A Generic Multi-Dimensional Feature Extraction Method using Multiobjective Genetic Programming

Y Zhang and P I Rockett

Technical Report No. VIE 2006/002
Department of Electronic and Electrical Engineering
University of Sheffield

A Generic Multi-dimensional Feature Extraction Method using Multiobjective Genetic Programming

Yang Zhang hegallis@gmail.com
Laboratory for Image and Vision Engineering, Department of Electronic and Electrical Engineering, The University of Sheffield, Sheffield, S1 3JD, UK

Peter I. Rockett p.rockett@sheffield.ac.uk
Laboratory for Image and Vision Engineering, Department of Electronic and Electrical Engineering, The University of Sheffield, Sheffield, S1 3JD, UK

Abstract

In this paper, we present a generic, feature extraction method for pattern classification using multiobjective genetic programming. This not only evolves the (near-)optimal set of mappings from a pattern space to a multi-dimensional decision space, but also simultaneously optimizes the dimensionality of that decision space. The presented framework evolves vector-to-vector feature extractors which maximize the class separability. We demonstrate the efficacy of our approach by making statistically-founded comparisons with a wide variety of established classifier paradigms over a range of datasets and find that for most of the pairwise comparisons, our evolutionary method delivers statistically smaller misclassification errors. At very worst, our method displays no statistical difference in a few pairwise comparisons with established classifier/dataset combinations; crucially, none of the misclassification results produced by our method is worse than any comparator classifier. Although principally focused on feature extraction, feature selection is also performed as an implicit side-effect; we show that both feature extraction and selection are important to the success of our technique. The presented method has the practical consequence of obviating the need to exhaustively evaluate a large family of conventional classifiers when faced with a new pattern recognition problem in order to attain a good classification accuracy.

Keywords

Feature Extraction, Multiobjective Optimization, Genetic Programming, Optimal Dimensionality, Search, Multi-dimensional Mapping, Pattern Recognition

1 Introduction

In the field of pattern recognition, feature extraction received a great deal of attention up to the 1970s after which point, work on increasingly sophisticated classifier paradigms such as neural networks took center stage. Despite advances in classifier design, feature extraction still maintains a key role in pattern recognition since it is commonly observed that classification performance can be substantially improved by appropriate pre-processing of the raw measurement data. Identifying (near-)optimal feature extraction sequences is the central focus of this work.

In a typical pattern recognition system, the vector of raw measurements is *transformed* (or mapped) into a decision space thus forming a new set of *extracted*

features. In many classifier paradigms, however, the feature extraction stage (together with its related/combined feature selection stage) is simply omitted or is implicit in the recognition paradigm - a multilayer perceptron is a good example of a classifier in which there is no readily identifiable feature extraction stage. Feature selection and extraction techniques have been reviewed previously in (Addison et al., 2003; Park et al., 2004). In the present paper we focus primarily on feature extraction but there is inevitably some overlap with the related area of feature selection which we discuss where appropriate.

Although feature extraction has been long-used in pattern recognition, an enduring difficulty with the technique is that designing a feature extractor usually requires deep domain-specific knowledge. For example, most of the work on detecting image features such as edges and corners actually involves hand-crafted feature extraction. Trier et al. (1996) have reviewed eleven popular feature extraction methods used for optical character recognition (although as these authors admit, one of these - template matching - is not really a feature extraction method at all). Of the remaining ten methods, none is applicable to the general pattern recognition problem of classifying n-dimensional vectors, a limitation which is quite typical of the feature extraction literature. Principal component analysis (PCA) and independent component analysis (ICA) have also been widely used as pre-processing stages (see, for example, Leiva-Murillo et al., 2002) although whether this results in improved or degraded classification performance depends on the particular class distributions (Jolliffe, 1986). Critically, the issue of the *optimality* of the extracted features has rarely, if ever, been addressed in the traditional feature extraction literature.

Methods of feature extraction can be understood as wrappers or filters - see (Park et al., 2004). Feature extraction can also be divided into: linear and non-linear techniques. Most prominent among the linear feature extraction methods is principal component analysis (PCA) and related techniques. In PCA the focus is usually on reducing the dimensionality of the problem by projecting the pattern into a lower dimensional sub-space which captures most of the data variability although this is not necessarily synonymous with optimizing any measure of class separability (Jolliffe,1986). The main attraction of methods like PCA is that they are analytically tractable. Of the most notable non-linear feature extraction methods, supervised learning networks, such as multilayer perceptrons, radial basis functions and learning vector quantizers (LVQs) offer model-free or semi-parametric means by which a suitable mapping can be induced over a training set of examples. In these techniques, however, the feature extraction (and selection) stages and the classification stage are subsumed into a single entity. Ensuring optimality with such non-linear methods remains a fraught problem which is typically addressed by cross-validation and related model selection methods.

From the standpoint that feature extraction is a mapping from a measurement space, $\mathbf{x}$ to a decision space $\mathbf{y}$, what is required is a transformation, $\mathbf{x} \rightarrow \mathbf{y}$ which maximizes the separability of the pattern classes in the decision space. Ensuring the optimality of such a mapping is difficult for hand-crafted methods. The required transformation can, however, be arbitrarily well-approximated by a sequence of elementary mappings and identifying such a sequence (or program) of basic transformations has been extensively explored before using genetic programming (GP) in which typically,

a chromosome is encoded as a parse tree. Indeed, GP has been used previously to optimize feature extraction and selection stages (Bot, 2001; Kotani et al., 1999; Smith & Bull, 2005; Guo & Nandi, 2006).

Ebner (Ebner, 1998; Ebner & Zell, 1999) has used GP to evolve operators for image processing while Bot (Bot, 2001) has evolved transformed features, one-at-a-time, for a k-nearest neighbor (k-NN) classifier, utilizing each new feature only if it improved the overall classification performance by more than a user-defined threshold. As Bot himself notes, his approach is a greedy, therefore, sub-optimal algorithm. Tackett (1993) used GP to produce a symbolic expression for the classification of images while Koza (1994) generated character detectors. Sherrah et al. (1997) describe an evolutionary pre-processor (EPrep) which searches for an optimal feature extractor by minimizing the misclassification error over three randomly selected classifiers, using the validation error as a single scalar fitness value. Harris (1997) adopted a similar strategy using co-evolution although Harris's approach evolves the feature extraction stage in tandem with a classifier in such as way as to make it impossible to disentangle a poor feature extractor paired with an excellent classifier, or vice versa. Smith & Bull (2005) developed a hybrid feature construction and selection method using GP together with a GA using a single objective and Guo & Nandi (2006) have recently optimized a modified Fisher discriminant using GP, although demonstrated their method on only a single dataset.

Kotani et al. (1999) determined the optimal polynomial combinations of raw features to pass to a k-NN classifier which improved classification performance. Subsequently, Harvey et al. (2002) evolved sequences of transformations for the classification of synthetic aperture radar images using GP while Krawiec (2002) evolved vector-to-vector mappings of pre-determined length, introducing a protection mechanism to preserve valuable building blocks. Unfortunately, Krawiec's proposed protection mechanism clearly contributed to over-fitting; further, some of this author's classification results after feature extraction were *worse* than just using the raw data. Consequently, any notions of optimality in Krawiec's work need to be viewed with circumspection. Recently, Guo et al. (2005) evolved a vector of features for a condition monitoring task but it is unclear if the elements of the vector of decision variables were simultaneously evolved or were selected by hand afterwards. Finally, in previous work (Zhang & Rockett, 2005b) we have used multiobjective GP to learn optimal feature extraction stages which project the high-dimensional pattern vector to a one-dimensional decision space in which a simple threshold was used to separate classes. This method yielded classification results which were better than, or at very worst, statistically identical to a wide range of conventional classifiers over a range of datasets.

Previous research on GP feature extraction can be broadly divided into two categories: 1) Evolving a distinct feature extraction stage, the transformed features from which are passed to a conventional classifier, or 2) Evolving a classifier (with embedded and implicit feature extraction) which directly outputs a class label.

We have opted for the former category since we contend that all the available computational resource should be put into solving the unknown problem of optimal feature extraction. By its very nature of being a stochastic procedure, evolutionary search provides *near*-optimal rather than provably, mathematically optimal solutions

(although in most fields, evolutionary methods yield a sufficiently close approximation to optimal). We see little justification for simultaneously evolving or co-evolving classifiers which will only be close-to-optimal when the theory underpinning classifiers has been very extensively studied and is well understood. A secondary reason for evolving only a feature extraction stage is that we will be dealing with a smaller search space than trying to evolve an intertwined feature detection/classifier combination. Consequently, we should expect more rapid convergence. A corollary of evolving only a feature extraction stage is that we need to employ a conventional classifier which has to be trained *within* the evolutionary loop as part of the fitness evaluation step. In this work we have used a Fisher linear discriminant (FLD) since this classifier is fast to train while having been shown to give reasonable results over a wide range of pattern recognition problems - see (Lim et al., 2000), for example.

A further significant distinction between the present work and the research cited above - apart from (Zhang & Rockett, 2005b) - is that all other GP feature extraction has been carried-out using a *single* objective, typically classification error. One of the widely-observed practical problems with GP optimization is the tendency for the evolved trees to continue growing in size, a phenomenon known as *tree bloat*. Aside from the computational burden ever-growing trees impose, tree bloat can be interpreted as *over-fitting*, a problem common to all data modeling. Ekárt & Németh (2001) have shown that tree bloat can be effectively suppressed by using multiple objectives, where one of the objectives is a measure of tree size (i.e. the complexity of the candidate solution). This imposes a selective pressure which favors smaller trees and tends to produce better generalization than larger trees, an observation which can be understood in terms of both Tikhonov regularization and Occam's razor. Zhang & Rockett (2005b) have previously used a multiobjective approach in evolving feature detection stages with a high degree of success. The use of tree complexity measures to reduce bloat has also been examined by de Jong & Pollack (2003) who studied the inter-relationship between complexity minimization and genetic diversity. Along with the fact that we have optimized the *dimensionality* of the evolved decision space, the key distinguishing factor of the present work is our use of multiple objectives, one of which encourages parsimonious solutions which will therefore tend to generalize better than the unconstrained solutions obtained by other workers.

In terms of which multiobjective evolutionary algorithm to employ, we have shown previously that the steady-state Pareto converging genetic algorithm (PCGA) (Kumar & Rockett, 2002; Zhang & Rockett, 2006b) produces smaller evolved trees and by implication, superior generalization to competitor techniques. Consequently, we have used PCGP, a GP adaptation of the PCGA evolutionary strategy. More generally, multiobjective evolutionary algorithms have been well reviewed by Coello (2000).

Our objective in the present work has been to evolve the (near-)optimal series of mathematical transformations which project the pattern data into a new, multi-dimensional decision space such that we obtain maximized class separability but completely independent of domain knowledge. Although we have previously published results on projecting the patterns into a 1D decision space (Zhang & Rockett, 2005b), it is well-known that for a given size of training dataset there is an optimal dimensionality for which misclassification error is minimized. This optimal dimensionality is, however, data dependent and needs to be determined as part of

the evolutionary process rather than fixed a priori as in (Krawiec, 2002). We can thus state the problem to be addressed as: Evolving the optimal n-to-m mapping from an n-dimensional raw pattern space to the m-dimensional decision space which maximizes the class separability in the decision space and where m itself is an unknown parameter to be (implicitly) optimized. Furthermore, we require the method to be generic and domain-independent, and to make no assumptions about the data distributions.

The remainder of this paper is organized as follows: We present our generic evolutionary framework for optimal feature extraction in Section 2. In Section 3, we present the results from applying the method to a range of problems from the UCI Machine Learning (Blake & Merz, 1998) and StatLog (Michie et al., 1994) databases as well as making statistical comparisons with eight established (non-evolutionary) classifiers. In this section we also discuss the properties of the evolved mappings. Finally, we offer some conclusions and suggestions for future work in Section 4.

2 Methodology

We seek the mapping from the input space to an m-dimensional decision space, where the optimal value of m has to be determined during the learning phase. Constructing an effective chromosomal tree structure to represent a solution is critical to the success of the method and two fundamental choices are available: Firstly, we could employ multiple trees, one per dimension of the decision space but such a representation would require a new set of genetic operators to allow the breeding of uncoupled feature transforming trees. The second possible chromosome implementation is to employ a vectorizing multi-tree representation. Langdon (1998) developed a multi-tree program with redefined genetic operations to reduce the extent of building block disruption. Similar operations were also used by Koza (1994) in his work involving automatically defined functions (ADFs). Sherrah et al. (1997) has attempted to address the issue of generating n-to-m mappings ($m > 1$) by applying a multi-tree chromosome structure but had to impose some severe restrictions to compensate for the limitation of a single fitness objective. Muni et al. (2004) applied a multi-tree representation to produce a multi-class classifier but actually, simultaneously evolved multiple independent dichotomizers.

Of the two possible chromosomal representations we have followed the second option of using a single, vectorizing tree in order to reuse the highly effective genetic operators we have devised previously (Zhang & Rockett, 2005b). To facilitate this representation we have added two special node types.

The first special node type we have added is a *Root* node. As the name suggests, each vectorizing tree comprises exactly one Root node placed at the root of the tree. In reality, this 'node' is just an array (of arbitrarily large length) of pointers to genuine feature transforming sub-trees, each of which generates one element of the m-dimensional decision space vector.

The second type of special node we have added is the *Dummy* node - which is broadly equivalent to a null node - and allows uniformity of treatment with the crossover and mutation operators used previously (Zhang & Rockett, 2005b). For the purposes of genetic manipulations, a Dummy node is treated in the same way as a

terminal node - in effect, it is a special kind of terminal node. During tree evaluation, however, it returns a zero value, hence a Dummy node contributes no discriminatory power to the classification task - it is a mere placeholder. If a child of the Root node evolves to become a Dummy node, this reduces the dimensionality of the decision space by one. Similarly, if one of the Root's children which was previously a Dummy node evolves to a real sub-tree then this increases the dimensionality of the decision space by one. In counting the m-value of a tree, we simply ignore all the Dummy children of the Root node. The vectorizing tree structure employed here is illustrated in Figure 1.

2.1 Crossover

We define crossover and mutation operations on this vectorizing tree as minor extensions to those used successfully in (Zhang & Rockett, 2005b). We use *depth-fair* crossover (Ito et al., 1998; Zhang & Rockett, 2005a) except we avoid selecting the Root node since a vectorizing tree can have only one Root node at its root. Having selected a tree depth, d, we select one of the sub-trees at this depth, biased in their complexity - that is, we prefer to swap trees with higher node counts. The depth-fair crossover procedure (Ito et al., 1998) can be summarized as follows:

1. Starting with a depth in a tree, $d \in [0 \ldots d_{max}]$, we assign a *bias* to each depth in the tree of $1/2^d$. These quantities are termed *depth selection ratios* in (Ito et al., 1998).
2. A uniformly distributed random number, $r_1 \in [0 \ldots 1]$ is chosen. We select the depth, d, at which to perform the crossover from the relation:

$$r_1 \in \left[1/2^d \ldots 1/2^{d-1}\right]$$

3. Given the tree has n sub-trees at the selected depth level, d, we again pick a random number, $r_2 \in [0 \ldots 1]$ from a uniform distribution and select for crossover, the p-th sub-tree for which:

$$\frac{\sum_{i=1}^{p-1} Size(i)}{\sum_{i=1}^{n} Size(i)} < r_2 \leq \frac{\sum_{i=1}^{p} Size(i)}{\sum_{i=1}^{n} Size(i)}$$

where $Size(i)$ is the number of nodes contained in the i-th sub-tree.

In addition, we have found that in order to preserve useful genetic building blocks and speed of convergence, it is helpful to sometimes invoke a constrained type of crossover, *sub-tree preservation crossover*, to operate on the corresponding sub-trees of the two parents. With some probability, we choose sub-tree preservation crossover under which we select two parents, A and B as normal. In normal, unconstrained crossover we would select two splicing points from anywhere within A and B and exchange the sub-trees. In sub-tree preservation crossover, however, we first perform depth-fair crossover between sub-tree 1 of A and sub-tree 1 of B, then between sub-tree

2 of A and sub-tree 2 of B, and so on up to sub-tree q, where $q = \min[m_A, m_B]$, where $m_{A,B}$ are the dimensionalities of the two parents, A and B. We have found this sub-tree preservation crossover refinement to be highly effective in speeding convergence and by implication, preserving valuable building blocks.

2.2 Mutation

We have used the same depth-fair, size-dependent mutation operator as (Zhang & Rockett, 2005a) in which a single mutation point is selected in the same manner as crossover and the selected sub-tree is replaced with a new, randomly-created tree. We add the extension that if the Root node is selected, a whole new random tree of random dimensionality is generated. After the population has been initialized, it is the mutation operator which is responsible for introducing new Dummy nodes into a chromosome.

The initial population was generated with half the trees of some maximum dimensionality ($m = 50$, here) and the other half being trees of random dimensionality ($m \in [1...50]$). All sub-trees were generated with a random initial depth in the range $[1...5]$ although subsequently, they were then free to grow without limit. We did not impose any maximum tree depth during evolution.

We term the present GP framework *multi-dimensional multiobjective genetic programming* (MMOGP) to distinguish it from our previous MOGP work (Zhang & Rockett, 2005a; Zhang & Rockett, 2005b). It is also distinct from other multi-output GP tree structures such as MRtree (Zhang & Zhang, 2004) and EPrep (Sherrah et al., 1997). In the present work, the input pattern vector is projected into a new, real-valued decision space whereas in MRtree, the output vector comprises tentative class labels. In EPrep, the output vector elements are extracted from various locations over the tree using special Output Points which are treated the same as other nodes during evolution. Outputs extracted by Output Points at lower levels of the EPrep tree, however, will be strongly correlated with those collected at higher levels. This may (partly) explain why the performance of EPrep is very variable across different problems.

2.3 Objective Vector

We have employed a three-dimensional objective vector within a Pareto optimizing framework in which each of the three objectives exerts independent selective pressure on the population. Ekárt & Németh (2001) have demonstrated that using tree complexity as one objective provides selective pressure which suppresses tree bloat; we too have found this objective to be highly effective (Zhang & Rockett, 2005a; Zhang & Rockett, 2005b) although see also (de Jong & Pollack, 2003). In the present setting, however, simply using the total number of tree nodes will create a hidden bias in favor of minimizing the dimensionality of the decision space since trees with fewer dimensions will tend to contain fewer nodes overall. Minimizing the number of dimensions is not something we necessarily wish to do. Consequently, we have used a tree complexity measure which seeks to minimize the *average* number of nodes per dimension:

$$Complexity = \frac{1}{m}\sum_{i=1}^{m} Size(i)$$

where m is the dimensionality of the decision space vector and $Size(i)$ is the number of nodes in the i-th sub-tree. In our preliminary experiments, we found out that without a complexity objective, the tree bloated in an unconstrained manner. In addition, on hard problems the evolution frequently stagnated yielding trees with very poor generalization errors.

Our second objective is the conventional misclassification error (so-called $0/1$ loss) which we obviously wish to minimize. To evaluate this objective for a candidate solution, we project each of the patterns in the training set into the candidate m-dimensional decision space and then use a Fisher linear discriminant (FLD) (Duda et al., 2001) to predict a class label based on these m-dimensional decision space patterns. The misclassification error is simply the fraction of training set examples which are incorrectly labeled. Fisher's linear discriminant projects the m-dimensional data into a one dimensional space in which the class-conditioned distributions are maximally separated; we have used the conventional Fisher criterion, $J(\mathbf{w})$:

$$J(\mathbf{w}) = \frac{|p_1 - p_2|^2}{s_1^2 + s_2^2}$$

where $\mathbf{w}$ is the projection vector, $p_{1,2}$ are the sample means of the two projected class-conditioned densities and $s_1^2 + s_2^2$ is the sum of the within-class scatter. Key for the present application, the maximization of $J(\mathbf{w})$ can be performed rapidly in closed form (Duda et al., 2001) which makes a minimal contribution to the time for one iteration. The final quantity we need in the FLD training is the scalar decision threshold in the one-dimensional projected Fisher direction and this we determine using golden section search, terminating when the misclassification rate does not improve with further refinement of the threshold. Although the FLD is a simple and rapidly trained classifier, it is well-known that occasionally it fails to provide any meaningful class separation. In such a situation, the lack of selective pressure will slow the evolution although in our experience, this shortcoming is more than offset by the advantage of fast training.

In the previous MOGP work (Zhang & Rockett, 2005b) we projected the raw pattern data directly into a 1D decision space in which a simple threshold sufficed to determine a class label. One of the observations we made in this earlier work was that relying solely on misclassification rate to drive the evolution was slow since in the initial stages, most of the randomly-created population were ill-suited to the task and produced very high error rates. Hence initially, there was little selective pressure to drive the evolution. Consequently, we added an estimate of the Bayes error in the projected 1D decision space (calculated by histogram overlap) which proved much more sensitive at differentiating individuals with slightly more promise than others despite both having very high misclassification error. It is worth noting that the Bayes error is a fundamental lower bound on misclassification error independent of the classifier employed. (Incidentally, although the Bayes error objective is very beneficial in the early stages of evolution, if used on its own without misclassification error it tends to lead to *final* solutions which are pathological – see (Zhang & Rockett, 2005b) for further details.) In the present MMOGP setting, estimating the Bayes error in the m-dimensional decision space is not practical so we have used the straightforward method of calculating the overlap of the class-conditioned PDFs in the 1D Fisher projection direction. When all the training set patterns from each class in a

two-class problem are projected onto the 1D Fisher direction, they will form two class-conditioned PDFs. We calculate a Bayes-error-like measure by histogramming these two class-conditioned PDFs and calculating the overlap between them; minimizing the histogram overlap is equivalent to maximizing the class separability. Consequently, this third and final objective is not strictly an estimate of Bayes error since it is not calculated in the mD decision space although it has the same effect as the Bayes error in the 1D MOGP case.

2.4 Hierarchial Feature Extraction for Multi-class Problems

As presented above, our strategy is intrinsically a two-class method in that it uses an FLD dichotomizer in the decision space. Several of the datasets we consider in the next section have multiple classes and we handle these by a hierarchical decomposition into a sequence of two-class problems, as follows: For an h-class problem, first we deduce a tree to discriminate between one (arbitrarily chosen) class and the other $(h-1)$ classes. We then generate a tree to distinguish between one of the remaining $(h-1)$ classes and the other $(h-2)$ classes, and so on until we are left with only two classes. At this point we produce a classifier to discriminate one of these classes. As a very final step, we train a classifier to differentiate this final class from any other unclassified patterns. This last step is necessary due to the effects of (inevitable) type I errors; if we simply trained $(h-1)$ classifiers and assigned any remaining patterns by default to the last class left, many of these patterns would be incorrectly labeled.

Binary decompositions of multi-class problems possess the well-known drawback of yielding multiply labeled and unlabeled patterns. It is likely that some patterns will be assigned more than one label while others will remain completely unlabeled (Bailey, 2001). Although techniques such as error correcting output codes have been designed based on domain knowledge to resolve such ambiguities, these introduce further complexity. Here we resolve labeling ambiguities using a measure of labeling confidence. After a pattern is projected into the Fisher direction it is assigned a class label based on its distance from the decision threshold value. We treat the distance of a pattern to the decision threshold as a confidence level measure (CLM). For patterns classified as belonging to only one class, the confidence level measure is not needed and is therefore ignored. For patterns which are multiply-labeled, the confidence level measures for each labeling decision are compared and the pattern assigned to the class which maximizes its CLM. In other words, it is given the class label in which we have the greatest confidence. For unlabeled patterns, all CLMs are negative since they fall on the 'wrong' side of the decision boundary for a positive outcome. In the case of patterns which are unlabeled, we assign the class label in which we have the greatest confidence; that is, the label for which the *magnitude* of the CLM is smallest. Thus we assign the label for which the pattern comes closest to the respective decision boundary, albeit that it falls on the wrong side of the boundary. This use of CLMs can be related to maximizing the posterior probability - see Zhang & Rockett (2006a) for further details of multi-class hierarchical decomposition.

The MMOGP parameters used throughout this work are summarized in Table 1. The "if-then-else" node returns the value of the second child if the first child value is greater than zero, otherwise the third child value is returned. The "max" node returns the larger value from its two successors while "min" returns the smaller. If both

Table 1: MMOGP (PCGP) Settings

Terminal set	Input pattern vector elements
	Dummy nodes
	10 floating point numbers $\in \{0 \ldots 1\}$
Function set	sqrt, log, pow2, -, sin, not
	-, +, *, /, max, min, xor, or, and
	if-then-else
Population size	200
Initial population	Half full trees, half random trees
Initial max. tree depth	5
Max. dimensionality	50
Sub-tree preservation probability	0.2
Max. no. of breeding cycles	20,000
Stopping criterion	Max generations exceeded

successor node values are larger than zero, "xor" returns zero, otherwise it returns unity.

We have used the steady-state PCGP evolutionary paradigm (Kumar & Rockett, 2002; Zhang & Rockett, 2006b) with a population of 200 individuals and terminated the optimization after 20,000 cycles of breeding pairs of offspring. This is equivalent to $\sim$100 generations of a generational GA.

3 Results and Comparisons

We re-emphasize that the method we present here is not a classifier but rather a *framework to design a classifier* based on a feature extraction stage adapted to an individual problem, followed by a Fisher linear discriminant; we neither make use of prior knowledge nor make any distributional assumptions. To justify our method, we have investigated the design process on a comprehensive range of thirteen benchmark datasets from the UCI Machine Learning database (Blake & Merz, 1998) and the StatLog project (Michie et al., 1994). For each dataset we draw a statistical comparison of the classification performance between our MMOGP algorithm and a wide range of established classifiers. The key issue is that the evolution of the feature extraction stage as well as the dimensionality of the projected decision space are driven by the notion of optimality.

The thirteen datasets from the UCI Machine Learning and Statlog databases used in this work are:

1. 2Glass - 163 instances with nine attributes - This multi-class dataset has been converted to a two-class problem by seeking to distinguish between float glass and non-float glass.
2. 6Glass (GLASS) - 214 instances with 9 attributes - This dataset comprises 6 classes. Includes the 2GLASS data above.
3. BUPA Liver Disorders – Prediction of whether a patient has a liver disorder. There

are two classes, six numerical attributes and 345 records.

4. Wisconsin Diagnostic Breast Cancer (WDBC) - This dataset has been discussed before by Mangasarian et al. (1995). 569 examples with thirty numerical attributes.

5. Pima Indians Diabetes (PID) - This dataset comprises 532 complete examples with seven attributes; all records with missing attributes were removed.

6. Wisconsin Breast Cancer (WBC) - 683 out of original 699 instances have been used; the sixteen instances with missing values were removed; each record comprises ten attributes. This dataset has been used previously in Mangasarian & Wolberg (1990).

7. Teaching Assistant Evaluation (TAE) - The data consist of evaluations of teaching performance for 151 teaching assistant with 5 attributes, classified into three types. Lim et. al .(2000) have previously used this dataset.

8. Thyroid (THY) – This dataset comprises 7200 instances with 21 attributes (15 binary, 6 continuous) and 3 classes. This dataset has been discussed by Schiffmann et al. (1992).

9. Thyroid Gland (TGD) – 215 instances and 5 attributes in 3 classes; previously used by Coomans & Broeckart (1988).

10. Wine recognition (WIN) – 179 samples with 13 continuous attributes from 3 classes. See Aeberhard et al. (1992).

11. Australian credit approval (AUS): This datasets concerns credit card applications. 690 instances exist with 14 attributes from each. There are 55.5% instances from positive decisions. This dataset has been investigated by Quinlan (1993) using decision trees.

12. Heart disease (HEA): This database contains 13 attributes, 270 samples. 120 samples present heart disease.

13. German credit (GER): Classifies people as good or bad credit risks. 1000 instances with 24 attributes. 700 have been assigned a good credit rating.

Information on the thirteen datasets is summarized in Table 2.

As the basis for assessing MMOGP, we have used eight existing (non-evolutionary) classification algorithms with implementations taken from the Weka Machine Learning system (Witten & Frank, 2005); we tuned any parameter settings as noted below, trying to obtain 'good' performance although we do not claim to have identified optimal parameters. Instead of using computationally demanding model selection methods such as cross-validation to obtain suitable parameters, we scanned the parameter values in the chosen classifier for each dataset, split into test and validation sets. If more than one parameter needed to be tuned, we determined each in sequential order. Nonetheless, we believe the comparisons with MMOGP are fair. In passing, we point-out that apart from the parameters which control the speed of evolutionary convergence, MMOGP is a parameter-free method. The comparator classifiers, which span a wide range of paradigms, were:

Table 2: The Thirteen Datasets

Name	Number of Features	Size and Distributions
2GLASS	9	163 = 87 (Float) + 76 (Non-float)
BUPA	6	345 = 200 (Benign) + 145 (Malignant)
PID	7	532 = 355 + 177 (Diabetic)
WBC	10	699 = 458 (Benign) + 241 (Malignant)
WDBC	30	569 = 357 (Benign) + 212 (Malignant)
AUS	14	690 = 383 (Positive) + 307 (Negative)
GER	24	1,000 = 700 (Positive) + 300 (Negative)
HEA	13	270 = 120 (Diseased) + 150 (Benign)
TGD	5	215 from 3 classes
WIN	13	179 from 3 classes
TAE	5	151 from 3 classes
THY	21	7,200 from 3 classes
6GLASS	9	214 from 6 classes

1. Radial Basis Functions (RBF) – A normalized Gaussian radial basis function network using the k-means clustering algorithm. We estimated the number of clusters (k) for any given dataset by taking a random split of the dataset, training the classifier on the first half and calculating a validation error on the second half. By repeating this process for different values of k we were able to identify that value of k which gave the lowest validation error for each dataset.

2. Logistic – Modified multinomial logistic regression model with a ridge estimator.

3. NNge – Nearest-neighbor-like algorithm using non-nested generalized exemplars.

4. BayesNet – Bayes Network classifier using the K2 learning algorithm.

5. IB1 – Instance-based learning algorithm which uses a simple distance measure to find the training instance closest to the given test instance and predict the same class as this training instance.

6. ADTree – The alternating decision tree learning algorithm; note that this is intrinsically a two-class classifier.

7. SMO – Sequential minimal optimization algorithm for training a support vector classifier. The complexity parameter (c) and the exponent for the polynomial kernel (e) were tuned as described above.

8. C4.5 – The well-known decision tree algorithm (which is referred to as "J48" in Weka).

3.1 Classification Performance

Insofar as making statistical comparisons between classifiers is concerned, Dietterrich (1998) has argued that N-fold cross-validation followed by a t-test is unsound as the implicit assumptions about independence are violated. He proposed an empirical 5×2 cv test although Alpaydin (1999) subsequently suggested a modification to Dietterich's test to remove the unsatisfactory aspect of the result depending on the

ordering of the folds: it is Alpaydin's F-test which we use here to statistically compare classifier performance. We perform five repetitions of splitting the dataset into two equal folds, treating one fold as the training set and the other as the test set. Repeating this process ten times allows us to compute an F-statistic from which we decide whether or not to reject the hypothesis that the performances of the two classifiers are identical. (See (Alpaydin, 1999) for further details.) Throughout this work we have used a 95% confidence level to infer a statistical difference.

The mean errors over ten folds for each of the classifiers considered and for every dataset are shown in Table 3. Notice that no results are shown for the ADTree classifier on the multi-class TGD, WIN, TAE, THY and 6GLASS datasets since ADTree is inherently a two-class classifier.

From Table 3 it is clear that the MMOGP (or MOGP) methods return the smallest mean error rates on every dataset. The statistical significance of the differences between MMOGP and all other classifiers are summarized in Table 4. In Table 4 a tick indicates that MMOGP is statistically superior to the comparator classifier while a dash denotes that any difference between the errors is not statistically significant. Of the 116 possible pairwise comparisons, MMOGP exhibits superiority in 98 and is statistically identical in the remaining 18. Most significantly, MMOGP is not bettered by *any* of the comparator classifiers over *any* of the thirteen datasets. Interestingly, in comparison to MOGP which projects the data into a 1D decision space, we observe that MMOGP is statistically superior in 7 out of the 13 tests (rightmost column of Table 4), thus reinforcing the point that the dimensionality of the decision space is an important factor to be optimized.

It is also illuminating to compare, where possible, the results of the present MMOGP method with previous evolutionary attempts at optimizing feature extractors. Table 5 contains the error rates obtained by other workers as well as our previous results on MOGP using projection to a 1D decision space. The figures in Table 5 are the averages over (varying) numbers of folds and due to differences in methodologies, it is not possible to make *statistical* comparisons on the basis of the published data. What is clear is that MMOGP yields the smallest error rates although in some cases these differences are unlikely to be statistically significant. In particular, the result of Guo & Nandi (2006) on the WDBC dataset is very little larger than ours. Unfortunately, further comparison is difficult because these authors have only used one dataset (WDBC) and include only sparse experimental details of their GP procedure. Overall, from Table 5 we can conclude that at very least, MMOGP is at least as good other methods and in reality, is likely to be significantly better than almost all other evolutionary feature extraction approaches.

The dimensionalities of the optimal solutions extracted by MMOGP for each dataset are listed in Table 6 from which it is apparent that there is some variation from one problem to the next, as one might expect. (The multiple entries in the last five rows of the table reflect the fact that these last five datasets are multi-class problems and the multiple entries are the dimensionalities of the set of trees which successively separate the classes for that particular dataset. The six class 6GLASS dataset, for example, generates six trees ranging in dimensionality from 2 to 26.) For the PID, WBC and WDBC datasets, there is a *reduction* in dimensionality compared to the

Table 3: Mean Error Comparisons of the Ten Classifiers on Thirteen Datasets from *5 x 2 cv* Test

Datasets	Classifiers									
	RBF	LOG	NNge	Bayes Net	IB1	ADTree	SMO	C4.5	MOGP	MMOGP
2GLASS	0.354	0.364	0.322	0.311	0.300	0.317	0.392	0.338	0.227	0.135
BUPA	0.442	0.383	0.449	0.420	0.388	0.343	0.423	0.391	0.264	0.215
PID	0.255	0.233	0.249	0.249	0.301	0.248	0.222	0.263	0.205	0.203
WBC	0.048	0.045	0.038	0.026	0.042	0.043	0.030	0.057	0.021	0.024
WDBC	0.061	0.068	0.077	0.054	0.046	0.052	0.030	0.067	0.026	0.024
AUS	0.182	0.130	0.176	0.139	0.202	0.153	0.162	0.162	0.126	0.121
GER	0.288	0.270	0.250	0.274	0.324	0.272	0.266	0.304	0.243	0.230
HEA	0.178	0.185	0.233	0.184	0.244	0.235	0.179	0.242	0.139	0.140
TGD	0.041	0.046	0.059	0.045	0.055	N/A	0.131	0.086	0.018	0.009
WIN	0.033	0.028	0.028	0.022	0.052	N/A	0.017	0.061	0.022	0.003
TAE	0.531	0.618	0.409	0.656	0.445	N/A	0.513	0.486	0.449	0.384
THY	0.068	0.037	0.010	0.016	0.080	N/A	0.064	0.010	0.014	0.010
6GLASS	0.363	0.439	0.350	0.390	0.340	N/A	0.435	0.348	0.289	0.261

Table 4: *F*-test comparisons between algorithms for each dataset at 95 % confidence level. A tick represesnts superiority of MMOGP over the comparator classifer/dataset combination; a dash denotes no statistical difference

Datasets	Classifiers								
	RBF	LOG	NNge	Bayes Net	IB1	ADTree	SMO	C4.5	MOGP
2GLASS	√	√	√	√	√	√	√	√	√
BUPA	√	√	√	√	√	√	√	√	√
PID	√	√	√	√	√	√	√	√	√
WBC	√	√	—	—	√	√	—	√	—
WDBC	√	√	√	√	√	√	—	√	—
AUS	√	—	√	—	√	√	√	√	—
GER	√	√	√	√	√	√	√	√	—
HEA	—	√	√	√	√	√	√	√	—
TGD	√	√	√	√	√	N/A	√	√	√
WIN	√	√	√	√	√	N/A	√	√	√
TAE	√	√	—	√	√	N/A	√	√	√
THY	√	√	—	—	√	N/A	√	√	√
6GLASS	√	√	√	√	√	N/A	√	√	√

Table 5: Comparison with Other Evolutionary GP Feature Extraction Methods by Dataset

	2GLASS	BUPA	PID	WBC	WDBC	AUS	GER
Muni et al., 2004	-	0.3007	-	0.0281	-	-	-
Bot, 2001	0.48	0.416	0.305	-	-	0.169	0.37
Bot & Langdon, 1999	0.368	-	0.25	-	-	-	-
Krawiec, 2002	0.3361	-	0.2359	-	-	-	-
Loveard & Ciesielski, 2001	-	0.308	0.242	0.032	-	-	-
Smith & Bull, 2005	-	0.3403	0.265	0.0437	0.0438	-	-
Guo & Nandi, 2006	-	-	-	-	0.025	-	-
MOGP (Zhang & Rockett, 2005b)	0.2271	0.2644	0.2057	0.0263	0.026	0.126	0.248
MMOGP	0.135	0.215	0.203	0.024	0.024	0.121	0.23

Table 6: Dimensionality of the Optimal Solutions for the Thirteen Datasets

Name	Original Features	Projected Dimensionality
2GLASS	9	19
BUPA	6	14
PID	7	6
WBC	10	6
WDBC	30	23
AUS	14	20
GER	24	34
HEA	13	13
TGD	5	23, 6, 5
WIN	13	7, 6, 4
TAE	5	7, 29, 29
THY	21	9, 21, 9
6GLASS	9	15, 26, 6, 7, 2, 6

original pattern spaces. For the HEA dataset, the dimensionality of the projected space is identical to that of the original input space; we stress, however; these two spaces are, of course, not the same. Optimal performance on the other datasets is obtained by an *increase* in dimensionality compared to the original pattern spaces. Clearly the MMOGP algorithm is constructing new, typically non-linear features which improve class discriminability.

3.2 The Pareto-front

In order to illustrate the outcome of the multiobjective optimization process, Figure 2 shows a 2D projection of a typical Pareto-front, here obtained for the BUPA dataset for one MMOGP run. Note that we are using three objectives: misclassification error, average node count and a measure of class overlap along the Fisher projection direction. The visualization of the Pareto front in this 3-space is not very clear, however, and consequently we project the Pareto front into a two-dimensional plot for ease of interpretation. This projection has the side-effect that some of the depicted points appear to be dominated. For such points, when the third objective (class overlap) is taken into account, all the points are indeed non-dominated. Figure 2 shows the non-dominated solutions judged over the training set together with the corresponding validation errors.

In Figure 2, the misclassification errors over the validation set are generally larger than those evaluated over the training set, a difference which is somewhat larger for solutions with greater average node counts. (This is exactly what the sort of relationship between in-sample error and complexity would expect in, say, a multilayer perceptron.) We observe that just because individuals are members of a Pareto set with respect to a *training set*, this does not mean that the same relationship necessarily holds with respect to the validation set.

Consistent with the findings in our previous work (Zhang & Rockett, 2006b), PCGP is able to prevent trees from bloating while providing a reasonable sampling of

the solution space. In particular, we do not see any tendency for the tree complexity objective to force all solutions to some minimal size despite the fact that the algorithm we have employed (PCGP) does not use any (explicit) diversity mechanism. In their work on multiobjective GP with a tree complexity objective, de Jong & Pollack (2003) found that unless they used a diversity preserving mechanism, all the solutions evolved to be very small. As to the reason for this discrepancy, in their original work on PCGA - on which PCGP is based - Kumar & Rockett (2002) noted that their steady-state multiobjective GA was able to yield good sampling of Pareto fronts without any of the diversity preserving mechanisms which were required by other, generational multiobjective GAs - see (Kumar & Rockett, 2002) for a full discussion. Crucially, the GP algorithm used by de Jong & Pollack (2003) was of the generational type. Hence we speculate that the reason we did not observe any drift towards minimally sized solutions with our PCGP algorithm is connected to some intrinsic property of steady-state evolutionary algorithms which preserves diversity as a natural outcome; this is clearly an area for further study.

3.3 Examples of Evolved MMOGP Trees

In this sub-section we show some examples of evolved trees which we have selected from the final non-dominated solutions of arbitrarily-chosen GP runs on the basis of having the lowest misclassification error over the validation set. This is reasonable since, in practice, our ultimate aim is a low validation error; both the Bayes error and tree complexity objectives are secondary measures used to speed convergence and prevent bloat, respectively. These typical example tree are shown in Figures 3 to 6. Leaf nodes are labeled X_n, $n \in \{1 \ldots N\}$ where there are N raw attributes in the input pattern space and all the function nodes are the elementary functions listed in Table 1. All the trees shown here are shown in the form produced by the MMOGP process and have not been pruned.

Figure 3(a) shows an evolved fourteen-dimensional tree for the HEA dataset although close inspection reveals that sub-tree 8 returns a constant value of $(0.6)^2$ and thus will contribute no discriminatory power. Our complexity objective is tending to minimize tree size but in this (and a few other cases), it does not exert quite enough selective pressure to remove minor redundancies like the one in Figure 3(a). It is clear that the GP procedure is constructing a set of new, more discriminatory features both by linear and non-linear combinations of raw pattern attributes of varying degrees of complexity. Interestingly, the raw attributes: X_1, X_5 and X_{10} are not used by this tree – clearly within the optimization framework these attributes do not possess sufficient discrimination power to warrant inclusion in the final solution. Consequently, as well as performing feature extraction, we are also gaining *feature selection* as a beneficial side-effect.

The foregoing comments about Figure 3(a) also hold for Figure 3(b) which shows an induced tree for the BUPA dataset except here, a number of raw attributes are being (re-)used in several sub-trees. For example, X_1 is used by sub-trees 7, 10, 16 and 19. This presents no conceptual problem since by analogy with principal component analysis (PCA) where raw attributes typically contribute to many eigenvectors, the tree could well be constructing more independent features in the decision space by repeated use of the same raw attributes.

Figures 4, 5 and 6 provide further examples of the generated trees for the WBC, PID and WIN datasets. The most notable point about these trees compared to those in Figure 3 is their simplicity. This is reassuring since, all other things being equal, we desire the simplest tree possible. Indeed, Figure 6(c) which discriminates class 3 of the WIN dataset uses nothing but a subset of raw attributes and in this case the optimization has delivered only feature selection.

In most cases, ready human interpretation of the evolved trees is not feasible, particularly in the case of non-linear transformations. One interesting insight, however, into the structure that the MMOGP method is identifying in the data comes from the BUPA dataset. This dataset comprises six raw attributes and aims to predict liver disorders based on five blood tests: mean corpuscular volume plus the levels of four enzymes which are associated with liver damage. The sixth attribute is the patients' daily intake of alcoholic beverages. Somewhat surprisingly, in predicting liver disease our MMOGP method discards alcohol intake during the (implicit) feature selection. We stress we have no medical qualifications but we suggest that although excessive alcohol intake is well-known to be associated with liver damage, this is indirect in the sense that excessive drinking has to be sustained for a considerable period before organ damage is apparent. The blood tests on the other hand are a *direct* measure of liver damage having occurred and are thus selected since they have greater predictive power. Alcohol intake does have sufficient predictive power to overcome the resistance to its inclusion from the parsimony objective and is hence ignored.

The inter-relationship between feature extraction and selection under our MMOGP method justifies further analysis. For the five datasets shown in Table 6, MMOGP has discarded at least one of the original pattern attributes; the rightmost column of this table shows which of the original attributes MMOGP has selected in generating the best misclassification error. We have retrained all eight of the conventional comparator classifiers on the BUPA, PID, WBC, HEA and WIN datasets but using only the subsets of attributes selected by the MMOGP procedure. For example, we have used only raw attributes 1, 2, 3, 4 and 5 from the BUPA dataset and omitted attribute 6. Our objective was to determine if MMOGP gives superior classification results solely because of judicious feature selection or whether the feature extraction operations are important. In Table 7, an error value shown in a bold typeface denotes that the classifier retrained using only the selected subset is statistically superior to the same classifier trained on the original complete attribute set. Similarly, a figure in italic type denotes that retraining that classifier with the reduced set produces a statistically worse result. Roman type denotes no statistical difference. In eight of the 39 pairwise comparisons, the subset produces superior results and an inferior result in seven although all of these degraded performances occur with the WBC dataset. In the remaining 24 comparisons there is no statistical difference. More critically, statistical comparisons of the performance of the conventional classifiers trained on the selected subsets with the corresponding results from the MMOGP method are summarized in Table 8 in which a tick denotes the superiority of MMOGP and a dash denotes no statistical difference. It is clear that MMOGP almost completely maintains its relationship to conventional classifiers trained on the selected subsets compared to the same classifiers trained on the full attribute sets. The only exceptions to this are the HEA/RBF and WBC/SMO pairings (which also displayed no difference for the

full attribute sets) and the HEA/RBF combination which has been promoted from a position of inferiority relative to MMOGP (Table 4) to being statistically identical when trained on the selected subset. Nonetheless, MMOGP still produces results which are better than or at worst, equivalent to all the comparator classifiers even when trained on selected subsets. As a consequence, we conclude that (the implicit) feature selection process alone is not sufficient to explain the superiority of the MMOGP-based classifiers and it is the combination of feature extraction together with feature selection which yields superior misclassification errors.

Over the years the pattern recognition community has expended a great deal of effort on finding a 'best' overall classifier (Lim et al. 2000; Michie et al., 1994) but this work has lead to the conclusion that no single classifier is best for every problem. This has been formalized in the 'No Free Lunch' theorems for supervised learning by Wolpert - see for example (Wolpert, 2001). In practice, when presented with a new problem, pattern recognition practitioners test the battery of classifiers at their disposal and select the one which performs best on that new problem; in the context of the No Free Lunch theorems, they are empirically determining the classifier which is the best match to the (invariably unknown) structure of the problem. Here we present a classification framework where, in effect, the *structure* of the classifier is adapted to the problem in hand by the evolutionary learning procedure. Thus we *conjecture* that our methodology is evolving a classifier which will have a performance which is comparable to the best of the set of all classifiers. This may involve the re-invention of an existing classifier but we suspect more often, involves the evolutionary tailoring of a classifier which is highly specialized to the target problem. Such a conjecture is fully consistent with the No Free Lunch theorems since we are not evolving a *single* classifier but a new, 'structurally-matched' classifier in each learning problem. We stress, however, that providing a *proof* of our conjecture will probably require significant theoretical advances in the understanding of classifiers; for the present we simply advance this plausible explanation for why our method systematically gives error rates which are either superior to, or at worst statistically identical to a comprehensive range of conventional classifiers.

Further, in this work we have employed a Fisher linear discriminant to classify the projected patterns in the decision space. Our choice of this particular classifier has been largely dictated by its speed of training; our wrapper approach requires us to train the classifier *inside* the evolutionary loop as part of the fitness evaluation of a new individual and in order not to slow the evolution, we require a fast-training algorithm. It is, however, entirely feasible to employ any classifier in our framework although it is worth pointing-out it is highly improbable that the same feature extraction stages would be evolved for an arbitrary choice of classifier as for a Fisher linear discriminant. As the classifier used is intimately connected to the optimization of the misclassification fitness objective, it is reasonable to expect the evolved features will be *conditioned* on that classifier. This again is entirely consistent with the No Free Lunch theorems; we believe our evolutionary procedure is adapting the feature extraction stages so that the *combination* of the feature extraction and the classifier are 'structurally' matched the classification problem at hand.

Table 7: Mean Error Comparisons of the Eight Conventional Classifiers on Five Datasets from 5 $\times$ 2 *cv* Test on Selected Feature Subsets. See text for an explanation of the significance of bold, roman or italic typefaces.

Datasets	Classifiers								
	RBF	LOG	NNge	Bayes Net	IB1	ADTree	SMO	C4.5	Selected Subset
BUPA	0.377	0.370	**0.359**	0.423	0.404	0.349	0.422	0.395	1,2, 3,4,5
PID	**0.226**	0.252	0.246	**0.239**	**0.278**	**0.232**	0.231	0.250	1, 2, 5, 7
WBC	*0.056*	*0.061*	*0.041*	*0.038*	*0.057*	*0.050*	*0.042*	0.050	1, 3, 4, 5, 6, 8
HEA	0.185	0.188	0.22	0.183	0.242	0.211	0.175	**0.225**	2,3,4,6,7,8,9,11, 12, 13
WIN	**0.011**	0.028	0.028	0.016	**0.040**	N/A	*0.039*	0.061	1, 3, 4, 5, 7, 8, 10, 11, 12, 13

Table 8: *F*-Test Comparisons between algorithms for each dataset at 95% confidence level. A tick denotes superiority of MMOGP against the comparator classifier/dataset combination; a dash denotes no statistical difference

MMOGP								
Datasets (Selected Features)	RBF	LOG	NNge	BayesNet	IB1	ADTree	SMO	C4.5
BUPA	√	√	√	√	√	√	√	√
PID	√	√	√	√	√	√	√	√
WBC	√	√	-	-	√	√	-	√
HEA	-	√	√	√	√	√	√	√
WIN	-	√	√	√	√	N/A	√	√

4 Conclusions

In this paper we demonstrate the use of multiobjective genetic programming to generate multi-dimensional mappings from an n-dimensional raw pattern space to an m-dimensional decision space. Crucially, not only is the near-optimal sequence of mathematical transforms evolved but also the (near-)optimal dimensionality of the decision space, m. We demonstrate the efficacy of our approach by making statistical comparisons between the results from our multi-dimensional multiobjective genetic programming (MMOGP) approach and a wide range of other classifier paradigms over thirteen publicly available datasets. We find that MMOGP produces misclassification errors which are in most cases statistically superior to the conventional classifiers and at worst, show no statistical difference. This result has great practical significance since rather than having to train a battery of classifiers on a new problem and select the best-performing, we conjecture that MMOGP is adapting the *structure* of the evolved classifier to be a good match to the classification problem. In effect, MMOGP is 'inventing' a new classifier tailored to the particular problem at hand. Moreover, we argue that this conjecture is fully consistent with the No Free Lunch theorems for supervised learning.

Finally, we conclude that although feature selection emerges as a beneficial side-effect, the superiority of MMOGP-based classifiers is not due solely to feature selection but to a combination of feature extraction and selection working in tandem.

Acknowledgments

One of us (YZ) is grateful for the financial support of a Universities UK Overseas Research Student Award Scheme (ORSAS) scholarship and the Great Britain-China Educational Trust together with Sino-British Fellowship Trust.

References

Addison, D., Wermter, S. and Arevian, G. (2003). A Comparison of Feature Extraction and Selection Techniques. In Kaynak, O., Alpaydin, E., Oja, E.and Xu, L., editors, *Proceedings of International Conference on Artificial Neural Networks (Supplementary Proceedings)*, pages 212-215, Springer-Verlag

Aeberhard, S., Coomans, D. and de Vel, O. (1992), The Classification Performance of RDA. Technical Report 92-01, Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University, North Queensland

Alpaydin, E. (1999), Combined 5 *x* 2 *cv F*-test for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 11(8):1885-1892.

Bailey, A., Class-dependent Features and Multi-category Classification, PhD. Thesis, Dept. Electronics & Computer Science, University of Southampton, UK, Feb. 2001.

Blake, C. L. and Merz, C. J. (1998). UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/∼mlearn/MLrepository.html], Irvine Ca. University of California, Department of Information & Computer Science.

Bot, M. J. C. (2001). Feature Extraction for the K-Nearest Neighbour Classifier with Genetic Programming. In *Proceedings of EuroGP 2001*, pages 256-267.

Bot, M. J. C. and Langdon, W. B. (1999). Application of Genetic Programming to Induction of Linear Classification Trees, In 11^{th} *Belgium/Netherlands Conference on Artificial Intelligence (BNAIC'99)*, pages 107-114.

Coello, C. A. C. (2000). An Updated Survey of GA-Based Multiobjective Optimization Techniques. *ACM Computing Surveys*, 32:109-143.

Coomans, D. and Broeckaert, I. (1988). Potential Pattern Recognition in Chemical and Medical Decision Making. *Research Studies Press*, Letchworth, England

de Jong, E.D. and Pollack, J.B. (2003) Multi-Objective Methods for Tree Size Control, *Genetic Programming and Evolvable Machines*, 4:211-233.

Dietterich, T.G. (1998). Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10:1895-1923.

Duda, R. O., Hart, P. E. and Stork, D. G. (2001). *Pattern Recognition*. John Wiley & Sons,

Ebner, M. (1998). On the Evolution of Interest Operators Using Genetic Programming. In Banzhaf, W. and Fogarty, T. C., editors, *Proceedings of Late breaking papers in 1 st European Workshop on Genetic Programming*, pages 6-10, Springer-Verlag

Ebner, M. and Zell, A. (1999). Evolving a Task Specific Image Operator. In *Proceedings of Joint Proceedings of the 1st European Workshop on Evolutionary Image Analysis, Signal Processing and Telecommunications (EvoIASP'99 and EuroEcTel'99)*, pages 74-89, Springer-Verlag

Ekárt, A. and Németh, S. Z. (2001). Selection Based on the Pareto Nondomination Criterion for Controlling Code Growth in Genetic Programming. *Genetic Programming and Evolvable Machines*, 2:61-73.

Guo, H., Jack, L. B. and Nandi, A. K. (2005). Feature Generation Using Genetic Programming with Application to Fault Classification. *IEEE Transactions on Systems, Man & Cybernetics - Part B*, 35:89-99.

Guo, H. and Nandi, A. K. (2006). Breast Cancer Diagnosis using Genetic Programming Generated Feature. *Pattern Recognition*, 39:980-987.

Harvey, N. R., Theiler, J., Brumby, S. P., Perkins, S., Szymanski, J. J., Bloch, J. J., Porter, R. B., Mark, G. and Young, A., C (2002). Comparison of Genie and Conventional Supervised Classifiers for Multispectral Image Feature Extraction. *IEEE Transactions on Geoscience & Remote Sensing*, 40:393-404.

Ito, T., Iba, H. and Sato, S. (1998). Non-Destructive Depth-Dependent Crossover for Genetic Programming. In *Proceedings of 1 st European Workshop on Genetic Programming*, pages 14-15.

Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer, London.

Kotani, M., Nakai, M. and Azakawa, K. (1999). Feature Extraction Using Evolutionary Computation. In *Proceedings of Congress on Evolutionary Computation*, pages 1230-1236, IEEE Press

Koza, J. R. (1994). *Genetic Programming Ii: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge, MA

Krawiec, K. (2002). Genetic Programming-Based Construction of Features for Machine Learning and Knowledge Discovery Tasks. *Genetic Programming and Evolvable Machines*, 3:329-343

Kumar, R. and Rockett, P. I. (2002). Improved Sampling of the Pareto-Front in Multiobjective Genetic Optimizations by Steady-State Evolution: A Pareto Converging Genetic Algorithm. *Evolutionary Computation*, 10:283-314.

Langdon, W. B. (1998). *Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming*. Kluwer Academic Publishers, London

Leiva-Murillo, J. M., Santiago-Mozos, R., Pérez-Cruz, F. and Artés-Rodriguez, A. (2002). Comparison of Supervised Feature Extraction Methods for Multispectral Images. *Proceedings of Learning '02*.

Lim, T., Loh, W. and Shih, Y. (2000). A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms *Machine Learning*, 40:203-228.

Loveard, T. and Ciesielski, V. (2001). Representing Classification Problems in Genetic Programming. In *Congress of Evolutionary Computing (CEC'01)*, pages 1070-1077.

Mangasarian, O. L., Street, W. N. and Wolberg, W. H. (1995). Breast Cancer Diagnosis and Prognosis Via Linear Programming. *Operations Research*, 43:570-577.

Michie, D., Spiegelhalter, D. J. and Taylor, C. C. (1994). *Machine Learning, Neural and Statistical Classification*. Prentice Hall.

Muni, D. P., Pal, N. R. and Das, J. (2004). A Novel Approach to Design Classifiers Using Genetic Programming. *IEEE Transactions on Evolutionary Computation*, 8:183-196.

Park, C. H., Park, H. and Pardalos, P. (2004). Comparative Study of Linear and Nonlinear Feature Extraction Methods. In *Proceedings of 4 th IEEE International Conference on Data Mining (ICDM'04)*, pages 495-498, IEEE Computer Society Press

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA

Schiffmann, W., Joost, M. and Werner, R. (1992), Synthesis and Performance Analysis of Multilayer Neural Network Architectures. Technical Report 16/1992, Institute für Physics, University of Koblenz, Koblenz

Sherrah, J. R., Bogner, R. E. and Bouzerdoum, A. (1997). The Evolutionary Pre-Processor: Automatic Feature Extraction for Supervised Classification Using Genetic Programming. In *Proceedings of 2 nd Annual Conference on Genetic Programming*, pages 304-312.

Smith, M.G. and Bull, L. (2005) Genetic Programming with a Genetic Algorithm for Feature Construction and Selection. *Genetic Programming and Evolvable Machines*, 6(3):265-281.

Tackett, W. A. (1993). Genetic Programming for Feature Discovery and Image Discrimination. In *Proceedings of 5 th International Conference on Genetic Algorithms*, pages 303-309, Morgan Kaufmann.

Trier, Ø. D., Jain, A. K. and Taxt, T. (1996). Feature Extraction Methods for Character Recognition - A Survey. *Pattern Recognition*, 29(4):641-662.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools*. Morgan Kaufmann,

Wolpert, D. H. (2001). The Supervised Learning No-Free-Lunch Theorems. *Proceedings of 6 th World Conference on Soft Computing (WSC6)*

Zhang, Y. and Rockett, P. I. (2005a). Evolving Optimal Feature Extraction Using Multi-Objective Genetic Programming: A Methodology and Preliminary Study on Edge Detection. In Beyer , H.-G., O'Reilly, U.-M., Arnold, D. V., Banzhaf , W., Blum , C., W, B. E., Cantu-Paz, E., Dasgupta, D., Deb, K.,Foster, J. A., de Jong, E. D., Lipson, H., Llora, X., Mancoridis, S., Pelikan, M., Raidl, G. R., Soule, T., Tyrrell, A. M., Watson, J.-P. and Zitzler, E., editors, *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2005)*, pages 795-802, ACM Press

Zhang, Y. and Rockett, P. I. (2005b). A Generic Optimal Feature Extraction Method Using Multiobjective Genetic Programming: Methodology and Applications. Submitted to *IEEE Transactions on Evolutionary Computation*

Zhang, Y. and Rockett, P. I. (2006a). Domain-Independent Approaches to Optimize Feature Extraction for Multi-Classification Using Multi-Objective Genetic Programming. Submitted to *Machine Learning*.

Zhang, Y. and Rockett, P. I. (2006b). *Feature Extraction Using Multi-Objective Genetic Programming.* In Jin, Y., editor. Multi-Objective Machine Learning, pages 75-99, Springer, Heidelberg

Zhang, Y. and Zhang, M. (2004), A Multiple Output Program Tree Structure in Genetic Programming. Technical Report CS-TR-04/14, Dept of Computer Science, School of Mathematical and Computing Sciences, Victoria University, Wellington, New Zealand

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., and da Fonseca, V.G. (2003) Performance Assessment of Multiobjective Optimizers: An Analysis and Review, *IEEE Transactions on Evolutionary Computation* 7(2):117-132.

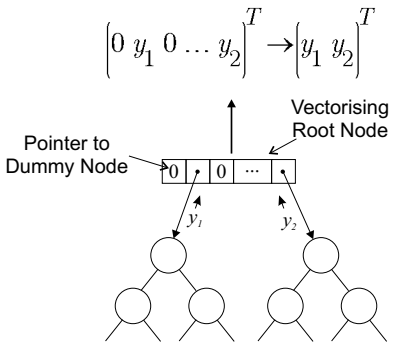

Figure 1: Illustration of the vectorizing tree chromosome structure

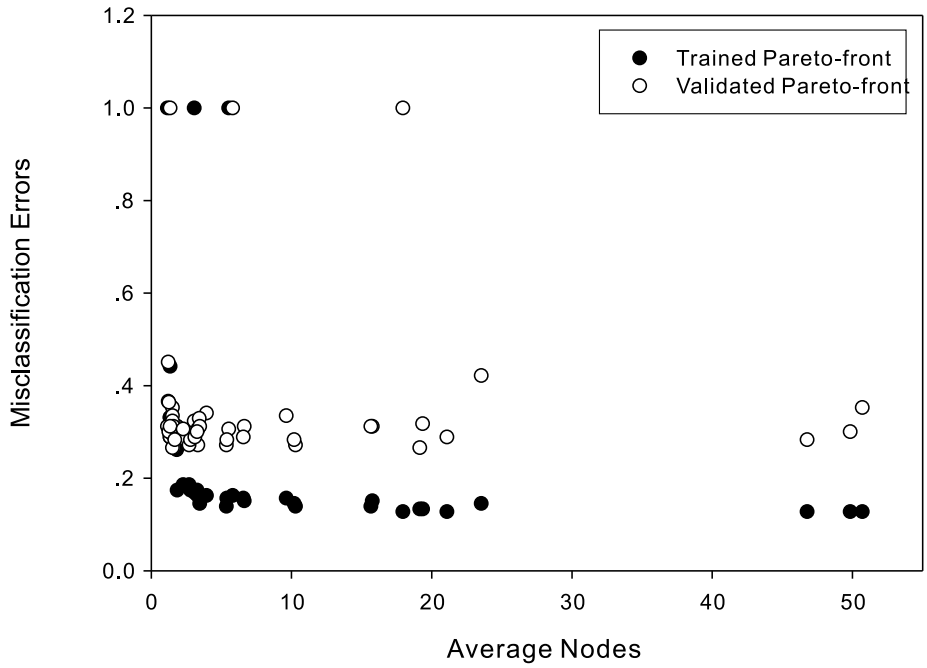

Figure 2: Converged MMOGP Pareto-front over training set and validation on BUPA dataset

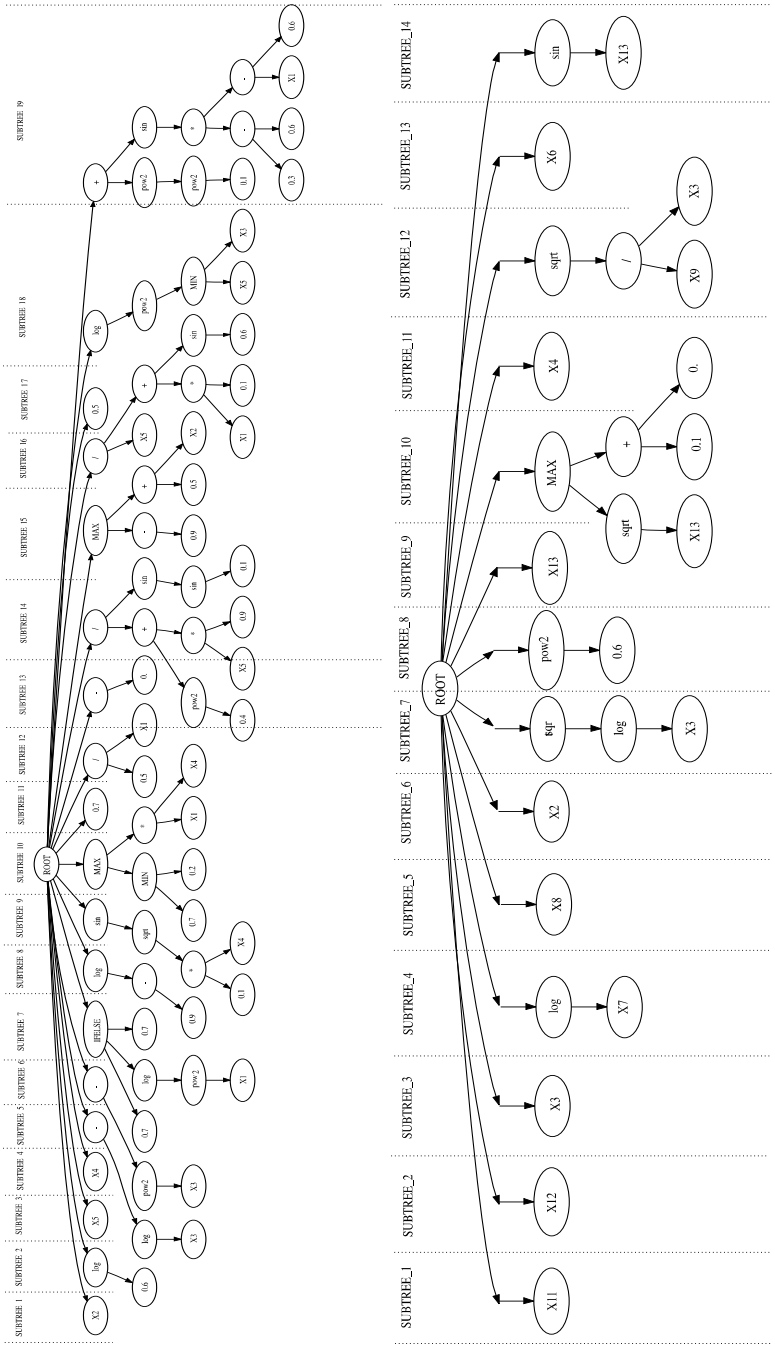

Figure 3: Example GP trees to perform multi-dimensional mappings on the BUPA and HEA datasets; Left tree is for the BUPA dataset (a), right tree for the HEA dataset (b)

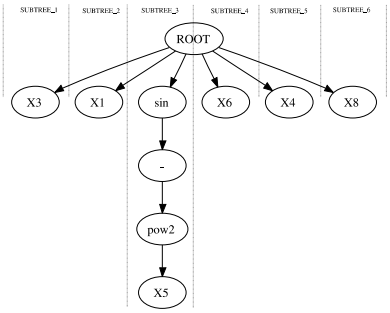

Figure 4: Example GP tree for the WBC dataset

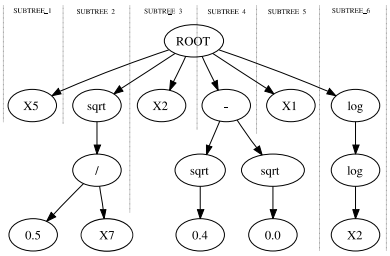

Figure 5: Example GP tree for the PID dataset

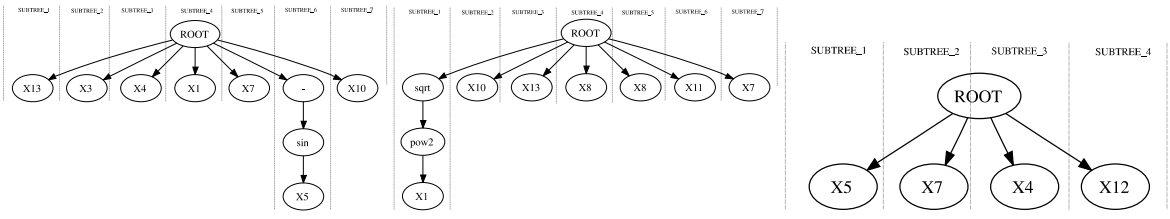

Figure 6: Typical GP trees for the 3-class WIN dataset, from left to right (a) to (c) are feature extraction trees for classes 1 , 2 and 3, respectively