A Comparison of Categorisation Algorithms for Predicting the Cellular Localization Sites of Proteins

Paul Cairns Christian Huyck Ian Mitchell
Wendy Xihyu Wu
AIR Group, School of Computing Science
Middlesex University
The Burroughs, Hendon, London NW4 4BT, UK
p.cairns@mdx.ac.uk

Abstract

A previous attempt to categorize yeast proteins based on certain attributes yielded only a 55% success rate of correct categorisation using a new type of decision procedure [4]. This paper considers using existing soft computing approaches to improve the categorisation. More specifically, learning algorithms based on neural networks, growing cell systems, a rule development algorithm and genetic algorithms are applied to the yeast data. All of the results are at least as good as the original data showing that new problems do not necessarily require new algorithms. More interestingly, as a consequence of using different algorithms, a consistent failure to achieve high success rates actually indicates features of the data rather than the failings of one or other of the algorithms.

1 Introduction

There is currently much interest in finding ways to analyse databases of biological information [2]. Horton & Nakai [4] propose a new method for classifying the localization sites of yeast proteins based on results from various biochemical tests. The claim is that the new method is good because it incorporates existing human knowledge about the classification task and probabilistic reasoning based on real data. Indeed, the described final system gives a correct classification of 55% of the proteins. However, there is no comparison of this approach with any other algorithms including existing, well-understood ones.

This paper re-analyses the yeast protein data using four different algorithms. The algorithms were chosen because they represent a spread of the possible variety of soft computing algorithms available for such tasks, for example, both supervised and unsupervised learning, symbolic and subsymbolic representations as well as clustering and evolutionary algorithms. From the results obtained, it is clear that a new approach was not really required — the existing approaches are at least as good. It is hard to see how incorporating existing human expertise really paid off.

An additional reason for using several algorithms is that where consistently there are many incorrect categorisations, it is not likely to be the algorithm that is unsuitable but rather that there is something inherently problematic in the data. This is discussed further after a description of how each algorithm was applied to the task. First, though, it is necessary to give a brief overview of the actual yeast protein data used.

2 The Yeast Data

The data used was the same as that used by Horton and Nakai [4] and a full description of the data can be found there. It is freely available from the UCI KDD on-line archive [1].

The data represents information on each of 1484 proteins. For each protein, there are eight features that correspond to the results of various tests performed on the proteins. The results of these tests are standardised to be two-place decimals between 0 and 1. Each of the proteins is put into one of 10 different categories depending on where the protein is localized. The categories are denoted by a three letter identifier, for example, CYT denotes cytoplasmic, including cytoskeletal, proteins. As can be see from table 1, the number of proteins in each category varies considerably by up to two orders of magnitude. This was a matter for concern in some of the algorithms.

To make a fair comparison with Horton & Nakai, the results of the categorisation algorithms were evaluated using a 10-fold cross-validation [2]. They obtained a 55% success rate for their final categorisation though it must be noted

Protein category	Number of proteins
CYT	463
NUC	429
MIT	244
ME3	163
ME2	51
ME1	44
EXC	35
VAC	30
POX	20
ERL	5
Total	1484

Table 1. The number of proteins in each category of the yeast data set

Algorithm	% correct categorisations
Growing Cell Structures	55%
Feed Forward Neural Networks	57%
Genetic Algorithms	55%
ERR	56%

Table 2. The percentage of correct yeast categorisations by each approach using a 10-fold test

that simply predicting CYT as the category for all proteins gives a 32% success rate!

3 The Algorithms

There were four approaches used to analyse the yeast data and produce a categorisation for all of the proteins based on the data. These are detailed below with their respective overall results. The results of all four approaches are summarised in table 2.

3.1 Growing Cell Structures

The growing cell structure (GCS) is inspired by Kohonen's self-organising maps [5] by which a network of nodes is structured to represent the training data through unsupervised learning [9]. However, unlike self-organising maps, as the network learns it is able to add new nodes to more accurately represent dense clusters in the data. Post-learning pruning removes redundant or superfluous nodes leaving a two-dimensional network of nodes that reflects the clustering of the data.

The first version of GCS was designed to work on binary attributes and so it was necessary to adapt it to use real-valued inputs. Because of the way in which GCS adds new nodes, it was not necessary to make any allowance for the variation in category size.

The network as it stands only helps to see clusters in the data. However, it is a simple matter to examine what each node represents and assign the node to a category. The network can then be used as a classifier by feeding new data into the GCS and finding out which node most strongly resembles the input data. The input is assigned the same category as the winning node.

The GCS algorithm has several parameters related to those used in self-organising maps. However, results seemed to be generally the same with a variety of settings. In particular, training required only six complete presentations of the training data. The average size of the final network was 86 nodes which naturally broke up into 12 clusters. In this way, GCS can be used to give protein classifications with an accuracy of 55%.

3.2 Feed Forward Neural Networks

Standard backpropagation neural networks (BPNN) have an established history of being used to classify data [3]. In this experiment, as it was not the backprop algorithm that was under investigation, the neural networks were developed on a commercial package, BrainCel [8]. BrainCel works as an add-on to MS Excel.

The inputs were simply the numerical features of each protein. If the final category had been a single output, for example, 1 = CYT, 2 = POX, ... 10 = ERL, errors may creep in owing to the numerical value of the output not having any meaningful relation to the output category. Instead, the network had an output for each category. An output of 1 at a node meant that a given protein belonged to the category corresponding to the node and 0 that it did not. Clearly, in the training data, one and only one output node ever had a value of 1. To classify a protein, the node with the largest output is taken to be the classification of the input protein. This has the added advantage that, if the network incorrectly classifies a protein, it is possible to see which categories are potential "runners-up" as other nodes also have a non-zero output.

There are many parameters to a neural network and it can be difficult to find a combination that performs well on a given task. As is common in most applications, a single layer of hidden nodes was used for all networks. To determine the number of hidden nodes, different configurations were tried in steps of five from five to twenty five. Similar tests with learning rate (often denoted α for BPNN) indicated that that lowest value possible in Braincel, namely 0.05, was optimal. Interestingly, despite the size of the data set, only 200 training epochs were necessary to get good results. More were tried but without further improvement, or for that matter, deterioration.

Unbiased categories are a well-known problem when using backprop [6]. In the first attempt, this was alleviated by simply repeating the data for the under-represented categories so that each category has the same size but there is a great deal of repetition of the data within some categories. This inflated data set was used for ten-fold cross validation but it was clear that the networks were memorising some of the smaller categories and so gained an overall accuracy of 60% on the inflated data set. To compensate for this, the accuracy was calculated in proportion to the size of the category. For example, when a network produced 270 correct categorisations of the 480 yeast in POX (with repetitions), this only counted as 11.25 correct categorisations ($= \frac{270}{480} * |POX|$). This adjusted accuracy only peaked at 49% for networks with 15 hidden nodes.

For comparison, the data was re-categorised without any compensatory strategies in either training or testing. This time the results were better producing an accuracy of 57% for networks with 20 hidden nodes. However, for this case, the network rarely produced any correct answers on the smaller categories.

3.3 Genetic Algorithms

Genetic algorithms are usually considered as a method for producing near-optimal solutions to a problem situation [7]. However, when categorising proteins, the solution is a system that categorises all of the proteins well. This means that the chromosomes in the population must encode a decision procedure for deciding on which proteins belong to which category. Here, a simple decision procedure is used. Each chromosome has ten genes, one for each category. A protein is compared to each gene in turn and whichever gene is closest to the given protein is declared to be the category of the protein. The fitness function promotes those chromosomes that most often give the correct category over all of the training data. After many generations, the fittest chromosome is used to predict the category of each yeast in the testing data.

Quintessentially, the final chromosome can be considered to encode a stereotypical element of each category. The genes were made up of the eight real-valued attributes of the yeast proteins. This encoding has the added advantage that the stereotypes could easily be compared to discover any similarities or differences between the categories.

Because each category had its own stereotype, there was no need to compensate the data for the mismatch in category size. However, in order to breed good stereotypes quickly, the crossover was not the simple one-point crossover usually used. Instead, a crossover point was chosen for each gene, in effect, producing a 10-point crossover where each crossover point was restricted to lie within a gene. With this approach, training typically took 200 generations and overall produced a 55% accuracy.

3.4 Expanding Range Rules

The above three methods are all subsymbolic. This means that there is ultimately no justification for the final form of the decision procedure. By using rules, for instance, the decision procedure can be made explicit. However the question then is how to produce the rules based on the data. This is the principle behind the expanding range rules algorithm (ERR).

The most comprehensive and precise decision procedure would be to individually assign each protein to its correct category. Whilst this would certainly produce good results on the training data, it is unlikely to generalize to noisy or previously unseen data. Instead, the rules have the following form:

IF	Feature1 $\geq \alpha_1$	AND	Feature1 $\leq \beta_1$
AND	Feature2 $\geq \alpha_2$	AND	Feature2 $\leq \beta_2$
$\vdots$			
AND	Feature8 $\geq \alpha_8$	AND	Feature8 $\leq \beta_8$
THEN	Category = XYZ		

Thus, in effect the value of each feature is bounded below and above by an α and a β. Initially, there is a rule per protein; for that rule the value of Feature$N$ is used as the values of both α_N and β_N; the category given by the rule is that of the protein. This means that initially each rule acts as the perfect decision procedure for each individual protein in the set. Interestingly though, if the data is noisy, it is possible to have two rules that directly contradict each other. This, of course, does not mean that the rule is wrong but rather that there can be no fool-proof decision procedure based on the given data.

The rules are generalised by incrementally decreasing (increasing) the α's (β's). The rules are then re-evaluated. If they lead to a misclassification on the training set then the change is reversed. In this way, the rules grow to cover more proteins and this growth allows the size of the rule set to be reduced. After the rules have been expanded, duplicate rules are removed. Of the initial 1488 rules, an average of 1025 are removed this way.

Expanding rules are checked on a portion of the training set while they are expanding. Once all the rules have been fully expanded, and duplicates removed, they are tested on the full training set. The number of correct categorisations by each rule is calculated. Rules that get more wrong than right are removed. On average this leaves about 350 rules. The rules are then sorted by number incorrect and applied to the test set. Each yeast may be categorised by several rules but the rule that produces fewest incorrect answers (on the training set) is taken as the categorisation. This mechanism correctly categorizes 56% of the yeasts.

In practice, there are many factors to be considered such as choosing the percentage of training yeasts to use during the expansion, choosing which rules to delete and the order in which the data is presented to the rules during evaluation.

4 Discussion

All of the approaches produced results that are comparable to those of Horton & Nakai. This calls into question why a new approach was needed to categorise this particular data set. Horton & Nakai's defence is that their approach, through using Bayesian probabilities, allows the incorporation of expert knowledge. Yet, without any expert knowledge at all, we achieved results that were even slightly better.

With GCS and BPNN, it is hard to give any definitive explanation of why the success rate is not higher, as is typical with many neural network systems. However, by looking at the runners up in the BPNN approach and the overlap of clusters in the GCS, it is clear that there are problems inherent in the data. Specifically the three largest protein categories, CYT, NUC and MIT, are commonly confused by both approaches. It seems that the features chosen are insufficient to discriminate between the three classes. This is not so surprising when the three classes make up more than 76% of all the data and results are around 55% accuracy. Nevertheless, it is worth making further analysis in order to see the degree of similarity between the three classes.

Because the BPNN gave a real-value for each category, it was possible to not only see what the runner-up category was but also, these real-values could be considered as confidence factors in the categorisation. The best BPNN did generally correctly predict CYT, NUC and MIT more often than not but when it was wrong, the protein was from one of the other two categories in the majority of cases. Moreover, the average confidence in the wrong predictions was close to the average confidence in the correct predictions (around 55%), though generally slightly lower. Turning to the runner up categories from the network, when the prediction was wrong on CYT, NUC and MIT, the runner up was predominantly correct however the confidence factors were considerably lower, averaging around 33%. Thus, even using the full output of the BPNN, it would not be possible to produce a better categorisation based on confidence factors and the runners up without considerably more analysis.

BPNN is very much a blunt tool for analysing the data in detail but it does indicate that the initial guess as to the source of errors is correct, namely, there is considerable similarity between the three most common categories.

5 Future Work

Both the genetic algorithms and the ERR approach are worth taking further having been developed here as simple ways of applying evolutionary and rule-based methods to categorisation. With the genetic algorithms, it is worth considering ways in which the whole population rather than one chromosome could contribute to the classification of a protein. To this end, work is under way to develop a genetic algorithm where each population member "votes" for the category of a particular protein and the result of the poll is used to determine the final category. As is to be expected, such an algorithm is considerably more complex especially in defining the fitness function because the fitness of an individual is determined by its rôle in the population as a whole. Whether or not such complexity yields better results can only remain to be seen.

There is considerable range for research on the ERR method. It is not particularly well motivated theoretically and this is obviously an area for exploration.

On this particular test there was a wide range of behaviour with one fold getting over 90% and one getting 36%. This is undoubtedly related to the initial training set that is used for generating rules. What makes a good training set for this algorithm is open for consideration, but it should have reasonable numbers of all categories. Additionally, the percentage of yeasts used for rule generation vs. yeasts used for eliminating rules is an area of exploration.

Finally, overgeneralisation is an open question. If no rule applied to a given yeast, the first and largest category was guessed. On most test runs there were few cases of no rules applying. The runs where rules did apply were generally much better than those where no rules applied. This implies that the algorithm is currently overgeneralising. It would be interesting to change the training, and modify the bad rule elimination to reduce this overgeneralisation.

A second line of research would be to make a more direct comparsion between the approaches. For example, are there certain yeast proteins that all approaches are able to correctly classify? Conversely are there ones that all approaches misclassify? This would give further insights into the data perhaps identifying not only the similarity between the three largest classes but also whether particular proteins are generally problematical. It is also noteworthy that GCS found 12 and not 10 clusters in the data. Analysis of this decomposition may give a deeper understanding of the data in light of analyses of the other approaches.

It would also be useful to consider alternative evaluations of the algorithms. Currently, the results are purely a measure of the number of successful categorisations but is this sufficient? If an algorithm entirely fails to correctly predict a particular category then in one sense it could be considered to be very poor, especially if that category were important in some way. Expert insight into the meaning and relevance of each category could be used but this would only apply to this task. We, therefore, propose to develop evaluations of categorisation procedures that appropriately consider the results on each category.

A longer term goal is to make it easier to implement each of the several algorithms and integrate their results in a single package. In this way, it may be possible to obviate the need to invent new algorithms when there is a whole repertoire of perfectly adequate ones already available.

6 Acknowledgments

Many thanks to the UCI KDD database for the use of the yeast data [1].

References

[1] Bay, S. D., *The UCI KDD Archive*, `http://kdd.ics.uci.edu`, Irvine, CA: University of California, Department of Information and Computer Science, 1999

[2] Han, J. and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2001

[3] Haykin, S., *Neural Networks: A Comprehensive Foundation, 2nd edn*, Prentice Hall, Upper Saddle River NJ, 1999

[4] Horton, P. and K. Nakai, "A Probabilistic Classification System for Prediction the Cellular Localization Sites of Proteins", *Intelligent Systems in Molecular Biology*, 1996, pp. 109-115

[5] Kohonen, T., *Self-organising maps, 2nd edn*, Springer-Verlag, Berlin, 1997

[6] Kubat, M. and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection", *Proc. of the 14th Int. Conf. on Machine Learning*, 1997, pp179-186

[7] Mitchell, M., *An Introduction to Genetic Algorithms*, MIT Press, Cambridge MA, 1996

[8] Promised Land Technologies, `http://www.promland.com`

[9] Wu, W. and W. Dubitzky, "Discovering Relevant Knowledge for Clustering Through Incremental Growing Cell Structures," *Proc. of 2nd Int. Conf. on Data Fusion*, 1999, pp120-126