An Experimental Comparison of Classification Algorithms for the Hierarchical Prediction of Protein Function

Andrew Secker[1], Matthew N. Davies[2], Alex A. Freitas[1],
Jon Timmis[3], Miguel Mendao[3], Darren R. Flower[2]
[1]Computing Laboratory and Centre for BioMedical Informatics,
University of Kent, Canterbury, CT2 7NF, UK
Email: {a.d.secker, a.a.freitas}@kent.ac.uk
[2]Edward Jenner Institute, Compton, Newbury, Berkshire, RG20 7NN, UK
Email: m.davies@mail.cryst.bbk.ac.uk, darren.flower@jenner.ac.uk
[3]Departments of Computer Science and Electronics, University of York, Heslington, YO10 5DD, UK
Email: {jtimmis, miguel}@cs.york.ac.uk

1 Introduction

Some data can be naturally organised as a hierarchy of classes. The classification of data in such a hierarchy poses some unique challenges to data miners such the need for classification at different levels, which may require the use of different characteristics of the data. One particular case of hierarchical classification is the classification of GPCR proteins by their function. G-Protein Coupled Receptors (GPCRs) are important as they can transmit messages from a cell's exterior to its interior, changing that cell's behaviour. For this reason, GPCR proteins are a common target for therapeutic drugs and approximately 50% of all marketed drugs are targeted towards one of the receptors [1].

This paper contributes an improved method of hierarchical classification based on the well-used "top-down" approach. This is tested on a real-world GPCR dataset. A second contribution is the comparison of ten different classification algorithms in the hierarchical prediction of GPCR function with each algorithm being applied in the conventional top-down approach for hierarchical classification. The paper is organised as follows. Section 2 describes GPCR proteins, section 3 introduces hierarchical classification and the top-down approach to classifying data. Section 4 contains a description of the revised system including the proposed method. Section 5 contains experimental protocol and results. Finally section 6 contains some concluding remarks.

2 Classification of GPCRs

This study concerns the prediction of GPCR function where a GPCR is a particular type of protein. Proteins are large molecules comprised of a long chain of amino acids. The order with which these amino acids are chained together is known as the protein's primary sequence. These long chains fold into complex structures allowing them to perform functions. GPCR proteins fold themselves such that some parts of the protein are found inside a cell, while other parts are external. GPCRs are bound by a variety of different molecules (ligands) found outside the cell. The binding activates the GPCR, which in turn binds a G-protein inside the cell, often changing the behaviour of that cell. More than one type of GPCR can interact with more than one kind of G-protein, creating a complex system involving a variety of mechanisms [2].

One of the main aims of bioinformatics is to determine the function of novel protein sequences by comparison with the sequences of genes/proteins whose function has already been established. Conventional bioinformatics typically determines information about a protein function by aligning the protein's primary sequence with other protein sequences or using certain "motifs" (short subsequences of amino acids that typically occur in a given family of proteins) for the same task. This conventional bioinformatics approach may not be appropriate for GPCR function prediction because GPCRs with very different primary sequences may perform the same function.

The most widely used taxonomy of classes for GPCRs, GPCRdb [3], divides the full set of GPCR proteins into six families, designated A-F with class A as the largest human GPCR family. The

GPCRdb classification system is based on the ligand to which the receptor is bound rather than the primary sequence. Previous methods for GPCR classification have included machine learning techniques such as Hidden Markov Models [4] and Support Vector Machines [5]. These previous attempts tend to be based on the primary sequence of the protein. We use an alternative protein data representation based on proteochemometrics, whereby 26 separate physicochemical properties of the protein are used to calculate five empirical "z-values" for all twenty amino acids [6]. These five values can provide a purely numerical description of the protein's physiochemical properties and as such, contain more information than the primary sequence alone, potentially resulting in higher predictive accuracy.

3 Hierarchical Classification

First it is important to define the distinction between flat and hierarchical classification. The vast majority of the classifiers in the literature deal with a flat class structure where a single class is assigned to an example and there is no hierarchical relationship between classes. In hierarchical classification the classes are arranged in a hierarchical structure. An example may be assigned to one class at each of a number of levels of class specialisation. The most general level being immediately below the root of the tree and becoming more specialised as the tree's branches are traversed. In this paper we deal only with structures where each class has exactly 1 parent (i.e. a class tree). A flat class structure will contain, for example, classes A and B, which are both equally different from each other. While, in a hierarchy, some classes are more alike than others. Given the class tree in Figure 1 (a), A.1 and A.2 are more alike than A.1 and B.1 as A.1 and A.2 share a common parent class.

Figure 1: Example of a hierarchical dataset (a) and how that hierarchy may be reflected in a tree of classifiers (b) ready for a top-down approach to classification.

There are a range of strategies available for predicting hierarchical classes [7]. The simplest is to flatten the dataset to one single level so that no superclasses or subclasses are present, then use one of the plethora of standard classification algorithms to predict the class. However, this strategy does not take advantage of the information implicit in the class structure. At the other end of the range is the "big bang" approach. In this case a single (and typically complex) hierarchical classification algorithm is used, which implicitly takes into account the class hierarchy during training. In the test phase, each example may be assigned to one class at each level of the hierarchy by one single application of the learned model. Perhaps due to its complexity, implementations of such an approach are scarce, although one example is [8].

A middle ground between these two strategies is the top-down approach where the hierarchical classification problem is converted into a number of flat classification problems that may be solved independently by running a flat classifier for each.

The top-down approach works as follows. Given, for example, the class tree in Figure 1 (a), a tree of classifiers is built to reflect the structure of the classes, as shown in Figure 1 (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 be equal to the number of levels represented by the class attribute. As shown above, the class tree has two levels and so two levels of classifiers are present. As practically any standard, well known classifier can be used at each node the process of building a hierarchal classifier is greatly simplified. No special classifier must be written to perform the classification (other than the scaffolding required to support a classifier tree). Rather, common well understood classifiers can be used and as such, informed choices can be made about which to use.

To train the classifiers in the hierarchy, all data in the training set is used to train the root classifier. However this is not the case with subsequent classifiers. For example, as the "A class classifier" is only required to classify an instance as A.1 or A.2, only data of class A is used for training. Likewise the B class classifier is trained using only data of class B. When an unknown class test instance is presented to the classifier tree, the root level classifier will assign of the classes at the most general level to the instance. The instance will then be passed to one of the classifiers (A or B) in the next level and so on until the instance is assigned its most specific class by the classifier at a leaf node in the tree. This approach can, therefore, be appealing from the viewpoint of simplicity, however, it suffers from one major inadequacy. That is, any test example misclassified at the top level of the hierarchy has no chance of being assigned to the correct class in any subsequent stages. While this flaw is inherent in the way the top-down approach works, there is no such constraint placed on a big bang algorithm and as such there exists at least the potential for greater classification accuracy using this approach.

4 Extending the Top-down Approach Using Classifier Selection

The manner in which the top-down approach works takes advantage of the hypothesis that some characteristics of the data may be important to discern between two classes at one node of the class tree while being irrelevant at another. For example, certain attribute values may be equal in classes A.1 and A.2 (making this attribute useless to discern between the two) but yield 100% accuracy when used to choose between B.1 and B.2.

The top-down approach to classification exploits this as all classifiers are trained using only data instances of the class they are required to classify between. Despite this variation of the training data at each node in the class tree, in the standard top-down approach the same classification algorithm is used in each node of the class tree. Intuitively, this is unlikely to lead to a maximization of classification accuracy. It is natural to hypothesise that different classifiers may be more suited to different nodes in the class tree. Each type of classifier has its own bias and we hypothesise it is possible to maximise the classification accuracy of the top-down approach by using different classification algorithms in the classifier hierarchy. These classifiers are to be selected in a data-driven manner using the training set. We call this the selective top-down approach. While at the top level it may be true that large differences exist between classes and as such one classifier is particularly accurate, at lower levels a classifier must use much more subtle difference in the data, and as such a different classifier with a bias that exploits this situation is desirable.

The selective top-down approach proceeds as follows. A tree of classifiers is produced much as before, at each node the training data for that node is split into a sub-training and validation set with data instances being assigned randomly. A number of different classifiers are then trained using this sub-training data and tested using the validation set. The classifier which yields the highest classification accuracy in the validation set is selected as the classifier for this node in the class tree. The sub-training and validation sets are then merged to produce the original training set again, and the selected classifier is then re-trained. While appealing at first glance, a cross validation approach is not used when selecting the classifier at each node. In preliminary testing, a 5-fold cross validation technique was found to increase the training time greatly, while the improvement in classification accuracy was not found to have increased with statistical significance compared to a single evaluation of the validation set; and so the latter approach was used. Note that this selective approach effectively produces a hybrid hierarchical classification system, since different nodes in the class tree each use potentially different types of classifiers. At the time of writing no reference could be found where such a hybrid classifier tree had been used before. The following section details the test of this new selective top-down approach and compares it against the standard top-down technique.

5 Protocol and Results

Protein sequences for the GPCR dataset were identified using the Entrez search and retrieval system [9]. The program 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. Text-based searching was used to identify all sequences at the third level of the class tree and higher levels inferred from this classification. The dataset was built from human protein sequences with the exception of Class D proteins, which are found only in fungi. All proteins shorter than 280 amino acids in length were removed in order to eliminate incomplete protein sequences (mean length of protein was 473 amino acids) and all identical sequences within the dataset were removed to avoid redundancy. Any class containing fewer than 10 examples was discarded, which left 8408 examples in total with 89 classes at the most specific (third) level, 38 classes at the second level and 5 classes (GPCR families A-E) at the top level. As such this represents one of the largest GPCR datasets ever constructed. Most of the literature of GPCR class prediction focuses on predicting the first and/or second class levels only [10].

The next challenge was to create predictor attributes from the protein's primary sequence. Recall from Section 2, 5 z-values can be used to represent an amino acid and the primary sequence of a protein consists of a list of amino acids. It is therefore straightforward to substitute each set of z-values for each amino acid in the sequence. However, as the GPCR sequences will vary in length it is essential to normalise these values such that each protein has the same number of predictor attributes. There exist a small number of methods to do this such as the complex "Auto Cross Covariance" [11], although in preliminary tests this method proved inferior in terms of accuracy to the simpler method of computing the mean of each of the 5 z-values (z1…z5). Thus 5 predictor attributes are used to describe each protein in the data set, where each attribute has been derived from the mean of its corresponding value over all amino acids in the protein. A tree of classifiers is produced, with each classifier selected in a greedy fashion, as described in Section 4. The following classifiers were used, where most are described in [12]:

1. Naïve Bayes
2. Bayesian network
3. SMO (a support vector machine [13])
4. 3 nearest neighbours (using Euclidean distance)
5. PART (a decision list [14])
6. J48 (an implementation of C4.5)
7. Naïve Bayes tree (a decision tree with a naïve Bayes classifier at each node)
8. Multi-layer neural network with back propagation
9. AIRS2 (a classifier based on the Artificial Immune System paradigm [15, 16])
10. Conjunctive rule learner

This list of classifiers was carefully chosen to include a wide range of paradigms. All code was written using the WEKA data mining package [12] and the default parameters used for each algorithm.

All experiments were carried out using a 10-fold cross validation approach. Whilst data instances were randomly assigned to folds, care was taken to ensure that at least one instance of each class was present in each fold. This certainly could not be otherwise guaranteed as a small number of classes only contained 10 examples. At each node in the class tree, each classifier from the list above was trained using 80% of the training data (sub-training set) available to that node, and evaluated using the remaining 20% (validation set). Examples were again assigned randomly to these sets. Each entire 10-fold cross validation test was repeated 30 times. There are occasions in this dataset where an internal (non-leaf) node in the class tree contains just 1 child node, i.e. the subset of instances at the internal node contain only 1 class as a child. Without taking this into account the results can be unfairly biased as a classifier will always be able to predict the correct class at that internal node. Any case where only 1 child class is present is ignored, effectively contracting that branch.

The selective top-down classifier was constructed according to the protocol above. In addition, 10 different top-down classifiers were constructed, each algorithm using one of the classifiers available to the selective classifier. The results are shown in Table 1. For each classifier, the predictive accuracy on the test set at each level of the hierarchy is shown. As discussed in Section 3, any example misclassified at one level has no possibility of being correctly classified at deeper levels and therefore misclassifications can be seen to accrue as the level number increases. A value denoting the significance of the difference between the accuracy of the selective approach and each particular algorithm was com-

puted using the corrected resampled t-test as detailed in [12]. This test attempts to eliminate the issues encountered when a standard t-test is used over multiple runs of a cross-validation procedure. Due to space constraints the figures are not reported, instead a shaded cell indicates that the corresponding accuracy value of the selective top-down classifier is significantly greater than the shaded value. The significance threshold was set at 1% and a 2-tailed test was used.

Naïve Bayes	Bayes Net	SMO	3 Nearest Neighbours	PART	J48	NB Tree	Neural Network	AIRS2	Conjunctive Rules	Selective
73.33	77.40	66.44	90.75	89.49	90.37	89.53	66.44	81.66	71.91	90.59
47.74	53.40	38.88	71.59	73.52	73.45	72.34	31.89	57.81	45.51	73.77
23.12	29.83	15.55	55.71	57.90	57.41	55.27	4.15	42.61	9.37	58.08

Table 1: Comparison of predictive accuracy (%) of classifiers at different levels.

Generally the selective approach performs with greater predictive accuracy than a standard top-down classifier with a single type of classifier. In the case of the top level of the 3 nearest neighbours (3NN) classifier a slightly higher accuracy is recorded, but the increase is not statistically significant.

In addition to the accuracy, a classifier tree can be constructed showing, for every position in the tree where a classifier was chosen, the frequency with which a particular classifier was selected in this position. This can provide an explanation for the similar accuracies obtained by the selective approach and the 3NN classifier at the first level. This analysis reveals that 3NN is selected 76% of the time at that node and as there is only one classifier selected as the root, it is expected that the mean accuracy of the top level of the selective top-down classifier should not differ significantly from the 3NN classifier.

The results for J48 were unexpected as J48 is never the most frequently-chosen classifier at any node in the selective classifier tree. However, the results showed that on this dataset J48 does perform well at all levels on the tree. It is worth noting however, that from a user's point of view, to discover this, the user would have to gather results from each of the 10 algorithms separately. By using the selective approach only one algorithm has to be run, and the accuracy of the selective algorithm should always be equal to or greater than the standard approach even when the best classifier is used.

The classifier tree also offers some evidence that selecting different classifiers at different class nodes tends to work better than using the same classifier at each node. It was found that, while the 3NN is predominantly chosen at the top level, at the second level, PART is the most frequently chosen algorithm at 2 of the 3 nodes. While at the third level PART is the most frequently chosen classifier for 3 nodes out of 7, with Naïve Bayes and 3NN both being the most frequent classifier 2 times out of 7. It is possible that 3NN is most suited to the large differences in proteins at the top class level while other algorithms are either more able to exploit the more subtle differences between data items at the second and third class level, or they exhibit better learning when few training items are available.

6 Conclusions and Future Research

Motivated by a real-world problem, that of predicting the hierarchical functions of GPCR proteins, the top-down approach to hierarchical classification was re-visited. It was hypothesised that biases in classifiers could be exploited to increase classification accuracy by using different classifiers at different nodes in the classifier tree with the specific classifier at each node being chosen in a data driven manner. The results of tests on real-world GPCR data showed that there was an improvement in accuracy for this selective hierarchical classifier over all comparison algorithms apart from one.

This paper details a way in which it is possible to increase the accuracy of a top-down classifier but this methodology still contains the problem that any misclassification at one level cannot be rectified at lower levels. While a big bang algorithm for GPCR classification is likely to be complex, it could be written in such a way so that this problem is not encountered. With the results in this paper as a baseline, a big bang style algorithm for GPCR classification is a research direction we might pursue in the future.

Acknowledgments

The authors gratefully acknowledge ESPRC grant EP/D501377/1 (project website: http://www.cs.kent.ac.uk/projects/biasprofs/)

References

[1] D. R. Flower. Modelling G-Protein-Coupled Receptors for Drug Design. Biochim Biophys Acta, 1422:207-234, 1999.

[2] T. E. Hebert and M. Bouvier. Structural and Functional Aspects of G Protein-Coupled Receptor Oligomerization. Biochemical Cell Biology, 76:1-11, 1998.

[3] L. F. Kolakowski Jr. GCRDb: a G-Protein-Coupled Receptor Database. Receptors Channels, 2:1-7, 1994.

[4] S. Möller, J. Vilo and M. D. Croning. Prediction of the Coupling Specificity of G-Protein Coupled Receptors to Their G-Proteins. Bioinformatics, 17(Suppl. 1):S174–S181, 2001.

[5] R. Karchin, K. Karplus and D. Haussler. Classifying G-Protein Coupled Receptors with Support Vector Machines. Bioinformatics, 18:147-159, 2002.

[6] M. Lapinsh, et al. Proteochemometrics Modelling of the Interaction of Amine G-Protein Coupled Receptors with a Diverse Set of Ligands. Molecular Pharmacology, 61(6):1465-1475, 2002.

[7] A.A. Freitas and A. C. P. L. F. de Carvalho. A Tutorial on Hierarchical Classification with Applications in Bioinformatics. In: D. Taniar (Ed.) Research and Trends in Data Mining Technologies and Applications, 175-208, Idea Group, 2007.

[8] Claire and R. D. King. Predicting Gene Function in Saccharomyces Cerevisiae. Bioinformatics, 19(2):ii42-ii49, 2003.

[9] D. L. Wheeler, et al. Database Resources of the National Center for Biotechnology Information. Nucleic Acids Research, 35(database issue), 2007.

[10] Q. B. Gao and Z. Z. Wang. Classification of G-Protein Coupled Receptors at Four Levels. Protein Eng Des Sel, 19(11):511-516, 2006.

[11] M. Lapnish, et al. Classification of G-Protein Coupled Receptors by Alignment-Independent Extraction of Principle Chemical Properties of Primary Amino Acid Sequences. Protein Science, 11:795–805, 2002.

[12] H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco, 2005. 2nd Edition.

[13] S. S. Keerthi, et al. Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation, 13(3):637-649, 2001.

[14] E. Frank and I. H. Witten. Generating Accurate Rule Sets without Global Optimization. In Fifteenth International Conference on Machine Learning, Morgan Kaufmann, 1998.

[15] A. Watkins, J. Timmis and L. Boggess. Artificial Immune Recognition System (AIRS): An Immune-Inspired Supervised Learning Algorithm. Genetic Programming and Evolvable Machines, 5(3):291-317, 2004.

[16] J. Brownlee. WEKA Classification Algorithms, Version 1.6, http://sourceforge.net/projects/wekaclassalgos. Accessed February 2007.