineering*, pages 26–36. IEEE Computer Society, 2003.