Hierarchical classification of G-Protein-Coupled Receptors with data-driven selection of attributes and classifiers

A. Secker

Computing Laboratory and Centre for BioMedical Informatics,
University of Kent,
Canterbury, CT2 7NF, UK
Fax: +44 (0)1227 76-2811
E-mail: andysecker@gmail.com

M.N. Davies

Edward Jenner Institute,
Compton, Newbury, Berkshire, RG20 7NNF, UK
Fax: +44 (0)207 631 6802
E-mail: m.davies@mail.cryst.bbk.ac.uk

A.A. Freitas*

Computing Laboratory and Centre for BioMedical Informatics,
University of Kent,
Canterbury, CT2 7NF, UK
Fax: +44 (0) 1227 76-2811
E-mail: a.a.freitas@kent.ac.uk
*Corresponding author

E. Clark and J. Timmis

Departments of Computer Science and Electronics,
University of York,
York, YO10 5DD, UK
Fax: +44 (0) 1904 432767
E-mail: edclark@cs.york.ac.uk
E-mail: jtimmis@cs.york.ac.uk

D.R. Flower

Edward Jenner Institute,
Compton, Newbury, Berkshire, RG20 7NNF, UK
Fax: +44 (0) 1904 432767
E-mail: darrenflower@googlemail.com

Abstract: We address the important bioinformatics problem of predicting protein function from a protein's primary sequence. We consider the functional classification of G-Protein-Coupled Receptors (GPCRs), whose functions are specified in a class hierarchy. We tackle this task using a novel top-down hierarchical classification system where, for each node in the class hierarchy, the predictor attributes to be used in that node and the classifier to be applied to the selected attributes are chosen in a data-driven manner. Compared with a previous hierarchical classification system selecting classifiers only, our new system significantly reduced processing time without significantly sacrificing predictive accuracy.

Keywords: hierarchical classification; supervised learning; attribute selection; feature selection; classifier selection; protein function prediction; GPCR; g-protein coupled receptor.

Reference to this paper should be made as follows: Secker, A., Davies, M.N., Freitas, A.A., Clark, E., Timmis, J. and Flower, D.R. (xxxx) 'Hierarchical classification of G-Protein-Coupled Receptors with data-driven selection of attributes and classifiers', *Int. J. Data Mining and Bioinformatics*, Vol. x, No. x, pp.xxx-xxx.

Biographical notes: Andrew Secker received a first class BSc with honours in Computer Science from the University of Kent, UK, in 2002 and was awarded a PhD from the same institution in 2006 for a thesis entitled Artificial Immune Systems for Web Content Mining: Focusing on the Discovery of Interesting Information. He interests lie in the use of novel machine learning for data mining, especially the application area of bioinformatics. He has published papers in machine learning and bioinformatics journals and was invited to speak at the UK Data Mining and Knowledge Discovery symposium in 2007.

Matthew Davies achieved his BSc in Biology from University College London in 1998; his MSc in Bioinformatics from University of Abertay Dundee in 1999 and his PhD in Crystallography from Birbeck College, London in 2004. He currently works as a postdoctoral researcher in the lab of Doctor Darren Flower at the Edward Jenner Institute. He has published over 20 research and review papers and is currently editing a book on Immunoinformatics for Springer. His research interests include bioinformatics, machine learning methods and structural modelling.

Alex A. Freitas obtained his BSc in Computer Science from FATEC-SP, Brazil, in 1989; his MSc in Computer Science from UFSCar, Brazil, in 1993; and his PhD in Computer Science from the University of Essex, UK, in 1997. He is currently a Reader (equivalent to Associate Professor) and the Head of Research at the Computing Laboratory, University of Kent, UK. He has authored two research-oriented books about data mining, and has published more than 100 peer-reviewed papers in journals and conferences. His current research interests are data mining, biologically-inspired computational intelligence and bioinformatics.

Edward B. Clark obtained his MPHYS in Theoretical Physics from the University of Kent, UK, in 2004. He is currently a Research Associate in the Departments of Biology and Computer Science at the University of York, on the 'Plazzmid' project EP/F031033/1. He has submitted his PhD thesis at the Department of Electronics at the University of York in November 2008. His current research interests are modelling optimisation algorithms with Markov chains, modelling receptor coverage in the human immune system and modelling bacterial evolution.

Jonathan Timmis is a Professor of Natural Computation at the University of York in a joint appointment with the Department of Computer Science and Department of Electronics. His primary research interest is in the computational abilities of the immune, neural and endocrine systems and how they relate to computer science and engineering. He has published over 80 papers on artificial immune system related research and is the co-author of the first text book on Artificial Immune Systems (AIS). He has worked on real-time error detection, immune inspired learning systems, robotics, theoretical aspects of AIS and complex systems modelling.

Darren R Flower is a Jenner investigator at the University of Oxford. He was formerly a Principal Research Scientist at the Astra Charnwood research laboratories, now part of AstraZeneca. He is the author of over 125 research papers and 30 invited book chapters; he has edited or written five books. He is Editor-in-chief of Immunology and Immunogenetics Insights, Editor-for-Europe for Current Computer-Aided Drug Design, and sits on nine other editorial boards. He is a Fellow of the Royal Society of Chemistry and Chair of its special interest group in Molecular Modelling. His interests range across all scientific disciplines that impact on the design of drugs and vaccines.

1 Introduction

Hierarchical classification is a variant of the well-known data-mining task of classification where classes are arranged in a hierarchy: typically either a tree or a DAG (direct acyclic graph) where each node corresponds to a class. Hierarchical classification is important in bioinformatics, particularly in protein function prediction, because such functions are often specified as a hierarchy. In this context, higher-level classes correspond to general functions, whilst lower-level classes correspond to more specific functions.

This paper focuses on predicting the functional class of (GPCRs), which are typically specified as a hierarchy. GPCRs are an important type of protein as they can transmit messages from a cell's exterior to its interior, modulating cellular behaviour. For this reason, GPCR proteins are a common target for therapeutic drugs: approximately 50% of all marketed drugs are targeted towards a GPCR (Flower, 1999; Klabunde and Hessler, 2002).

A popular strategy for hierarchical classification is the top-down approach. In the training phase, the system builds a tree of classifiers – where every classifier is associated with a node in the class tree. In the testing phase, a test example is classified in a top-down fashion, so that the example is first classified by the root node (0-th level) classifier. This prediction is used to decide to which child (1st level) classifier the example will be transferred. This decision process continues until the example reaches a leaf node. Typically, every node of the classifier tree uses the same kind of classification algorithm to build the classifier (classification model) at that node.

Davies et al. (2007b) and Secker et al. (2007) have recently proposed an alternative to this top-down hierarchical classification method which uses a set of candidate classification algorithms at each classifier node. The method obtaining the best classification accuracy is chosen as the classifier at that node. They called this the 'selective top-down approach', to emphasise that a classification algorithm is selected

at each class node. In general that approach outperforms conventional top-down classifiers that use only one form of classification algorithm throughout the hierarchy. However, the selective approach is often very slow due to the overhead of training and testing many different classifiers at each class node, particularly when the number of nodes is large.

This motivates the use of feature selection as a way to improve the computational efficiency of that approach. It is well-known that feature selection, when used in the data-preprocessing phase of the knowledge discovery process, often significantly reduces the time needed to train and/or test a classification algorithm. Feature selection may also increase classification accuracy on the test set as noisy, irrelevant or redundant attributes which confuse the classification algorithm can be removed. However, there is no guarantee that feature selection will improve predictive accuracy in practice.

We investigate here whether feature selection can be used to improve computational efficiency without compromising predictive accuracy, in the context of predicting the functional hierarchy of GPCR proteins.

We propose to increase the speed of the selective top-down approach for hierarchical classification using feature selection at each node of the class hierarchy. The features are selected independently at each node. Thus each classifier in the tree could use a different set of attributes for its predictions. Only attributes which are good predictors at a node will be used. This decision is made by the attribute selection method in a data driven manner.

In our experiments, feature selection maintained predictive accuracy yet reduced significantly the time necessary for training and testing of the top-down approach on a GPCR dataset.

The remainder of this paper is organised as follows. Section 2 reviews relevant bioinformatics concepts and methods, thus allowing better understanding of subsequent sections. Section 3 provides background on relevant data mining concepts and methods. Section 4 describes in detail the proposed top-down hierarchical classification system with both attribute selection and classifier selection. Section 5 discusses computational results evaluating the proposed system. Section 6 draws general conclusions and suggests future research directions.

2 Background on bionformatics

2.1 G-Protein-Coupled Receptors (GPCRs)

GPCRs are a type of protein. Proteins are large molecules comprising chains of amino acids. The order and identity of the amino acids in this chain together form the protein's primary sequence. Sequences fold into complex structures allowing them to perform functions. GPCRs are transmembrane proteins, meaning that they are embedded within the lipid bilayer of the cell's outer membrane so that some regions of the protein are exposed to the interior of the cell while other regions are exposed on the extracellular surface. GPCRs are bound by a variety of different molecules (ligands) found outside the cell. This binding activates the GPCR, which in turn binds a G protein inside the cell, causing an alteration in the cellular function.

The most widely used GPCR classification scheme, GPCRdb (Kolakowski, 1994), divides GPCR proteins into six families, designated A-F with class A being the largest human GPCR family. The GPCRdb classification scheme is a functional classification based on the ligand to which a receptor is bound rather than the primary sequence. The GPCRdb class structure is hierarchical. That is, general classes such as the families A-F subdivide naturally into subclasses. These subclasses divide again into sub-classes and so on producing a tree of classes, or a class hierarchy. Note that child classes have exactly one parent, unlike certain biological datasets where child classes may have numerous parent classes. In this study, GPCR families A – E are used. Class F was not considered as it contains too few sequences to induce a classification model. For similar reasons only the top 3 levels of the hierarchy are considered.

2.2 General approaches for GPCR classification

Previous methods for GPCR classification have included motif-based prediction (Flower and Attwood, 2004; Holden and Freitas, 2006) as well as machine learning techniques such as Hidden Markov Models (Moller et al., 2001) and Support Vector Machines (Karchin et al., 2002). Traditionally, protein sequence classification has been undertaken using alignment-based methods. It has been common to compare two proteins using amino acid substitution matrices, which measure relatedness (based on similarity) between two sequences. Over time, positions in a protein sequence mutate and traditional methods align one protein with another and measure the similarity between them.

Recently however, it is becoming more common to turn sequences into sets of numbers representing their properties. Called proteochemometrics, an advantage of this approach is that alignment is not needed, and standard classifiers drawn from the data mining literature may be used to categorise sequences. We use here an alignment-free representation to undertake sequence classification. Davies et al. (2007a, 2008) contains reviews of GPCR function prediction methods; extensive reviews of general protein function prediction (not just for GPCRs) are available (Friedberg, 2006; Rost et al., 2003).

2.3 Creation of attributes from protein sequence

Davies et al. (2007b) and Secker et al. (2007) have published previously using '*z*-values' to represent a protein sequence. 26 separate amino acid physicochemical properties were reduced to five composite '*z*' values, each '*z*' scale corresponding to the 20 natural amino acids. Each amino acid in the sequence is first transformed into its respective set of *z*-values. Averaging each of the five *z*-values over the sequence resulted in an accurate yet compact representation, where the entire sequence was reduced to five attributes. This was extended to 15 attributes in Davies et al. (2007b).

In this paper we explore instead the attribute creation technique defined in Tong and Tammi (2008), based on local descriptors, which led to better results in our experiments.

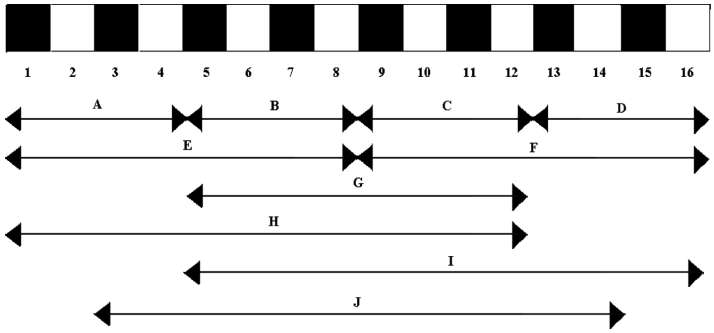
In developing these local-descriptors, Cui et al. (2007) divided the amino acids into three functional groups: hydrophobic (CVLIMFW), neutral (GASTPHY), and polar (RKEDQN), as suggested by Chothia and Finkelstein (1990). It is then possible to substitute the amino acids in the sequence for the group in which that amino acid belongs. Assuming H = hydrophobic, N = neutral and P = polar, the protein sequence CVGRK would be converted to HHNPP. The position or variation of these groups within

a sequence is the basis of three local descriptors: composition (*C*), transition (*T*), and distribution (*D*).

C is the proportion of amino acids with a particular property (drawn from a particular group such as hydrophobicity). As an example, given the group *H*, we can determine *C*(*H*) over the example sequence of HHNPP as 0.4 as 2 of 5 positions in the sequence are of value *H*. *T* is the frequency with which amino acids with one property are followed by amino acids with a different property. Thus to compute *T*(*N*) over the example sequence, we can see there is a transition between *H* and *N* from positions 2 to 3, then a transition from *N* to *P* between positions 3 and 4. In this case $T(N) = 2/4 = 0.5$ as there are four places where a transition may occur. Any transitions between *H* and *P* are ignored here as neither of these groups are the subject. Descriptor D measures the chain length within which the first, 25, 50, 75 and 100% occurrences of the particular property are located.

As the amino acids are divided into three groups, the calculation of the *C*, *T* and *D* descriptors generates 21 attributes in total (3 for *C*, 3 for *T* and 15 for *D*). While this technique is valid if applied over the whole amino acid sequence, Tong and Tammi (2008) split the amino acid sequences into 10 overlapping regions in order to better capture epitope binding patterns (see Figure 1). For sequences A–D and E–F there may be cases where the sequence cannot be divided exactly, necessitating extension of a subsequence by one residue. Each descriptor – C, T, and D – is calculated over the 10 subsequences, resulting in 210 features describing the protein. The number of attributes therefore generalises to 70*n*, where *n* is the number of amino acid groups.

Figure 1 The ten descriptor regions (A–J) for a theoretical protein sequence of 16 amino acids



Source: Tong and Tammi (2008)

Preliminary experiments showed that this data representation resulted in a good predictive accuracy when combined with the selective top-down classifier. A summary of our results compared with previously published results is shown in Table 1. The predictive accuracy given in Table 1 represents the accuracy computed at the leaf nodes of the class tree, which is the most specific (most informative to the user) prediction. Each result is generated using comparable protocols, using comparable data[1] and the same implementation of the top down hierarchical classification system with classifier selection. The exact experimental protocol is omitted; the interested reader is directed to the references cited in Table 1. The results shown in the last row have been produced using ten independent runs of a 10-fold cross validation.

Table 1 Comparison of the predictive accuracies (%) of different protein representations in top down hierarchical classification with classifier selection

Protein representation	*Accuracy* (%)
Means of *Z*-values Secker et al. (2007)	58.08
Means of *Z*-values with termini Davies et al. (2007b)	69.98
Local descriptors (used in this work)	70.46

Table 1 indicates that the 210 features generated by the local descriptors technique result in the best predictive accuracy out of the three types of protein representation (attribute sets). However, the burden of this many attributes results in unacceptably long runtimes for the selective top down classifier. As many of the 210 attributes may be redundant; running attribute selection prior to training and testing may be advantageous. This is explored fully in Section 4.

3 Background on data mining

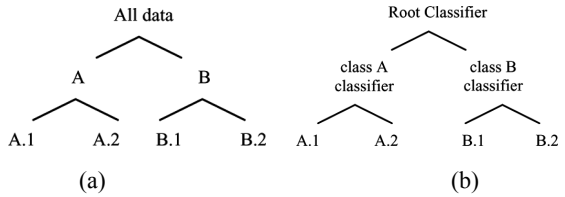
3.1 Top-down approach to hierarchical classification

The vast majority of the classifiers in the literature deal with flat data sets, with a single level of classes. Within a hierarchy, an example may be assigned to a class at a number of levels of specialisation. The most general level being nearest the root of the tree; levels become more specialised as the tree's branches are traversed.

The class structure of a typical flat dataset will contain, for example, classes A, B and C which are all equally distant from each other. However, in a hierarchy some classes may be more alike than others. Given the class tree in Figure 2(a), while classes A and B are equally dissimilar, classes, A1, A2, B1, and B2 are not. In this case A1 and A2 are more alike than A1 and B1, as A1 and A2 share a common super-class.

The top-down approach is an effective strategy for classifying data with a hierarchical class structure. In this strategy, the hierarchical classification problem is converted into a number of flat classification problems that are solved independently by running a flat classifier many times. A review on hierarchical classification techniques with particular emphasis on protein function prediction can be found in Freitas and Carvalho (2007).

Figure 2 Part (a) shows an example of a class hierarchy and Part (b) shows how that hierarchy may be reflected in a tree of classifiers ready for a top-down approach to classification



Given a class structure described by a tree, as in Figure 2(a), a tree of classifiers can be built reflecting this structure, as shown in Figure 2(b). Thus, a tree of classifiers is generated such that the output of one classifier constitutes the input for another. The number of layers of classifiers will equal the number of levels represented by the

class attribute, and one classifier is built for each internal (non-leaf) node in the class tree. To train the classifiers, all data in the training set is used to train the root classifier. However this is not the case with subsequent classifiers. For example, the 'class A classifier' needs only classify an instance as A.1 or A.2, so only data of class A is used for training. Likewise the 'class B classifier' is trained using only data belonging to class B.

When classifying new instances in the test set (unseen during training), the root level classifier will assign a class to the top level only – in the case of Figure 2, class A or B. The instance will then be passed to the next level based on the decision made by that root classifier; i.e., if the instance is assigned class A then it will be passed to the 'class A classifier'. This process continues until an instance is assigned its final leaf class.

3.2 Top-down hierarchical classification with classifier selection

In a naive approach, hierarchical data can be classified by simply flattening the class hierarchy. Here, a single classifier assigns every test instance the class of a leaf node, and parent nodes can then be inferred from this most specific class. However, many important real-world hierarchical classification problems have many tens or hundreds of classes at the leaf level. Given this superfluity of classes, the accuracy of the single flat classifier can be impeded.

The top-down approach tends to be more effective than the above approach, because it decomposes the problem into successively smaller sub-problems, reducing the number of classes a classifier must discern, and thus tends to improve predictive accuracy. The top-down approach takes advantage of the hypothesis that a particular characteristic of the data used to discern between certain classes may differ between branches and levels of the classifier tree. However, one may also hypothesise that different classifiers may be more suited to different tree nodes as each type of classifier has its own inductive bias.

The general top-down approach, as described above, may therefore be improved by using different classification algorithms at different nodes in the classifier tree, which is the basic idea of the selective top-down approach proposed in Davies et al. (2007b) and Secker et al. (2007). In this approach, as each classifier is selected in a data-driven manner from several candidates, each node should contain the classifier best suited to the available data. At each node in the classifier tree, the training data for that node is split into a sub-training and validation set – with data being assigned randomly to each data subset. A number of different classifiers are then trained using this sub-training data and tested using the validation set. The classifier yielding the highest classification accuracy on the validation set is selected. The sub-training and validation sets are then merged to produce the original training set again, and the selected classifier is then re-trained.

In the current paper, the just-described "selective top-down approach" is re-named to the "top-down approach with classifier selection", as this makes it clear what is being selected and avoids confusion with the major extension proposed here. More precisely, in this paper we propose a top-down approach with both attribute selection and classifier selection, as discussed in Section 4.

3.3 Attribute selection

Attribute or feature selection (Guyon and Elisseeff, 2003; Liu and Motoda, 2008) is a well-known technique in data mining. It functions by removing attributes from the dataset leaving only those that correlate highly with the class attribute, thus simplifying the classification problem. This often leads to a decreased run time, and can also lead to an increased predictive accuracy, since noisy, irrelevant or otherwise redundant attributes, which may confuse the classification algorithm, will tend to be removed.

Two broad types of attribute selection methods (applied in a data preprocessing stage) exist: wrapper and filter methods.

Filter methods undertake attribute selection without using the classification algorithm that will be applied to the selected attributes. Filter methods typically examine each attribute independently of all others, determining the level of correlation between the attribute and the predicted class. Attributes with low correlation are removed. Filter methods are usually fast, but are limited in that they do not take account of any interaction between attributes. It is possible that several attributes will combine to allow more accurate prediction than when assessed individually.

Wrapper methods use the classification algorithm that will be applied to the selected attributes. Unlike a filter method, a wrapper method selects attributes tailored to the particular classification algorithm. However, a wrapper method is much slower than a filter method and it is not practical in the context of this research, since a major motivation for this work is to reduce the time taken to train classification algorithms in our large dataset. Therefore, in this work we use filter methods only.

3.4 Related work on attribute selection in hierarchical classification

To the best of our knowledge, there are only two examples of attribute selection being used in conjunction with a top-down classifier; both are concerned with classification of text documents, rather than bioinformatics. Koller and Sahami (1999) implemented a top-down classifier that used an information-theoretic measure to choose attributes that "best capture the class distribution in the data" at each node. This attribute selection mechanism was run once at each node in a similar manner to our method. However, there are major differences between that work and this. Koller and Sahami used the same classifier at every node, since their method does not select classification algorithms. Moreover, the number of attributes to be selected at each node was a user-defined parameter. In Section 5.1 we show that this is seldom an ideal strategy. Furthermore, the two datasets used for testing their technique were somewhat simple compared to ours. Both hierarchies contained just 2 levels. Their first dataset contained three classes at the top (most general) level, each of which subdivided into two classes. Their second dataset has just two classes at the top level, each of which again divided into two classes. By contrast, our dataset contains 5, 38 and 87 classes at the first, second and third class levels, respectively, making the hierarchical classification problem we tackle considerably more difficult and complex.

Mladenic and Grobelnic (1998) classified documents into a hierarchy using a top-down approach that utilised a Naïve Bayes classifier at each node. They used an attribute quality measure called "weight of evidence for text". It seemed this attribute selection strategy was applied once, at the beginning of the training. The class hierarchy was thus effectively flattened.

4 The proposed top-down hierarchical classification system with both attribute selection and classifier selection

4.1 Choosing an attribute selection method

The WEKA data mining toolkit (Witten and Frank, 2005) was used to undertake a preliminary study of attribute selection. WEKA provides two kinds of attribute selection methods: methods that search the solution space and automatically decide both the best combination of attributes and the number of attributes, and methods that score each attribute, ranking them in order of importance, which requires the user to specify the number of attributes.

We used our own GDS dataset, which comprised 8,222 GPCR sequences (see Section 5), with each protein being represented by 210 attributes, as defined in Section 2.3. The 1-nearest neighbour classifier was used. In these preliminary experiments classification took place at the top level (GPCR family level) only. The results of these tests – comparing nine different attribute selection methods – are shown in Table 2; they were produced from one run of a 10-fold cross validation over the entire dataset.

Table 2 Preliminary experiments with WEKA to determine the best attribute selection method

Attribute selection method	*Predictive accuracy* (%)	*Mean number of selected attributes*
CfsSubsetEval	96.2	20.6
ClassifierSubsetEval	13.6	1
ConsistencySubsetEval	92.9	7
ChiSquared	94.0	20
GainRatioAttributeEval	96.0	20
PrincipalComponents	96.5	20
OneRAttributeEval	95.6	20
SymmetricalUncertAttributeEval	95.3	20
InfoGainAttributeEval	93.3	20

The first three attribute selection methods automatically select the number of attributes using a best-first search method. The remaining methods rank the attributes, and the top 20 were selected, where 20 was a user-defined threshold. The need to decide the number of attributes eliminates most of these attribute selection methods for use in a top-down hierarchical classification system. It is possible that different nodes will require different numbers of attributes for optimal performance. Specifying an arbitrary number of attributes will prevent the algorithm from selecting the best combinations of attributes in a data driven manner. Of the remaining three, neither 'ClassifierSubsetEval' or 'ConsistencySubsetEval' could compete with 'CfsSubsetEval' in terms of accuracy. Thus, 'CfsSubsetEval' was chosen as the single attribute selection method to be applied at each node. The CfsSubsetEval method is described in Hall (1998). In essence, this method evaluates each attribute individually, assigning a higher score to attributes which are more correlated with the class and less correlated with other attributes.

4.2 The proposed top-down hierarchical classification system

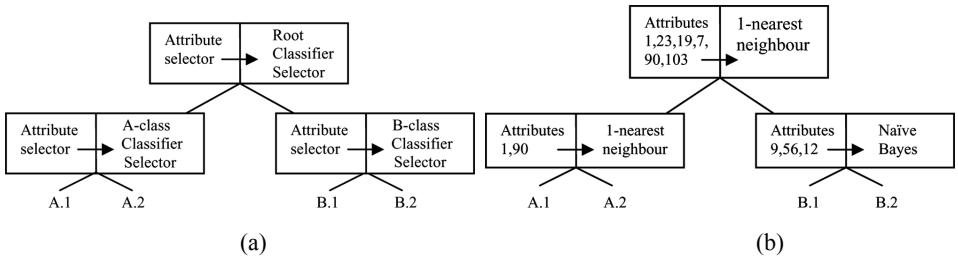
Naively, an attribute selection method could be run over the training data before beginning the training of the hierarchical classification system, in order to select a single, global set of attributes, and only those attributes would be passed to the top-down classification system with classifier selection. This is not in the spirit of the original top-down approach with classifier selection, where classification accuracy is maximised at each node by selecting the classifier with the best predictive accuracy. The proposed top-down hierarchical classification system with both attribute selection and classifier selection is therefore more complex, as follows.

Building a tree of classifiers proceeds using the same basic principles as described in Davies et al. (2007b) and Secker et al. (2007). A tree of classifiers is built up from the most generic root classifier as before. However, at every node where a classification must occur (i.e., more than one subclass exists at that point in the tree) the attribute selection method is used to reduce the dimensionality of the data. The attribute selection method (CfsSubsetEval) will be shown the training data for that node and then selects attributes based only on that data. Thus, at each classification node, the set of attributes best describing the class distribution at that point is first selected, and a classification algorithm which best uses the newly reduced data is selected. Different attributes are likely selected at each node. Hence, while a given subset of attributes may excel at predicting one subset of classes, it may perform poorly when predicting another.

All attributes are available at each node in the tree, since attributes are not 'lost' to child classifier nodes when not selected by CfsSubsetEval at a parent node. In cases where a class has only one subclass, no attribute selection occurs. A 'dummy' classifier is used to output the name of the subclass, further accelerating the algorithm.

Figure 3 shows the classifier tree before and after training. Each classifier takes the output of the attribute selection method as input. After training, the most appropriate set of attributes is recorded for each node. It may be noted that a single attribute selection method is used throughout the classifier tree. While several attribute selection methods are available, to attempt to optimise attribute selection and classifier at each classifier node would result in an unacceptable combinatorial explosion in the number of possibilities to be tested.

Figure 3 The top down classifier with attribute selection and classifier selection: (a) ready for training and (b) after training



5 Computational results

We compared our novel top-down approach using both attribute selection and classifier selection with the previous top-down approach proposed by Davies et al. (2007b) and Secker et al. (2007) using classifier selection only. Each top-down classifier used the same set of candidate classifiers at each node. These were:

- Naïve bayes
- Bayesian Network
- SMO (a support vector machine (Keerthi et al., 2001))
- 1 nearest neighbour (using Euclidean distance)
- PART (a rule induction algorithm (Frank and Witten, 1998))
- J48 (an implementation of C4.5)
- Naïve Bayes Tree (a decision tree with a naïve bayes classifier at each node)
- AIRS2 (a classifier based on the Artificial Immune System paradigm (Watkins and Timmis, 2002))
- Conjunctive rule learner.

Note that neural networks – used previously in the experiments reported by Davies et al. (2007b) and Secker et al. (2007) – were not used here, since in our experiments this algorithm took an unusually long time to train at each classifier node, yet constantly returned poor classification accuracies. As the accuracy was consistently poor, it was never selected as a classifier at any node. This phenomenon was observed both in our experiments and in the experiments reported by Davies et al. (2007b) and Secker et al. (2007). Therefore, the results reported here remain comparable with those previously published results.

The dataset is as described in Davies et al. (2007b). It comprises 8,222 GPCR protein sequences identified using the Entrez search and retrieval system (Wheeler et al., 2007), which searches protein databases such as SwissProt, PIR, PRF, PDB as well as translations from annotated coding regions in DNA databases such as GenBank and RefSeq. All sequences are longer than 280 residues in length and only classes with 10 or more examples at the 3rd level (most specific) in the hierarchy were retained. This left 87 classes at the most specific level, 38 at the 2nd level and five classes at most general level (GPCR families A – E). The raw protein sequences in this dataset were converted into 210 numerical attributes using the method described in Section 2.3.

The top-down approach with classifier selection only and the new top-down approach with both attribute selection and classifier selection were both run ten times, each run being a separate 10-fold cross validation of the data. The folds were constructed to ensure that at least one example of every 3rd level class is present in every fold.

The results for predictive accuracy for the two approaches are shown in Table 3. A statistical significance test of the difference between the two accuracies was computed using a corrected resampled *t*-test (2-tailed) as detailed in Witten and Frank (2005). This test avoids problems encountered when a standard *t*-test is used over multiple runs of a cross-validation procedure. The obtained *p* value was 0.3859, indicating that the

difference between the two methods is not statistically significant. Hence, the top-down approach with both attribute selection and classifier selection greatly reduces processing time – see Section 5.2 – without significantly reducing predictive accuracy.

Table 3 Comparison of predictive accuracies between two top down approaches

Hierarchical classification approach	*Accuracy* (%)
Top down with classifier selection only	70.46
Top down with both attribute selection and classifier selection	69.97

5.1 Investigating selected attributes

The number of attributes selected by the attribute selection method at different nodes of the hierarchy were analysed. Recall that the raw dataset contains 210 attributes. Only parent class nodes with more than one subclass (i.e., where a classification actually needs to take place) are shown. The figures in Table 4 show the mean number of attributes selected at each class node, where the numbers between brackets are the standard deviations of the figures.

Table 4 Comparison of the mean number of attributes selected at each node of the classifier tree

Level	*Parent class*	*Mean number of attributes*
Level 1	Root	19.70 (0.41)
Level 2	ClassA	31.46 (1.31)
	ClassC	26.88 (0.36)
	ClassB	85.14 (0.98)
Level 3	ClassA_Amine	28.04 (0.32)
	ClassA_Hormone	23.20 (0.13)
	ClassA_Nucleotide	16.82 (0.64)
	ClassA_Peptide	44.20 (0.29)
	ClassA_Prostanoid	10.04 (0.91)
	ClassA_Thyro	13.22 (0.60)
	ClassC_CalcSense	32.94 (0.46)

It is clear from Table 4 that the number of attributes selected at different nodes varies considerably. This corroborates the use of an attribute selection method that automatically determines the number of selected attributes. Results indicate that while a threshold of 20 would have been appropriate for the root class node, 20 attributes would probably not capture the intricacies of the predictor attribute/class attribute relationship for the node distinguishing between subclasses of ClassB, which required approximately 85 attributes. Conversely, for the ClassA_Prostanoid node, 20 attributes may have been wasteful.

For the sake of completeness, the actual attributes selected have been recorded and are shown in the Appendix. These results have been generated using a single run of the

algorithm. In this case, the entire dataset was used as the training set, so no testing set is required.

5.2 Investigating selected classifiers

We also analysed the frequency with which each classifier was selected at each node in the class tree, and compared those frequencies with the number of examples (data instances) available at each node – in order to determine whether there was a clear pattern indicating that a certain type of classifier was much more often selected at class nodes having much smaller or much larger numbers of examples.

Table 5 shows, for each node in the class tree, the frequency with which each type of classifier was chosen from the menu. As a 10-fold cross validation was run ten times, the figures along each row sum to 100. Some of the classifier names have been abbreviated as follows: NB – Naïve Bayes, BN – Bayesian Network, NBT – Naïve Bayes Tree, CRL – Conjunctive Rule Learner. This table also contains a label headed '#Ex' which shows the number of examples at the corresponding node. As 10-fold cross validation is used, the classifier at that node will be trained with 90% of those examples during each iteration (fold) of the cross-validation procedure. Note that only 11 class nodes are included in the table as these are the nodes that have more than one sub-class. The remainder of the nodes in the tree have a single subclass and a classifier is, therefore, unnecessary at that node. In the table, a cell is left blank when its associated frequency count is 0, in order to simply the reading of the table.

As shown in Table 5, of particular interest is the observation that the instance-based learner IBK1 is always selected at the root node and selected in 90% of the cases at the Class A node, respectively. These are the two class nodes with by far the largest number of examples, respectively 8,222 and 5,401 examples. In addition, IBK1 was never selected at the two class nodes with the smallest number of examples, namely Class A – Thyro and Class A – Prostanoid, with just 69 and 39 examples, respectively. This clearly suggests that IBK1 performs particularly well, by comparison with the other classifiers, in larger datasets.

Table 5 Frequency of selection of each classifier at each node of the classifier tree

Class node	*#Ex*	*NB*	*BN*	*SMO*	*IBK1*	*PART*	*J48*	*NBT*	*Airs2*	*CRL*
Root	8222				100					
Class A	5401				90	10				
Class B	618	4	15	43	38					
Class C	2172	3		1		64	30		2	
Class A, Amine	1489		1		34	44	13	8		
Class A, Hormone	159	34	36	12	9	3		1	5	
Class A, Nucleotide	266	45	11	9	19	8		1		7
Class A, Peptide	2703		30	2	68					
Class A, Thyro	69	75	21	3					1	
ClassA, prostanoid	39	94	6							
Class C, CalcSense	588	12	29	10	25	7	2	13	2	

In contrast, Naïve Bayes performs particularly well, by comparison with the other classifiers, in class nodes with very small number of examples. This classifier was selected in 94% and 75% of the cases in the aforementioned class nodes Class A – Thyro and Class A – Prostanoid, respectively, which have the smallest number of examples among all class nodes. In addition, Naïve Bayes was never selected in the two class nodes with the largest number of examples, the root node and the Class A node.

In the case of the 'medium-sized' class nodes, there are less strong associations between classifiers and numbers of examples, but it is worth noting that PART was selected in 64% and 44% of the cases in two medium-sized or moderately large class nodes, namely Class C and Class A – Amine, respectively, with 2,172 and 1,489 examples.

5.3 Processing time

As explained earlier, a key motivation for this study was to reduce execution times. Tests took place on a desktop PC with an Intel Pentium IV processor running at 2.4 Ghz and 512 MB of RAM. The baseline approach – the top-down approach with classifier selection only – took on average 307.6 h (standard deviation 0.31) to complete a 10-fold cross validation on the target dataset, with 210 attributes. This can be compared to the enhanced top-down approach with both attribute selection and classifier selection, which took, on average, 113.5 hrs (standard deviation 0.48) – including the time for both attribute selection and classifier selection – on the same dataset. This represents a net speed increase of approximately 2.7 times compared with the baseline. Even though the algorithm has the overhead of selecting attributes at each node, the increase in speed is large on this dataset, while no statistically significant predictive accuracy is lost.

While this appears to be a long time for each run, the majority of the time is taken with training the proposed top-down approach. Once training is complete, test examples require only three sequentially executed classifications (one for each level of the tree). The classification of unknown-class test examples is, therefore, extremely fast.

6 Conclusions

In this paper, the top-down approach with classifier selection proposed by Davies et al. (2007b) and Secker et al. (2007) was augmented with attribute selection at each node in the classifier tree. Attribute selection occurs independently at each node in a data-driven manner, leaving only those attributes that best discriminate classes at that node. The attribute selection method chosen automatically varies the number of attributes selected, again in a data-driven manner.

The proposed method of selecting the best attribute subset in a local manner at each node in the classifier tree avoids two drawbacks that are associated with the approach of performing attribute selection just once for the entire dataset in a preprocessing step, as follows. First, the proposed approach will not use, in a given current node of the classifier tree, attributes that have good predictive power in other nodes of the tree but have little or no predictive power at the current classifier node. In particular, if the current node is a relatively deep node, it is quite possible that attributes that have a good predictive power at ancestors of that node will not have a good predictive power at the current node, since different classifier nodes have to solve different classification

problems. The proposed method allows this situation to be easily detected and dealt with by selecting just the small subset of relevant attributes for the current node. Secondly, the proposed approach allows the identification and use of attributes that have a good predictive power at a given deep node in the classifier tree, even though those attributes might have very little predictive power when evaluated on the whole dataset.

In our experiments the number of attributes selected was highly variable between classifier nodes, but varied little for the same node over multiple runs of a cross-validation procedure. Thus, it can be seen that the attribute selection method is reacting to the varying levels of difficulty of predicting particular classes at different positions in the class tree. It was found that the addition of this attribute selection stage made no significant difference to the predictive accuracy of the hierarchical classification system, yet the processing time was reduced significantly. Such processing time reduction will tend to be even more important in other larger bioinformatics datasets where hierarchical classification can be applied; so the proposed technique helps to address the issue of scalability to larger datasets, an important issue in data mining.

A future direction for the current system would involve using different attribute selection methods at different nodes in the tree. This raises the issue of the numerous combinations of attribute selection methods and classifiers. Intuitively, the output of each different attribute selection method would be somewhat different and different classifiers would react differently to each different set of selected attributes, which suggests that, in order to maximise predictive accuracy, the system would have to consider, during training, every possible pair of attribute selection method and classifier. However, with numerous attribute selection methods to choose from, the time required for training would increase many fold. Therefore any potentially marginal gain in predictive accuracy would need to be offset by a massively increased training time.

Acknowledgement

The authors should like to gratefully acknowledge funding under the EPSRC grant EP/D501377/1.

References

Chothia, C. and Finkelstein, A.V. (1990) ‘The classification and origins of protein folding patterns’, *Annual Review of Biochemistry*, Vol. 59, pp.1007–1035.

Cui, J., Han, L.Y., Li, H., Ung, C.Y., Tang, Z.Q., Zheng, C.J., Cao, Z.W. and Chen, Y.Z. (2007) ‘Computer prediction of allergen proteins from sequence-derived protein structural and physicochemical properties’, *Mollecular Immunology*, Vol. 44, pp.514–520.

Davies, M.N., Gloriam, D.E., Secker, A., Freitas, A.A., Mendao, M., Timmis, J. and Flower, D.R. (2007a) ‘Proteomic applications of automated GPCR classification’, *Proteomics*, Vol. 7, No. 16, pp.2800–2814.

Davies, M.N., Secker, A., Freitas, A.A., Mendao, M., Timmis, J. and Flower, D.R. (2007b) ‘On the hierarchical classification of G-Protein-Coupled Receptors’, *Bioinformatics*, Vol. 23, No. 23, pp.3113–3118.

Davies, M.N., Secker, A., Freitas, A.A., Timmis, J. and Flower, D.R. (2008) ‘Alignment-independent techniques for protein classification’, *Current Proteomics*, Vol. 5, No. 4, pp.217–223.

Flower, D.R. (1999) 'Modelling G-protein-coupled receptors for drug design', *Biochim Biophys Acta*, Vol. 1422, pp.207–234.

Flower, D.R. and Attwood, T.K. (2004) 'Integrative bioinformatics for functional genome annotation: trawling for G-Protein-Coupled Receptors', *Seminars in Cell and Developmental Biology*, Vol. 15, pp.693–701.

Frank, E. and Witten, I.H. (1998) 'Generating accurate rule sets without global optimization', *Proc. 15th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, pp.144–151.

Freitas, A.A. and de Carvalho, A.C.P.L.F. (2007) 'A tutorial on hierarchical classification with applications in bioinformatics', in Taniar, D. (Ed.): *Research and Trends in Data Mining Technologies and Applications*, Idea Group, London, pp.175–208.

Friedberg, I. (2006) 'Automated protein function prediction – the genomic challenge', *Briefings in Bioinformatics*, Vol. 7, No. 3, pp.225–242.

Guyon, I. and Elisseeff, A. (2003) 'An introduction to variable and feature selection', *The Journal of Machine Learning Research*, Vol. 3, March, pp.1157–1182.

Hall, M.A. (1998) *Correlation-based Feature Subset Selection for Machine Learning*, PhD Thesis, University of Waikato, New Zealand.

Holden, N. and Freitas, A.A. (2006) 'Hierarchical classification of G-Protein Coupled Receptors With a PSO/ACO Algorithm', *Proc. IEEE Swarm Intelligence Symposium (SIS-06)*, IEEE Press, Piscataway, NJ, pp.77–84.

Karchin, R., Karplus, K. and Haussler, D. (2002) 'Classifying g-protein coupled receptors with support vector machines', *Bioinformatics*, Vol. 18, pp.147–159.

Keerthi, S.S., Shevade, S.K., Bhattacharyya, C. and Murthy, K.R.K. (2001) 'Improvements to Platt's SMO Algorithm for SVM Classifier Design', *Neural Computation*, Vol. 13, No. 3, pp.637–649.

Klabunde, T. and Hessler, G. (2002) 'Drug design strategies for targeting G-protein coupled receptors', *ChemBioChem*, Vol. 3, pp.928–944.

Kolakowski Jr., L.F. (1994) 'GCRDb: a G-Protein-Coupled Receptor Database', *Receptors Channels*, Vol. 2, pp.1–7.

Koller, D. and Sahami, M. (1999) 'Hierarchically classifying documents using very few words', *Proc. of the 14th International Conference on Machine Learning (ICML-1999)*, Morgan Kaufmann, San Francisco, CA, pp.170–178.

Liu, H. and Motoda, H. (2008) *Computational Methods of Feature Selection*, Chapman & Hall/CRC, London.

Mladenic, D. and Grobelnic, M. (1998) 'Feature selection for classification based on text hierarchy', *Working Notes of Learning From Text and the Web, Conference on Automated Learning and Discovery (CONALD-98)*, Pittsburgh, PA.

Moller, S., Vilo, J. and Croning, M.D. (2001) 'Prediction of the coupling specificity of G-protein coupled receptors to their g-proteins', *Bioinformatics*, Vol. 17, Suppl. 1, pp.S174–S181.

Rost, B., Liu, J., Nair, R., Wrzeszczynski, K.O. and Ofran, Y. (2003) 'Automatic prediction of protein function', *CMLS Cellular and Molecular Life Sciences*, Vol. 60, pp.2637–2650.

Secker, A., Davies, M.N., Freitas, A.A., Timmis, J., Mendao, M. and Flower, D.R. (2007) 'An experimental comparison of classification algorithms for the hierarchical prediction of protein function', *Expert Update (Magazine of the British Computer Society's SGAI), special issue on the 3rd UK Data mining and Knowledge Discovery Symposium (UKKDD 2007)*, Vol. 9, No. 3, pp.17–22.

Tong, J.C. and Tammi, M.T. (2008) 'Prediction of protein allergenicity using local descriptions of amino acid sequence', *Frontiers in Bioscience*, Vol. 13, 1 May, pp.6072–6078.

Watkins, A. and Timmis, J. (2002) 'Artificial Immune Recognition System (AIRS): revisions and refinements', *Proc. 1st International Conference on Artificial Immune Systems (ICARIS 2002)*, Canterbury, UK, pp.173–181.

Wheeler, D.L., Barrett, T., Benson, D.A., Bryant, S.H., Canese, K., Chetvernin, V., Church, D.M., DiCuccio, M., Edgar, R., Federhen, S., Geer, L.Y., Kapustin, Y., Khovayko, O., Landsman, D., Lipman, D.J., Madden, T.L., Maglott, D.R., Ostell, J., Miller, V., Pruitt, K.D., Schuler, G.D., Sequeira, E., Sherry, S.T., Sirotkin, K., Souvorov, A., Starchenko, G., Tatusov, R.L., Tatusova, T.A., Wagner, L. and Yaschenko, E. (2007) 'Database resources of the national center for biotechnology information', *Nucleic Acids Research*, Vol. 36, database issue, pp.D13–D21.

Witten, I.H. and Frank, E. (2005) *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed., Morgan Kaufmann, San Francisco, CA.

Appendix

The attributes selected for the GPCR dataset in the experiments reported in Section 5 are displayed below. For each parent class node (i.e., each class for which a classifier was built to discriminate among its child classes), we list all attributes selected by CfsSubsetEval at that parent class node.

For attributes created from the composition or transition metrics, the selected attributes are shown in the form <*X*, *Y*, *z*>, where *X* can be COMP or TRANS, showing the attribute has been created using the composition or transition metric respectively; *Y* represents the position of the subsequence as denoted in Figure 1; and *z* is the amino acid group that was the subject of this attribute. This can either be p (polar), h (hydrophobic) or n (neutral).

Attributes created by the distribution metric are shown in the form <*X*, *Y*, *z*, *n*>, where *X*, *Y* and *z* are as before and *n* shows the type of distribution used to create this attribute. This can take the values "first", for the first occurrence of the group represented by z, or 25, 50, 75 or 100%.

Root node (21 selected attributes)

COMP.A.p, COMP.B.p, COMP.C.p, COMP.D.n, COMP.E.p, TRANS.B.h, TRANS.E.n, TRANS.E.h, DIST.A.p.first, DIST.A.p.25%, DIST.A.p.50%, DIST.A.n.25%, DIST.A.h.50%, DIST.E.p.25%, DIST.F.p.50%, DIST.F.p.75%, DIST.F.n.25%, DIST.H.p.first, DIST.I.n.50%, DIST.J.p.25%, DIST.J.n.50%

Class A (25 selected attributes)

COMP.A.n, COMP.C.p, COMP.C.n, COMP.G.p, COMP.G.n, COMP.H.p, COMP.J.p, TRANS.E.n, TRANS.F.n, TRANS.J.p, TRANS.J.n, DIST.D.p.25%, DIST.D.p.50%, DIST.F.h.75%, DIST.G.p.25%, DIST.G.p.50%, DIST.G.p.75%, DIST.H.p.25%, DIST.H.p.50%, DIST.H.p.75%, DIST.H.p.100%, DIST.H.n.25%, DIST.H.n.50%, DIST.J.p.50%, DIST.J.p.100%

Class B (84 selected attributes)

COMP.A.p, COMP.A.h, COMP.B.n, COMP.B.h, COMP.C.p, COMP.D.p, COMP.F.n, COMP.G.h, COMP.I.p, COMP.I.h, COMP.J.p, COMP.J.n, COMP.J.h, TRANS.A.p, TRANS.A.n, TRANS.A.h, TRANS.B.p, TRANS.B.n, TRANS.B.h, TRANS.C.p, TRANS.C.n, TRANS.C.h, TRANS.D.p, TRANS.D.h, TRANS.E.n, TRANS.E.h, TRANS.F.n, TRANS.G.p, TRANS.I.n, TRANS.I.h, TRANS.J.p, TRANS.J.n, DIST.A.p.25%, DIST.A.p.50%, DIST.A.n.25%, DIST.A.n.50%, DIST.A.n.75%, DIST.A.h.25%, DIST.B.p.50%, DIST.B.n.25%, DIST.B.n.75%, DIST.B.h.50%, DIST.C.p.75%, DIST.C.h.25%, DIST.D.p.50%, DIST.D.p.75%, DIST.D.n.25%, DIST.D.n.50%, DIST.D.n.75%, DIST.D.h.25%, DIST.D.h.75%, DIST.E.p.first, DIST.E.p.50%, DIST.E.n.50%, DIST.E.n.75%, DIST.F.p.75%, DIST.F.p.100%, DIST.F.n.25%, DIST.F.n.50%, DIST.F.n.75%, DIST.F.h.50%, DIST.F.h.75%, DIST.G.n.50%, DIST.G.n.75%, DIST.G.h.50%, DIST.G.h.75%, DIST.G.h.100%, DIST.H.p.75%, DIST.H.n.50%, DIST.H.h.25%, DIST.H.h.75%, DIST.I.p.25%, DIST.I.p.75%, DIST.I.n.75%, DIST.I.h.50%, DIST.I.h.75%, DIST.I.h.100%, DIST.J.p.25%, DIST.J.p.100%, DIST.J.n.25%, DIST.J.n.50%, DIST.J.h.25%, DIST.J.h.50%, DIST.J.h.75%

Class C (24 selected attributes)

COMP.A.p, COMP.A.n, COMP.B.p, COMP.D.p, COMP.D.n, COMP.E.p, COMP.H.p, COMP.H.h, COMP.I.p, COMP.I.n, COMP.J.p, TRANS.A.n, TRANS.B.n, TRANS.G.n, TRANS.H.n, TRANS.I.p, DIST.B.n.75%, DIST.C.h.75%, DIST.E.p.first, DIST.F.p.75%, DIST.G.p.50%, DIST.G.n.50%, DIST.G.n.75%, DIST.I.p.100%

Class A_Amine (28 selected attributes)

COMP.A.n, COMP.B.n, COMP.C.p, COMP.D.n, COMP.F.p, COMP.F.n, COMP.G.p, COMP.I.p, COMP.I.n, COMP.J.p, TRANS.J.n, DIST.A.n.first, DIST.B.n.50%, DIST.D.n.50%, DIST.E.p.50%, DIST.E.n.25%, DIST.E.n.50%, DIST.F.p.25%, DIST.G.p.first, DIST.G.n.75%, DIST.H.n.25%, DIST.H.n.75%, DIST.H.h.75%, DIST.H.h.100%, DIST.I.p.100%, DIST.I.n.first, DIST.I.n.100%, DIST.J.p.25%

Class A_Hormone (22 selected attributes)

COMP.B.h, COMP.E.p, COMP.G.n, COMP.H.n, TRANS.A.n, TRANS.A.h, TRANS.D.n, TRANS.E.h, DIST.A.p.first, DIST.A.p.50%, DIST.D.p.100%, DIST.E.p.first, DIST.G.p.50%, DIST.G.p.75%, DIST.G.h.25%, DIST.H.n.75%, DIST.H.h.50%, DIST.I.p.first, DIST.I.n.50%, DIST.I.n.100%, DIST.J.h.75%, DIST.J.h.100%

Class A_Nucleotide (18 selected attributes)

COMP.A.p, COMP.A.n, COMP.F.p, TRANS.A.p, TRANS.B.h, TRANS.I.p, DIST.A.n.75%, DIST.A.h.75%, DIST.B.p.25%, DIST.B.n.25%, DIST.C.p.75%, DIST.F.p.25%, DIST.G.p.first, DIST.G.p.75%, DIST.H.n.50%, DIST.H.n.75%, DIST.I.p.25%, DIST.I.n.50%

Class A_Peptide (39 selected attributes)

COMP.A.h, COMP.D.p, COMP.D.n, COMP.F.p, COMP.F.n, COMP.F.h, COMP.G.n, COMP.H.n, COMP.I.p, COMP.I.n, COMP.I.h, COMP.J.p, COMP.J.n, COMP.J.h, TRANS.C.n, TRANS.G.n, TRANS.J.n, DIST.A.p.25%, DIST.C.p.75%, DIST.C.n.75%, DIST.D.p.100%, DIST.D.n.first, DIST.D.n.100%, DIST.D.h.75%, DIST.D.h.100%, DIST.E.n.first, DIST.E.n.50%, DIST.F.p.50%, DIST.H.p.75%, DIST.H.n.75%, DIST.I.p.first, DIST.I.p.75%, DIST.I.p.100%, DIST.I.n.25%, DIST.I.n.100%, DIST.J.p.25%, DIST.J.n.50%, DIST.J.h.50%, DIST.J.h.100%

Class A_Prostanoid (9 selected attributes)

TRANS.E.n, DIST.D.p.50%, DIST.D.h.25%, DIST.F.h.75%, DIST.H.p.75%, DIST.I.h.75%, DIST.J.n.25%, DIST.J.h.25%, DIST.J.h.100%

Class A_Thyro (15 selected attributes)

COMP.A.n, COMP.I.p, TRANS.B.h, TRANS.E.n, TRANS.F.n, DIST.A.p.75%, DIST.B.h.50%, DIST.C.n.75%, DIST.E.p.75%, DIST.G.n.25%, DIST.G.h.25%, DIST.H.p.75%, DIST.H.n.50%, DIST.J.n.25%, DIST.J.h.50%

Class C_CalcSense (29 selected attributes)

COMP.A.n, COMP.D.h, COMP.E.p, COMP.F.n, COMP.H.h, COMP.I.p, COMP.I.n, TRANS.F.h, DIST.A.p.first, DIST.A.n.first, DIST.A.n.75%, DIST.B.p.first, DIST.C.n.25%, DIST.C.n.100%, DIST.D.p.first, DIST.D.n.75%, DIST.D.h.100%, DIST.F.h.75%, DIST.G.p.25%, DIST.G.p.50%, DIST.G.h.first, DIST.G.h.75%, DIST.H.p.50%, DIST.H.p.75%, DIST.H.n.75%, DIST.H.h.75%, DIST.I.h.50%, DIST.J.n.100%, DIST.J.h.100%

Note

[1]There is a slight change in the data between the data used for ‘Means of *Z*-values’ and that used in this investigation and ‘Means of *Z*-values with termini’. We found that, in the data used in the former, two duplicate classes exist at the third level. These were removed, reducing the number of classes from 89 to 87.