Third International Conference on Advances in Pattern Recognition, 2005

Discovering Predictive Variables when Evolving Cognitive Models

Peter C. R. Lane[1] and Fernand Gobet[2]

[1] School of Computer Science, University of Hertfordshire,
College Lane, HATFIELD AL10 9AB, Hertfordshire, UK
peter.lane@bcs.org.uk
[2] School of Social Sciences and Law, Brunel University,
UXBRIDGE UB8 3PH, Middlesex, UK
fernand.gobet@brunel.ac.uk

Abstract. A non-dominated sorting genetic algorithm is used to evolve models of learning from different theories for multiple tasks. Correlation analysis is performed to identify parameters which affect performance on specific tasks; these are the predictive variables. Mutation is biased so that changes to parameter values tend to preserve values within the population's current range. Experimental results show that optimal models are evolved, and also that uncovering predictive variables is beneficial in improving the rate of convergence.

1 Introduction

Cognitive science aims to devise explanations for the observed behaviour of human or animal participants in different experimental settings. An important component of this science is the construction of computational models which can simulate the observed behaviour. Different classes of models, or *theories*, may be defined based on the underlying representation or learning mechanisms employed. Optimisation with single models on specific tasks has been shown to produce better results than hand optimisation [1, 2].

In previous work, we have formalised the process of developing robust computational models, applicable to multiple domains [3, 4]. This framework enables us to treat the problem of finding optimal models as one of multi-criteria optimisation, and so apply an evolutionary technique to develop cognitive models. We use a non-dominated sorting genetic algorithm (NDSGA) [5, 6] to locate the set of models which are not outperformed on all tasks (taken from categorisation experiments) by any other.

We continue this paper by introducing the psychological data on categorisation in Section 2, describing the classes of models which we explore in Section 3, and then introducing our evolutionary system in Section 4. Section 5 describes our technique of attempting to locate important variables through correlation analysis. Section 6 discusses some experimental results in developing a model of categorisation. The paper is completed with a discussion section and conclusions.

		P($R_A|E_i$)			
EXAMPLE	Value	1ST	2ND	AVG	TIME (S)
E1	1 1 1 0	0.78	0.97	0.83	1.11
E2	1 0 1 0	0.88	0.97	0.82	1.34
E3	1 0 1 1	0.81	0.92	0.89	1.08
E4	1 1 0 1	0.88	0.81	0.89	1.27
E5	0 1 1 1	0.81	0.72	0.74	1.07
E6	1 1 0 0	0.16	0.33	0.30	1.30
E7	0 1 1 0	0.16	0.28	0.28	1.08
E8	0 0 0 1	0.12	0.03	0.15	1.13
E9	0 0 0 0	0.03	0.05	0.11	1.19
E10	1 0 0 1	0.59	0.72	0.62	
E11	1 0 0 0	0.31	0.56	0.40	
E12	1 1 1 1	0.94	0.98	0.88	
E13	0 0 1 0	0.34	0.23	0.34	
E14	0 1 0 1	0.50	0.27	0.40	
E15	0 0 1 1	0.62	0.39	0.55	
E16	0 1 0 0	0.16	0.09	0.17	

Table 1. Target behaviours of the 5-4 structure. The first column labels the examples, the second major column gives the probability of responding with category A given that example, and the final column gives the average response time in one experiment. (We show data from two specific experiments, labelled '1ST' and '2ND', and the average data ('AVG') for $P(R_A|E_i)$; classification time was not collected for items E_{10} to E_{16}.)

2 Psychological data

The problem of categorisation is one of assigning categories to items, and has been widely studied by psychologists and computer scientists for several decades. From a machine learning perspective, the problem is to minimise the error when categorising new examples. However, from a psychological perspective, the problem is much more subtle. Firstly, the aim of a model is to produce a similar pattern of data to that obtained by human participants in the experiment. Secondly, the data to be obtained may be of various kinds: for example, the proportion of correct responses, the time to make a response, or the number of errors during training.

We use data from an experiment called the 5-4 structure, and specifically the experimental data collected by Smith and Minda [7] from thirty earlier studies for proportion of correct responses, and timing data gathered by Gobet *et al.* [8]. Table 1 summarises some of the psychological data used within this paper. Each example is represented by selecting the binary values for four attributes. The examples are arranged into three groups: the first group (E1-E5) are examples of category A, the second group (E6-E9) examples of category B, and the third group (E10-E16) are known as the 'transfer' examples. During training, the examples of the first two groups are seen and learnt. Finally, all examples, includ-

ing the training and unseen transfer examples, are presented, and the responses recorded.

Based on the notion of behavioural tests, introduced in [3], we consider the data and how it is compared with the model's performance in each experiment as forming a specific experimental criterion or *task*. The aim of the modeller is to find a model which matches the task as best as possible. For example, a connectionist model may be trained on the examples from the first two groups, and then tested on all the training examples: the output of the model will assign each example into one or other of the categories. By training a collection of models, we can obtain the probability of any given model responding with a category label for each example.

We have described two kinds of experiment against which to judge the behaviour of a computational model: the probability of the model making an error on any given item, and the time to produce a response. Each of these experiments produces, from the model, a behavioural measurement when presented with the examples. There are many ways of quantifying the degree to which a model's performance matches that of the human experimental participants: the degree of match is known as the *fitness* of the model in that experiment. We use two standard techniques: the sum-squared error (SSE) and the average absolute deviation (AAD) of the model's responses from the observed data. SSE is computed by taking the sum of the squared difference between the model's response and the target response across the examples. AAD is the average of the absolute difference between the model's response and the target response.

An individual behavioural test is used to compute the fitness of a model to some behaviour, and thus requires the behavioural data and the fitness comparison method to be specified. Later in the paper, we shall refer to behavioural tests as, for example, 'SSE Time'. This means that we are measuring the deviation of the model's timing responses using the sum-squared error fitness function. Similarly, 'AAD 1st' refers to the average absolute deviation on the first set of data from Table 1 on the probability of responding with category A, etc.

3 Computational models

We use three different classes of model, all of which are capable of performing the categorisation experiment, and all typical of the kinds of model used within computational modelling. We briefly introduce each class of model below, but first we describe how the models are used within our optimisation technique.

The critical factor behind our technique is that examples of each class can be created by selecting values for the *parameters* which determine how the model performs. For example, the mathematical models have weights, which signify the relative importance of each observed attribute, and connectionist networks have a parameter for the learning rate. The aim of modelling is to find those parameter settings which enable the model to reproduce the observed behaviour (the figures in Table 1) as closely as possible. The novel challenge which we address is to attempt to model multiple kinds of task with each model, and

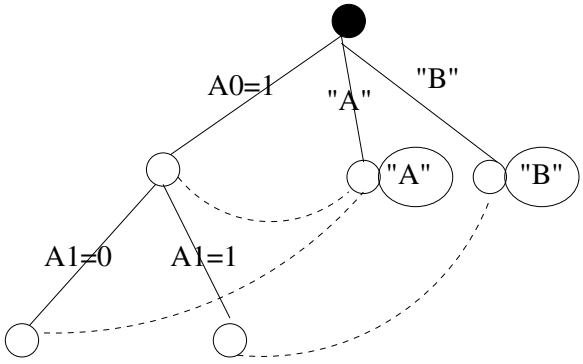

Fig. 1. An example discrimination network, classifying the 5-4 structure.

also to locate those models which perform best when compared with the other models.

3.1 Mathematical models

For the class of mathematical models, we use two of the eight models considered by Smith and Minda [7], specifically the *context* model, which uses all previously seen examples to determine its response to a novel example, and the *prototype* model, which forms an overall template for examples of each category, determining its response by how close a novel example fits each template. All of the mathematical models use a formula to provide the probability of responding with category A given certain sets of training data.

The mathematical models are determined by seven parameters:

weights four weights determine the relative significance of each of the attributes defining the examples.

sensitivity is a factor used to scale the response to the observed attribute values.

guessing is a parameter used to capture the fact that people sometimes simply guess a category, without doing any reasoning.

time is used to capture the time required to make a classification.

3.2 Discrimination-network models

Discrimination-network models, such as EPAM [8], or CHREST [9], have had a long history within cognitive science. Their strength is in modelling the incremental processes of learning and classification which underpin human behaviour in the categorisation experiments. Fig. 1 illustrates a sample discrimination network, learnt by CHREST when trained on data from the 5-4 experiment. Information is stored as chunks within individual nodes. Tests on the links between

nodes are used when sorting a pattern from the root node (the black disc). The dashed links represent *naming links*, which are used by CHREST to associate categories with perceived information. The discrimination network is built up incrementally as the model is given each training example.

There are three parameters used within the model:

learning probability determines the likelihood that CHREST will learn a given training pattern.
reaction time determines how long it takes CHREST to perceive and react to a new example.
sorting time determines the time to match and pass along a test link.

3.3 Connectionist models

The typical connectionist network [10] comprises a set of nodes interconnected by weighted links. The links pass activation between the nodes, and each node uses an activation function to determine its own output based on the input. We use a single-unit perceptron to model the categorisation experiment, with four input links to capture the four attribute values, and the activation of the perceptron used as the model's output category.

There are four parameters for the perceptron model:

theta is the output threshold value for the perceptron.
eta is the learning rate.
learning probability determines the likelihood that the network will learn a given training pattern.
time is used to capture the time required to make a classification.

4 Multi-Criteria Optimisation with a Non-Dominated Sorting Genetic Algorithm

We define a space $\mathcal{M}$ of cognitive models by collecting together the abstract space of models of the four theories. Thus, a model, $m \in \mathcal{M}$, will be a specific set of parameter values for one of the classes of theories. Each task described in Section 2 is defined as a function, $f_i(m)$, which produces the fitness of a model for that task. We also assume that we aim to minimise $f_i(m) \geq 0$.

The presence of multiple constraints, f_i, makes the problem a multi-criteria optimisation problem. One of the key challenges is to define 'optimal', because two models may outperform each other on different constraints. Our aim is instead to obtain the *set* of models which are not worse in *all* constraints than any other model. Formally, we say that model m_1 *dominates* model m_2 if:

$$\forall i \bullet f_i(m_1) \leq f_i(m_2) \ \wedge \ \exists j \bullet f_j(m_1) < f_j(m_2)$$

In other words, m_1 does at least as well as m_2 everywhere, but there is at least one constraint in which m_1 does better.

1. Four separate, equal-sized populations, are created, each population representing a random collection of models from one of the four theory types.
2. The four populations are pooled, into a set $\mathcal{P}$, and the following four sets are extracted:
   set 1 the non-dominated members of $\mathcal{P}$
   set 2 the non-dominated members of $\mathcal{P} \setminus$ (set 1)
   set 3 the non-dominated members of $\mathcal{P} \setminus$ (set 1 $\cup$ set 2)
   set 4 the remaining elements, $\mathcal{P} \setminus$ (set 1 $\cup$ set 2 $\cup$ set 3)
3. Four new populations are created, each population consisting of models from a single theory, and each of equal size. The populations are constructed by cross-over, using relevant individuals from each of the four sets as parents: items in set 1 are twice as likely to be selected as from set 2, etc. Members of set 1 are retained.
4. Mutation is performed with probability *mutate* on the whole population.
5. The process begins again at step 2, until the maximum number of cycles has been reached.

Fig. 2. Modified non-dominated sorting genetic algorithm.

NDSGA is a standard genetic algorithm using the property of non-dominance as a fitness function. Essentially, cross-over is performed across the entire population, but the selection of parents is biased towards those individuals which are not dominated in the current population. We also require the algorithm to maintain populations of each theory type, for the purposes of comparison. Our adapted NDSGA is described in Fig. 2.

5 Discovering Predictive Variables

Each class of model is defined by a set of parameters, or variables, with varying parameter values defining different models. Each model applied to a different task generates a different performance against that task, however it is likely that the importance of each parameter will vary with the task. For instance, timing parameters will clearly be important in tasks measuring response time, but they may also be critical in determining accuracy where training examples are only presented for fixed amounts of time. The question we ask here is whether our genetic algorithm for evolving cognitive models can also identify those parameters which are 'predictive', in the sense of being strongly correlated with performance on specific tasks.

Fig. 3 illustrates the performance of the time parameter for 10,000 instances of the connectionist type of model in two different tasks. As is readily apparent, the performance of the model has a clear global minimum for task 'AAD Time' (on the right), but no such optimum value is apparent for task 'SSE 1st' (on the left). We use a statistical measure, Pearson's product moment correlation, to locate those parameters which take on optimal values in individual tasks. By storing every model tested by the genetic algorithm, along with its fitness on all of the tasks, we test the degree to which any parameter's value correlates with

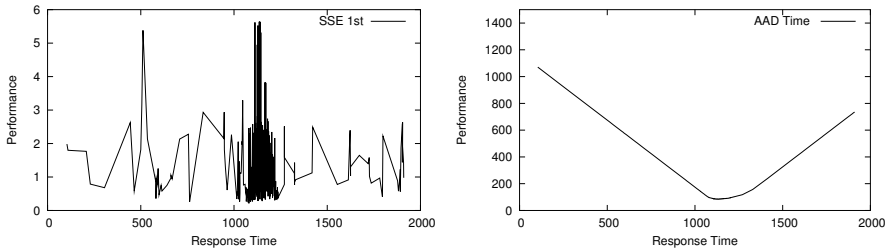

Fig. 3. Graph of performance against parameter value on two tasks

task performance. Specifically, we locate the value of the parameter, p, in the stored models which corresponds to the lowest fitness value for each test. The degree of correlation is then computed between the value of the fitness and the absolute difference of each parameter value from p. A high degree of correlation (> 0.8) means that the parameter acts like the right-hand side of Fig. 3.

The correlated values are used in two ways. Firstly, reporting the correlations is useful in providing additional explanation as to where and why a particular model does well in any given task. Secondly, the stored best values are used to bias mutation of an individual in the population. During mutation of a given parameter, if there is a best value stored, then mutation will pick a new random value near to the best value, in half the cases. In the other half, a new random value is chosen near the current one.

6 Experiments

There were two aims to these experiments: firstly, to confirm that good models were found by the evolutionary process, and secondly, to explore whether the discovery of predictive variables had an impact on the convergence rate. The experiments were run using a population of 200 models, 50 of each type, and allowing training to progress for 500 cycles. The probability of mutation was set at 1.0. Two sets of experiments were run, one with the transfer of bias turned off, and one with it turned on.

Table 2 summarises, for selected tasks, which model type achieved the best performance on that task. Interestingly, different model types do the best on different tasks. The performance measures found here are comparable with the best models in the literature, and, in some cases, exceed the fits obtained. For example, Gobet *et al.* [8] achieved a fit around 0.300 for the 'AAD Time' criterion, whereas our system produced a fit of 0.069. These data confirm that our system discovers useful cognitive models in line with those obtained by practitioners.

Table 3 lists selected correlations detected by the system. The correlations mostly agree with what might be expected. There are strong, and readily apparent, correlations between the different models' timing parameters and the task involving a timed response. No behavioural effects were apparent for most of the

Task	Performance	Class
SSE Avg	0.082	Connectionist
AAD Avg	0.057	Connectionist
SSE Time	63814	CHREST
AAD Time	0.069	CHREST

Table 2. Best performance, and model class, on selected individual tasks.

Class	Parameter	Value	Task	Correlation
Context	response time	1174.74	SSE Time	0.97
Prototype	response time	1129.73	AAD Time	0.99
Connectionist	response time	1130.62	AAD Time	0.98
CHREST	reaction time	242.52	AAD Time	0.82
CHREST	sorting time	512.60	AAD Time	0.86

Table 3. (Selected) reported parameter values and tasks with high (> 0.8) correlation.

other parameters. Partly, this is a limitation in the current approach, which only seeks a correlation between one parameter's value and the fitness against a task. Most parameters, such as the weights within the mathematical models, work together to produce a behaviour, and more sophisticated analysis techniques are required to locate such dependencies; individually, their correlation with task performance is of the order of 0.3. However, it may also be the case that for many of the parameters, e.g. the learning rate for a connectionist network, there simply is no correlation of the parameter value with the model's performance. This would make the parameter a 'free' variable in the theory; one needed to make the implementation work but which has no explanatory power.

We explored the effect of using the predictive variables discovered by correlation analysis by comparing the rate of convergence of the average performance of the evolving population both with and without the transfer of bias to mutation. Fig. 4 plots the average performance of the entire population of models against training time for a single task, both with and without the transfer of bias. There is a clear improvement in the rate of convergence of the non-dominated set of models when transfer is included. This is readily explained, as the effect of transfer is to reduce the chance that a model will vary from its optimal value.

7 Discussion

Computational modelling is a complex application, to which evolutionary techniques may be profitably applied, as was first proposed by Ritter [1, 2]. We have formalised this application as one of multi-criteria optimisation, in which models are drawn from classes of theories, and are optimised against separate experimental criteria. Genetic algorithms are an important technique for solving such multi-criteria problems, and Coello [11] provides a useful summary. Unique to

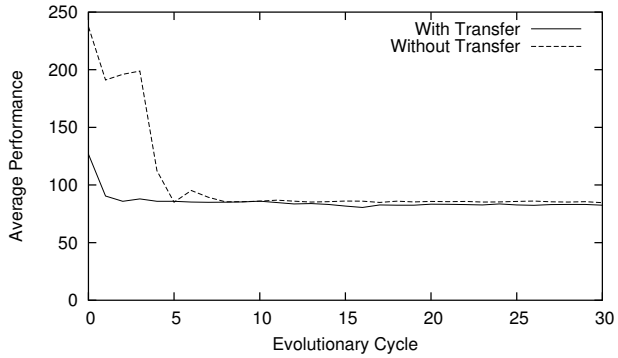

Fig. 4. Convergence rate, with and without transfer of bias, for first 30 learning cycles, of average population performance.

our application is the selection of models from multiple classes of theories, where most approaches to multi-criteria optimisation restrict themselves to individuals drawn from a single class. We have tailored our genetic algorithm to support the evolution of models from multiple theories; some of the difficulties in maintaining useful competing models are discussed in [12].

We have focused on the development of optimal models within established theories. A more complex approach is to evolve new theories. For example, Langley *et al.* [13] have developed a technique for inducing process models from continuous data. Our approach, based on well-defined behavioural tests for specific model types, is readily expanded to include the development of novel theories, although this will increase the complexity of the search space.

Our suggestion in this paper of using correlation techniques to uncover predictive variables improves the convergence rate of the genetic algorithm. Michalski [14] uses stronger machine learning techniques within evolutionary algorithms, suggesting that inductive hypotheses about the performance of specific individuals may be developed. In later work, we intend to extend the range of model types and experiments, and such stronger machine learning techniques may prove beneficial in place of our direct computation of correlations.

8 Conclusions and Further Work

We have described an evolutionary system for developing optimal sets of cognitive models which satisfy multiple experimental criteria. Analysing the evolution of specific model parameters against individual tasks enables the system to pick out optimal values for individual variables. An evolutionary bias in mutation is then employed to guide the system towards these optimal values. Experiments support the value of the technique in locating optimal models across multiple experimental tasks, and also that the identification of predictive variables speeds up the rate of convergence to optimal values.

Further work will focus on widening the range of model types and tasks being explored. As we have argued elsewhere [4], the evolutionary techniques described here are suitable for developing complex models of human behaviour. The selection of predictive variables from the system's own training history will be extended to seek more complex correlations between multiple variables.

Acknowledgements

The authors thank the program chair and three anonymous reviewers for helpful comments which have improved the quality of this paper.

References

1. Ritter, F.E.: Towards fair comparisons of connectionist algorithms through automatically optimized parameter sets. In: Proceedings of the Annual Conference of the Cognitive Science Society, Hillsdale, NJ: Lawrence Erlbaum (1991) 877–881
2. Tor, K., Ritter, F.E.: Using a genetic algorithm to optimize the fit of cognitive models. In: Proceedings of the Sixth International Conference on Cognitive Modeling, Mahwah, NJ: Lawrence Erlbaum (2004) 308–313
3. Lane, P.C.R., Gobet, F.: Developing reproducible and comprehensible computational models. Artificial Intelligence **144** (2003) 251–263
4. Gobet, F., Lane, P.C.R.: A distributed framework for semi-automatically developing architectures of brain and mind. In: Proceedings of the First International Conference on e-Social Science. (2005)
5. Goldberg, D.E.: Genetic Algorithms in Search Optimization and Machine Learning. Reading, MA: Addison-Wesley (1989)
6. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation **2** (1994) 221–248
7. Smith, J.D., Minda, J.P.: Thirty categorization results in search of a model. Journal of Experimental Psychology **26** (2000) 3–27
8. Gobet, F., Richman, H., Staszewski, J., Simon, H.A.: Goals, representations, and strategies in a concept attainment task: The EPAM model. The Psychology of Learning and Motivation **37** (1997) 265–290
9. Gobet, F., Lane, P.C.R., Croker, S.J., Cheng, P.C.H., Jones, G., Oliver, I., Pine, J.M.: Chunking mechanisms in human learning. Trends in Cognitive Science **5** (2001) 236–243
10. McLeod, P., Plunkett, K., Rolls, E.T.: Introduction to Connectionist Modelling of Cognitive Processes. Oxford, UK: Oxford University Press (1998)
11. Coello, C.A.C.: An updated survey of GA-based multiobjective optimization techniques. ACM Computing Surveys **32** (2000)
12. Lane, P.C.R., Gobet, F.: Multi-task learning and transfer: The effect of algorithm representation. In: Proceedings of the ICML-2005 Workshop on Meta-Learning. (2005)
13. Langley, P., Sanchez, J., Todorovski, L., Dzeroski, S.: Inducing process models from continuous data. In: Proceedings of the Nineteenth International Conference on Machine Learning, Sydney: Morgan Kaufmann (2002) 347–54
14. Michalski, R.S.: Learning evolution model: Evolutionary processes guided by machine learning. Machine Learning **38** (2000) 9–40